SAP Business Workflow
(BC-BMT-WFM)

Release 4.6C
Copyright

© Copyright 2001 SAP AG. All rights reserved.

No part of this publication may be reproduced or transmitted in any form or for any purpose without the express permission of SAP AG. The information contained herein may be changed without prior notice.

Some software products marketed by SAP AG and its distributors contain proprietary software components of other software vendors.

Microsoft®, WINDOWS®, NT®, EXCEL®, Word®, PowerPoint® and SQL Server® are registered trademarks of Microsoft Corporation.

IBM®, DB2®, OS/2®, DB2/6000®, Parallel Sysplex®, MVS/ESA®, RS/6000®, AIX®, S/390®, AS/400®, OS/390®, and OS/400® are registered trademarks of IBM Corporation.

ORACLE® is a registered trademark of ORACLE Corporation.

INFORMIX®-OnLine for SAP and Informix® Dynamic Server™ are registered trademarks of Informix Software Incorporated.

UNIX®, X/Open®, OSF/1®, and Motif® are registered trademarks of the Open Group.

HTML, DHTML, XML, XHTML are trademarks or registered trademarks of W3C®, World Wide Web Consortium, Massachusetts Institute of Technology.

JAVA® is a registered trademark of Sun Microsystems, Inc.

JAVASCRIPT® is a registered trademark of Sun Microsystems, Inc., used under license for technology invented and implemented by Netscape.

SAP, SAP Logo, R/2, R/3, ABAP, SAP ArchiveLink, SAP Business Workflow, WebFlow, SAP EarlyWatch, BAPI, SAPPHIRE, Management Cockpit, mySAP.com Logo and mySAP.com are trademarks or registered trademarks of SAP AG in Germany and in several other countries all over the world. All other products mentioned are trademarks or registered trademarks of their respective companies.
## Icons

<table>
<thead>
<tr>
<th>Icon</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>⚠️</td>
<td>Caution</td>
</tr>
<tr>
<td>📜</td>
<td>Example</td>
</tr>
<tr>
<td>📘</td>
<td>Note</td>
</tr>
<tr>
<td>🧵</td>
<td>Recommendation</td>
</tr>
<tr>
<td>🔧</td>
<td>Syntax</td>
</tr>
<tr>
<td>💡</td>
<td>Tip</td>
</tr>
<tr>
<td>Chapter / Section</td>
<td>Page</td>
</tr>
<tr>
<td>----------------------------------------------------------------------------------</td>
<td>------</td>
</tr>
<tr>
<td>Details of Workflow Definition</td>
<td>85</td>
</tr>
<tr>
<td>Preparation and Customizing (BC-BMT-WFM)</td>
<td>86</td>
</tr>
<tr>
<td>Setting up the Organizational Plan</td>
<td>87</td>
</tr>
<tr>
<td>Performing Task-Specific Customizing</td>
<td>88</td>
</tr>
<tr>
<td>Administration Required for Shared Office Settings</td>
<td>89</td>
</tr>
<tr>
<td>Operation and Connection to Application Functionality (BC-BMT-WFM)</td>
<td>90</td>
</tr>
<tr>
<td>General Procedure Model</td>
<td>91</td>
</tr>
<tr>
<td>Role Documentation</td>
<td>93</td>
</tr>
<tr>
<td>Roles in Workflow</td>
<td>94</td>
</tr>
<tr>
<td>Process Consultant</td>
<td>96</td>
</tr>
<tr>
<td>Tutorial: Maintaining the Organizational Plan</td>
<td>98</td>
</tr>
<tr>
<td>Unit 1: Creating an Organizational Structure</td>
<td>99</td>
</tr>
<tr>
<td>Unit 2: Creating Staff Assignments</td>
<td>102</td>
</tr>
<tr>
<td>Creation of Business Process Models</td>
<td>106</td>
</tr>
<tr>
<td>Ad Hoc Workflow</td>
<td>108</td>
</tr>
<tr>
<td>Organizational Plan</td>
<td>109</td>
</tr>
<tr>
<td>Working with Screen Areas</td>
<td>111</td>
</tr>
<tr>
<td>Validity</td>
<td>114</td>
</tr>
<tr>
<td>Validity of Objects and Assignments</td>
<td>115</td>
</tr>
<tr>
<td>Selecting a Date and Preview Period</td>
<td>117</td>
</tr>
<tr>
<td>Activating/Deactivating Query Time Period</td>
<td>118</td>
</tr>
<tr>
<td>Validity of Object Characteristics</td>
<td>119</td>
</tr>
<tr>
<td>Object Manager</td>
<td>120</td>
</tr>
<tr>
<td>Finding/Selecting Objects</td>
<td>122</td>
</tr>
<tr>
<td>Using Search Tools</td>
<td>123</td>
</tr>
<tr>
<td>Using Search Variants</td>
<td>125</td>
</tr>
<tr>
<td>Displaying/Editing Objects</td>
<td>126</td>
</tr>
<tr>
<td>Creating Objects</td>
<td>129</td>
</tr>
<tr>
<td>Copying Objects</td>
<td>131</td>
</tr>
<tr>
<td>Assigning/Moving/Repositioning Objects</td>
<td>132</td>
</tr>
<tr>
<td>Terminating/Deleting Objects or Assignments</td>
<td>134</td>
</tr>
<tr>
<td>Undoing/Recreating Changes</td>
<td>135</td>
</tr>
<tr>
<td>Confirmation Prompt</td>
<td>136</td>
</tr>
<tr>
<td>Configuring Columns</td>
<td>137</td>
</tr>
<tr>
<td>Section</td>
<td>Page</td>
</tr>
<tr>
<td>------------------------------------------------------------------------</td>
<td>------</td>
</tr>
<tr>
<td>Switching Maintenance Interfaces</td>
<td>138</td>
</tr>
<tr>
<td>Business Workflow Explorer</td>
<td>139</td>
</tr>
<tr>
<td>Business Workplace: Workflow Functions</td>
<td>141</td>
</tr>
<tr>
<td>Work Item</td>
<td>144</td>
</tr>
<tr>
<td>Dialog Work Items (Type W)</td>
<td>145</td>
</tr>
<tr>
<td>Status of a Dialog Work Item</td>
<td>146</td>
</tr>
<tr>
<td>Status Transitions of a Dialog Work Item</td>
<td>148</td>
</tr>
<tr>
<td>Missed Deadline Work Item (Type D)</td>
<td>150</td>
</tr>
<tr>
<td>Status of a Missed Deadline Work Item</td>
<td>151</td>
</tr>
<tr>
<td>Status Transitions of a Missed Deadline Work Item</td>
<td>152</td>
</tr>
<tr>
<td>Work Queue Work Item (Type A)</td>
<td>153</td>
</tr>
<tr>
<td>Status of a Work Queue Work Item</td>
<td>154</td>
</tr>
<tr>
<td>Workflow Work Item (Type F)</td>
<td>155</td>
</tr>
<tr>
<td>Status of a Workflow Work Item</td>
<td>157</td>
</tr>
<tr>
<td>Status Transitions of a Workflow Work Item</td>
<td>158</td>
</tr>
<tr>
<td>Background Work Item (Type B)</td>
<td>160</td>
</tr>
<tr>
<td>Status of a Background Work Item</td>
<td>161</td>
</tr>
<tr>
<td>Error Handling for Background Work Items</td>
<td>162</td>
</tr>
<tr>
<td>Wait Step Work Item (Type E)</td>
<td>164</td>
</tr>
<tr>
<td>Work Item with Express Notification</td>
<td>165</td>
</tr>
<tr>
<td>Workflow Settings</td>
<td>166</td>
</tr>
<tr>
<td>Personal Settings</td>
<td>167</td>
</tr>
<tr>
<td>Organizational Assignment of a User</td>
<td>169</td>
</tr>
<tr>
<td>Substitutes for Workflows</td>
<td>170</td>
</tr>
<tr>
<td>Maintaining and Activating Substitutes</td>
<td>172</td>
</tr>
<tr>
<td>Views</td>
<td>174</td>
</tr>
<tr>
<td>Maintaining Views</td>
<td>175</td>
</tr>
<tr>
<td>Adopting and Exiting Views</td>
<td>176</td>
</tr>
<tr>
<td>Dynamic Columns for the Business Workplace</td>
<td>177</td>
</tr>
<tr>
<td>Selectable Columns for the Business Workplace: Workflow</td>
<td>178</td>
</tr>
<tr>
<td>Workflow Inbox</td>
<td>180</td>
</tr>
<tr>
<td>Work Item Display: Standard View</td>
<td>183</td>
</tr>
<tr>
<td>Tab Page Basic Data</td>
<td>186</td>
</tr>
<tr>
<td>Tab Page Activities</td>
<td>187</td>
</tr>
<tr>
<td>Tab Page Available Objects</td>
<td>188</td>
</tr>
</tbody>
</table>
Work Item Display: Technical View ................................................................. 190
Workflow Log: Standard View ................................................................. 192
Workflow Log: Technical View ................................................................. 194
Graphical Workflow Log .................................................................................. 196
Send Mail ........................................................................................................ 197
Display Agents .................................................................................................... 199
Changing Deadlines .............................................................................................. 200
Attachment Management ...................................................................................... 201
Reject Execution .................................................................................................... 202
Execute Together .................................................................................................. 203
Grouping According to Sort Key .......................................................................... 204
Change Priority .................................................................................................... 205
Create Link ......................................................................................................... 206
Forward Work Item ............................................................................................. 207
Workflow Relationships ....................................................................................... 208
Change Work Item ............................................................................................... 210
Set Work Item to Done ......................................................................................... 212
Workflow Outbox .................................................................................................... 213
Workflow Resubmissions ........................................................................................ 215
Work Item Preview .............................................................................................. 217
WebFlow .................................................................................................................... 218
Workflow Developer ............................................................................................... 220
Tutorial: Workflow Modeling ............................................................................... 222
  Unit 1: Organizational Structure ........................................................................ 226
  Unit 2: Creating a Workflow .............................................................................. 230
  Unit 4: Defining and Inserting Task ................................................................. 232
  Unit 3: Starting the Workflow Builder .............................................................. 236
  Unit 5: First Test ................................................................................................... 241
  Reporting and analysis .......................................................................................... 244
  Unit 6: Creating a Task ....................................................................................... 245
  Unit 7: Including .................................................................................................. 248
  Unit 8: Second Test ............................................................................................ 252
  Unit 9: Adding a User Decision ......................................................................... 254
  Unit 10: 3rd Test .................................................................................................. 256
  Unit 11: Define and Include ............................................................................... 257
<table>
<thead>
<tr>
<th>Tasks and Task Groups</th>
<th>319</th>
</tr>
</thead>
<tbody>
<tr>
<td>Creating and Maintaining Tasks</td>
<td>320</td>
</tr>
<tr>
<td>Definition of a Single-Step Task</td>
<td>322</td>
</tr>
<tr>
<td>Maintenance of Basic Data of Single-Step Task</td>
<td>324</td>
</tr>
<tr>
<td>Maintenance of Reference to Object Method</td>
<td>325</td>
</tr>
<tr>
<td>Maintenance of Execution Properties</td>
<td>326</td>
</tr>
<tr>
<td>Editing Names</td>
<td>327</td>
</tr>
<tr>
<td>Maintenance of Agent Assignment</td>
<td>328</td>
</tr>
<tr>
<td>Maintaining Task Texts</td>
<td>331</td>
</tr>
<tr>
<td>Definition of the Task Container</td>
<td>332</td>
</tr>
<tr>
<td>Creating Container Elements</td>
<td>333</td>
</tr>
<tr>
<td>Assigning Initial Values</td>
<td>334</td>
</tr>
<tr>
<td>Creating Triggering Events</td>
<td>335</td>
</tr>
<tr>
<td>Definition of Terminating Events of Task</td>
<td>336</td>
</tr>
<tr>
<td>Roles in Single-Step Tasks</td>
<td>338</td>
</tr>
<tr>
<td>Maintenance of Classification and Lock Indicators</td>
<td>339</td>
</tr>
<tr>
<td>Maintenance of SAPphone Properties</td>
<td>340</td>
</tr>
<tr>
<td>Definition of a Multistep Task</td>
<td>341</td>
</tr>
<tr>
<td>Maintenance of Basic Data of Multistep Task</td>
<td>343</td>
</tr>
<tr>
<td>Editing Names</td>
<td>344</td>
</tr>
<tr>
<td>Maintenance of Agent Assignment</td>
<td>345</td>
</tr>
<tr>
<td>Maintaining Task Texts</td>
<td>346</td>
</tr>
<tr>
<td>Definition of the Workflow Container</td>
<td>348</td>
</tr>
<tr>
<td>Creating Container Elements</td>
<td>351</td>
</tr>
<tr>
<td>Assigning Initial Values</td>
<td>352</td>
</tr>
<tr>
<td>Creating Triggering Events</td>
<td>353</td>
</tr>
<tr>
<td>Maintenance of SAPphone Properties</td>
<td>354</td>
</tr>
<tr>
<td>Definition of a Task Group</td>
<td>355</td>
</tr>
<tr>
<td>Definition of a Container</td>
<td>356</td>
</tr>
<tr>
<td>Creating Container Elements</td>
<td>359</td>
</tr>
<tr>
<td>Definition of the Workflow Container</td>
<td>360</td>
</tr>
<tr>
<td>Definition of the Task Container</td>
<td>362</td>
</tr>
<tr>
<td>Definition of the Method Container</td>
<td>363</td>
</tr>
<tr>
<td>Definition of Event Container</td>
<td>364</td>
</tr>
<tr>
<td>Definition of Binding</td>
<td>365</td>
</tr>
</tbody>
</table>
Maintenance of Tab Page Notification ......................................................... 421
Maintenance of Deadline Tab Pages ............................................................ 422
Maintenance of Tab Page Methods ............................................................... 424
Maintenance of Tab Page Work Item Display ............................................. 426
Maintenance of Tab Page Other ................................................................. 427
Maintenance of Documents from a Template ............................................. 428
 Maintenance of Tab Page Document Templates ......................................... 429
   Creating Document Templates ................................................................. 430
Maintenance of Tab Page Control ............................................................... 431
Maintenance of Tab Page Outcomes ........................................................... 432
Maintenance of Tab Page Notification ...................................................... 433
Maintenance of Deadline Tab Pages ........................................................... 434
Maintenance of Tab Page Methods ............................................................... 436
Maintenance of Tab Page Work Item Display ............................................. 438
Maintenance of Tab Page Other ................................................................. 439
Maintenance of Conditions ........................................................................ 440
Maintenance of Multiple Conditions ......................................................... 441
Maintenance of Event Creator ................................................................. 442
Maintenance of Wait Steps ......................................................................... 443
 Maintenance of Tab Page Control ................................................................. 445
 Maintenance of Tab Page Outcomes ............................................................ 446
 Maintenance of Tab Page Notification ....................................................... 447
 Maintenance of Deadline Tab Pages ........................................................... 448
Maintenance of Process Control ................................................................. 450
Maintenance of Container Operations ....................................................... 452
Maintenance of User Decisions ................................................................. 453
 Maintenance of Tab Page Decision .............................................................. 455
 Maintenance of Tab Page Control ............................................................... 456
   Creating Decision Tasks ......................................................................... 457
 Maintenance of Tab Page Outcomes ........................................................... 458
 Maintenance of Tab Page Notification ....................................................... 459
 Maintenance of Deadline Tab Pages ........................................................... 460
 Maintenance of Tab Page Methods ............................................................... 462
 Maintenance of Tab Page Work Item Display ............................................. 464
 Maintenance of Tab Page Other ................................................................. 465
Creation of Events When Change Documents are Written ................................517
  Maintaining the Assignment Between Change Document and Event Without
  Field Restrictions ................................................................................................................519
  Maintaining the Assignment Between Change Document and Event With Field
  Restrictions ........................................................................................................................521
  Maintenance of Workflow-Relevant Settings for Change Documents ..........522
  Creation of Additional Event Relevant Parameters With Event Creation........524
Creation of Events Upon Status Changes .......................................................... 526
  Maintaining the Assignment Between Status Change and Event ...............527
  Creation of Events via Message Control ............................................................ 529
  Creation of Events Upon LIS Exceptions ......................................................... 530
  Creation of Events via Business Transaction Events ....................................... 531
  Creation of Events by Calling a Function Module ............................................. 532
Definition of Event Container ............................................................................ 535
  Event Receiver ............................................................................................................ 536
  Receiver Function Module ..................................................................................... 537
  Check Function Module ....................................................................................... 539
  Receiver Type .......................................................................................................... 541
  Receiver Instance .................................................................................................... 542
  Receiver Type Function Module ........................................................................ 543
  Maintenance of Linkage Tables ........................................................................... 545
  Evaluation and Maintenance of Type Linkage Table ....................................... 546
  Evaluation and Maintenance of Instance Linkage Table ............................... 551
  Event Simulation ...................................................................................................... 555
Creation of Events for Test Purposes ................................................................. 557
  Event Trace ................................................................................................................ 558
  Role Resolution ...................................................................................................... 560
  Role .............................................................................................................................. 561
  Role Container .......................................................................................................... 564
  Role Definition .......................................................................................................... 565
  Define Roles Using Responsibilities ................................................................. 567
    Creating a Container Definition ........................................................................ 569
    Creating a Responsibility .................................................................................... 571
    Assigning Agents ............................................................................................... 572
    Defining Roles Using Organizational Data ...................................................... 573
SAP Organizational Object ................................................................. 575
Assigning SAP Organizational Objects ............................................. 577
Creating Object Assignments ............................................................. 579
Function Module .................................................................................. 581
Delimiting Object Assignments ........................................................... 582
Deleting Object Assignments ............................................................... 583
  Define Role .......................................................................................... 584
  Define Binding .................................................................................... 585
  Agent as Attribute of SAP Organizational Object ............................... 586
  Role to Determine Design Office ......................................................... 588
Defining Roles Using Function to Be Executed ....................................... 589
  Function Module for Role Resolution ............................................... 591
Interface of Function Module for Role Resolution ............................... 593
  Role to Determine Manager .............................................................. 595
  Role Resolution Using Evaluation Paths ........................................... 596
Define Role to Evaluate Evaluation Paths ............................................ 597
Define Binding ...................................................................................... 598
Role for Determining Organizational Unit of a User ............................. 600
  The Cancellation for Role Resolution Without Result Indicator .......... 601
Selecting Objects from the Organizational Plan ..................................... 602
Specify Agent, Recipient, or Administrator by His or Her Role ............ 604
Business Workplace: Workflow Functions .......................................... 605
Work Item ............................................................................................. 608
  Dialog Work Items (Type W) .............................................................. 609
  Status of a Dialog Work Item ............................................................ 610
  Status Transitions of a Dialog Work Item ......................................... 612
Missed Deadline Work Item (Type D) .................................................... 614
  Status of a Missed Deadline Work Item ............................................ 615
  Status Transitions of a Missed Deadline Work Item ....................... 616
Work Queue Work Item (Type A) .......................................................... 617
  Status of a Work Queue Work Item .................................................. 618
Workflow Work Item (Type F) ............................................................... 619
<table>
<thead>
<tr>
<th>Section</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>Forward Work Item</td>
<td>672</td>
</tr>
<tr>
<td>Workflow Relationships</td>
<td>673</td>
</tr>
<tr>
<td>Change Work Item</td>
<td>675</td>
</tr>
<tr>
<td>Set Work Item to Done</td>
<td>677</td>
</tr>
<tr>
<td>Workflow Outbox</td>
<td>678</td>
</tr>
<tr>
<td>Workflow Resubmissions</td>
<td>680</td>
</tr>
<tr>
<td>Work Item Preview</td>
<td>682</td>
</tr>
<tr>
<td>Workflow System Administrator</td>
<td>683</td>
</tr>
<tr>
<td>Administration of the Workflow System</td>
<td>684</td>
</tr>
<tr>
<td>Administration Tools</td>
<td>685</td>
</tr>
<tr>
<td>Diagnosis of Workflows with Errors</td>
<td>689</td>
</tr>
<tr>
<td>Executing Work Items Without Checking Agents</td>
<td>690</td>
</tr>
<tr>
<td>Workflow Restart After Error</td>
<td>691</td>
</tr>
<tr>
<td>Deleting Work Items</td>
<td>692</td>
</tr>
<tr>
<td>Administration of Event Queue</td>
<td>693</td>
</tr>
<tr>
<td>Maintenance of Tab Page Basic Data</td>
<td>695</td>
</tr>
<tr>
<td>Maintenance of Tab Page Activation</td>
<td>697</td>
</tr>
<tr>
<td>Maintenance of Tab Page Background Job</td>
<td>699</td>
</tr>
<tr>
<td>Maintenance of Tab Page Event Delivery</td>
<td>701</td>
</tr>
<tr>
<td>Maintenance of Tab Page Linkages With Errors</td>
<td>703</td>
</tr>
<tr>
<td>Event Queue Browser</td>
<td>704</td>
</tr>
<tr>
<td>Defining the Web Server</td>
<td>706</td>
</tr>
<tr>
<td>Displaying XML Documents</td>
<td>708</td>
</tr>
<tr>
<td>Authorization Management</td>
<td>709</td>
</tr>
<tr>
<td>Administration of Organizational Management</td>
<td>710</td>
</tr>
<tr>
<td>Workflow-Relevant Data for Organizational Management</td>
<td>713</td>
</tr>
<tr>
<td>Evaluation Paths</td>
<td>714</td>
</tr>
<tr>
<td>Maintaining Evaluation Paths</td>
<td>715</td>
</tr>
<tr>
<td>Work Item Selection</td>
<td>717</td>
</tr>
<tr>
<td>Workflow Relationships</td>
<td>720</td>
</tr>
<tr>
<td>RFC Monitor</td>
<td>722</td>
</tr>
<tr>
<td>Event Trace</td>
<td>723</td>
</tr>
<tr>
<td>Start Conditions for Workflows</td>
<td>725</td>
</tr>
<tr>
<td>Business Workplace: Workflow Functions</td>
<td>728</td>
</tr>
<tr>
<td>Work Item</td>
<td>731</td>
</tr>
</tbody>
</table>
Send Mail............................................................................................................ 785
Display Agents .................................................................................................... 787
Changing Deadlines ........................................................................................... 788
Attachment Management.................................................................................... 789
Reject Execution ................................................................................................. 790
Execute Together ............................................................................................... 791
Grouping According to Sort Key ......................................................................... 792
Change Priority ................................................................................................... 793
Create Link .......................................................................................................... 794
Forward Work Item ............................................................................................. 795
Change Work Item .............................................................................................. 796
Set Work Item to Done ....................................................................................... 798
Workflow Outbox ....................................................................................................... 799
Workflow Resubmissions .......................................................................................... 801
Work Item Preview .................................................................................................... 803
Transport and Client Copy ....................................................................................... 804
Workflow Agent ...................................................................................................................... 805
Working with the Business Workplace................................................................. 806
Business Workplace: Workflow Functions ......................................................... 807
Work Item .................................................................................................................. 810
Dialog Work Items (Type W)............................................................................... 811
  Status of a Dialog Work Item ........................................................................ 812
  Status Transitions of a Dialog Work Item ..................................................... 814
Missed Deadline Work Item (Type D)................................................................. 816
  Status of a Missed Deadline Work Item ....................................................... 817
  Status Transitions of a Missed Deadline Work Item .................................... 818
Workflow Work Item (Type F) ............................................................................. 819
  Status of a Workflow Work Item ................................................................ 821
  Status Transitions of a Workflow Work Item .............................................. 822
Background Work Item (Type B) ........................................................................ 824
  Status of a Background Work Item ............................................................. 825
  Error Handling for Background Work Items .............................................. 827
Wait Step Work Item (Type E) ............................................................................ 829
Work Item with Express Notification ................................................................... 830
Workflow Settings...................................................................................................... 831
Evaluation of Workflows ................................................................. 885
Work Item Analysis ............................................................................. 887
Workflow Relationships ........................................................................ 888
Workload Analysis ................................................................................ 890
Business Workplace: Workflow Functions ........................................... 893
Work Item ............................................................................................ 896
Dialog Work Items (Type W) ............................................................... 897
  Status of a Dialog Work Item ............................................................. 898
  Status Transitions of a Dialog Work Item ......................................... 900
Missed Deadline Work Item (Type D) ....................................................... 902
  Status of a Missed Deadline Work Item .............................................. 903
  Status Transitions of a Missed Deadline Work Item ......................... 904
Work Queue Work Item (Type A) ........................................................ 905
  Status of a Work Queue Work Item .................................................. 906
Workflow Work Item (Type F) ............................................................... 907
  Status of a Workflow Work Item ....................................................... 909
  Status Transitions of a Workflow Work Item .................................... 910
Background Work Item (Type B) ........................................................... 912
  Status of a Background Work Item .................................................... 913
  Error Handling for Background Work Items ....................................... 915
Wait Step Work Item (Type E) ............................................................... 917
Work Item with Express Notification .................................................... 918
Workflow Settings ................................................................................ 919
  Personal Settings ............................................................................... 920
Organizational Assignment of a User ..................................................... 922
Substitutes for Workflows ..................................................................... 923
  Maintaining and Activating Substitutes ............................................. 925
Views .................................................................................................... 927
  Maintaining Views ............................................................................ 928
Adopting and Exiting Views ................................................................. 929
Dynamic Columns for the Business Workplace ..................................... 930
  Selectable Columns for the Business Workplace: Workflow ............. 931
Workflow Inbox ..................................................................................... 933
  Work Item Display: Standard View .................................................. 936
  Tab Page Basic Data ......................................................................... 939
Tab Page Activities................................................................. 940
Tab Page Available Objects.................................................. 941
Work Item Display: Technical View....................................... 943
Workflow Log: Standard View............................................... 945
Workflow Log: Technical View............................................... 947
Graphical Workflow Log....................................................... 949
Send Mail.................................................................................. 950
Display Agents........................................................................ 952
Changing Deadlines............................................................... 953
Attachment Management...................................................... 954
Reject Execution...................................................................... 955
Execute Together..................................................................... 956
Grouping According to Sort Key.......................................... 957
Change Priority........................................................................ 958
Create Link.............................................................................. 959
Forward Work Item................................................................. 960
Change Work Item................................................................... 961
Set Work Item to Done............................................................ 963
Workflow Outbox...................................................................... 964
Workflow Resubmissions....................................................... 966
Work Item Preview................................................................... 968
Workflow Toolbox..................................................................... 969
Reference Documentation.......................................................... 971
BC - SAP Business Workflow - Navigation............................ 972
Business Workflow Explorer.................................................. 975
Workflow Wizard Explorer..................................................... 977
Workflow Wizards for Approval Procedures.......................... 979
  Dynamic Hierarchical Approval Workflow.......................... 980
  Extended Hierarchical Approval Workflow.......................... 982
  Release Subworkflow for FII (FIPP, BSEG).......................... 984
  Hierarchical Approval Workflow........................................ 986
  Parallel Approval Workflow............................................... 988
Workflow Wizards for Circulars.............................................. 990
  Circular.............................................................................. 991
  Dynamic Circular............................................................. 993
Dynamic Parallel Circular .......................................................... 995
Parallel Circular ........................................................................ 997
Other Workflow Wizards ........................................................... 999
  Creation of Customizing Workflows ....................................... 1000
  Simplified Workflow Definition ............................................ 1001
Workflow Builder ....................................................................... 1002
  Calling the Workflow Builder .................................................. 1004
Step Types and Their Symbols ................................................ 1005
Workflow Builder Functions ..................................................... 1006
  Create Step ............................................................................ 1008
Special Functions in the Workflow Builder ............................... 1009
  Inserting Multistep Tasks ...................................................... 1010
  Printing Workflow Definitions ............................................. 1011
Condition Editor ....................................................................... 1012
Workflow Definition Extras ...................................................... 1016
Definition of Workflows .......................................................... 1018
  Versions of a Workflow Definition ....................................... 1019
  Basic Data of a Workflow Definition ..................................... 1021
Using Wizards ......................................................................... 1023
Maintenance of Activities .......................................................... 1024
  Maintenance of Tab Page Control ....................................... 1026
  Maintenance of Tab Page Outcomes .................................. 1028
  Maintenance of Tab Page Notification ............................... 1030
  Maintenance of Deadline Tab Pages .................................. 1031
  Maintenance of Tab Page Methods ..................................... 1033
  Maintenance of Tab Page Work Item Display .................... 1035
  Maintenance of Tab Page Other ......................................... 1036
  Maintenance of a Web Activity ........................................... 1037
  Maintenance of Tab Page Call ............................................. 1038
  Maintenance of Tab Page Control ..................................... 1040
  Maintenance of Tab Page Outcomes .................................. 1041
  Maintenance of Tab Page Notification ............................... 1042
  Maintenance of Deadline Tab Pages .................................. 1043
  Maintenance of Documents from a Template .................... 1045
  Maintenance of Tab Page Document Templates ............... 1046
WebFlow ...............................................................................................................................1091
  Creation of a Wf-XML Document ..................................................................................1093
Ad Hoc Workflow .............................................................................................................1095
  Defining Ad Hoc Agent Assignment .........................................................................1096
  Executing Workflows with Ad Hoc Agent Assignment ............................................1097
  Defining Workflows Ad Hoc .........................................................................................1098
  Defining Task Groups for Ad Hoc Workflows ..........................................................1099
Business Object Builder ..................................................................................................1100
  Maintenance of Object Types ......................................................................................1101
    Creating Object Types ...............................................................................................1102
    Editing the Basic Data of an Object Type .................................................................1103
    Processing Object Types .........................................................................................1104
  Definition of Object Type Components ......................................................................1106
    Maintenance of an Object Type Component ..........................................................1107
    Displaying Details of an Object Type Component ..................................................1109
  Implementation of Interfaces .......................................................................................1110
    Interface Aggregate .................................................................................................1111
    Interface Approval ....................................................................................................1112
    Interface Extended Approval ..................................................................................1113
    Interface Create ........................................................................................................1114
    Interface Display ......................................................................................................1115
    Interface Edit ............................................................................................................1116
    Interface Check Existence .......................................................................................1117
    Interface Find ..........................................................................................................1118
  Standard Interface .......................................................................................................1119
    Interface Status Management ..................................................................................1120
    Interface Structure Object .......................................................................................1121
    Defining Key Fields .................................................................................................1122
    Definition of Attributes for an Object Type ............................................................1123
    Definition of Methods for an Object Type ...............................................................1125
    Definition of Method Parameters ............................................................................1127
    Definition of Exceptions ..........................................................................................1128
    Definition of Events for an Object Type ................................................................1129
    Definition of Event Parameters .............................................................................1130
  Programming in the Implementation Program ............................................................1131
<table>
<thead>
<tr>
<th>Section</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>Macro Instructions for Processing a Container</td>
<td>1133</td>
</tr>
<tr>
<td>Declaration and Initialization of a Container</td>
<td>1137</td>
</tr>
<tr>
<td>Writing a Field Value into a Container</td>
<td>1138</td>
</tr>
<tr>
<td>Writing a Table into a Container</td>
<td>1139</td>
</tr>
<tr>
<td>Writing an Object Reference into a Container</td>
<td>1140</td>
</tr>
<tr>
<td>Reading an Object Reference from a Container</td>
<td>1142</td>
</tr>
<tr>
<td>Writing an Multiline Object Reference into a Container</td>
<td>1143</td>
</tr>
<tr>
<td>Reading a Multiline Object Reference from a Container</td>
<td>1145</td>
</tr>
<tr>
<td>Macro Instructions for Accessing Objects, Attributes and Methods</td>
<td>1146</td>
</tr>
<tr>
<td>Data Declaration for Variables Used</td>
<td>1149</td>
</tr>
<tr>
<td>Programming Asynchronous Methods</td>
<td>1150</td>
</tr>
<tr>
<td>Programming Synchronous Methods</td>
<td>1152</td>
</tr>
<tr>
<td>Programming Methods with Parameters</td>
<td>1154</td>
</tr>
<tr>
<td>Programming Exceptions</td>
<td>1156</td>
</tr>
<tr>
<td>Extending and Adapting Object Types</td>
<td>1157</td>
</tr>
<tr>
<td>Creating a Subtype</td>
<td>1158</td>
</tr>
<tr>
<td>Defining Delegation Types</td>
<td>1159</td>
</tr>
<tr>
<td>Object Type SELFITEM</td>
<td>1160</td>
</tr>
<tr>
<td>Method SendTaskDescription</td>
<td>1162</td>
</tr>
<tr>
<td>Object Type WF_TASK</td>
<td>1163</td>
</tr>
<tr>
<td>Business Object Repository</td>
<td>1167</td>
</tr>
<tr>
<td>Business Object Repository Browser</td>
<td>1169</td>
</tr>
<tr>
<td>Workflow Configuration</td>
<td>1171</td>
</tr>
<tr>
<td>Tasks and Task Groups</td>
<td>1172</td>
</tr>
<tr>
<td>Creating and Maintaining Tasks</td>
<td>1173</td>
</tr>
<tr>
<td>Definition of a Single-Step Task</td>
<td>1175</td>
</tr>
<tr>
<td>Maintenance of Basic Data of Single-Step Task</td>
<td>1177</td>
</tr>
<tr>
<td>Maintenance of Reference to Object Method</td>
<td>1178</td>
</tr>
<tr>
<td>Maintenance of Execution Properties</td>
<td>1179</td>
</tr>
<tr>
<td>Editing Names</td>
<td>1180</td>
</tr>
<tr>
<td>Maintenance of Agent Assignment</td>
<td>1181</td>
</tr>
<tr>
<td>Maintaining Task Texts</td>
<td>1184</td>
</tr>
<tr>
<td>Definition of the Task Container</td>
<td>1185</td>
</tr>
<tr>
<td>Creating Container Elements</td>
<td>1186</td>
</tr>
<tr>
<td>Assigning Initial Values</td>
<td>1187</td>
</tr>
</tbody>
</table>

April 2001
<table>
<thead>
<tr>
<th>Section</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>Defining the Standard Task</td>
<td>1239</td>
</tr>
<tr>
<td>Defining the Standard Task</td>
<td>1240</td>
</tr>
<tr>
<td>Defining the Workflow (Section)</td>
<td>1241</td>
</tr>
<tr>
<td>Specifying Agents of Subsequent Steps Dynamically</td>
<td>1242</td>
</tr>
<tr>
<td>Copying a Selection Task</td>
<td>1244</td>
</tr>
<tr>
<td>Copying Tasks into Another Plan Version</td>
<td>1245</td>
</tr>
<tr>
<td>Organizational Plan</td>
<td>1246</td>
</tr>
<tr>
<td>Working with Screen Areas</td>
<td>1248</td>
</tr>
<tr>
<td>Validity</td>
<td>1251</td>
</tr>
<tr>
<td>Validity of Objects and Assignments</td>
<td>1252</td>
</tr>
<tr>
<td>Selecting a Date and Preview Period</td>
<td>1254</td>
</tr>
<tr>
<td>Activating/Deactivating Query Time Period</td>
<td>1255</td>
</tr>
<tr>
<td>Validity of Object Characteristics</td>
<td>1256</td>
</tr>
<tr>
<td>Object Manager</td>
<td>1257</td>
</tr>
<tr>
<td>Finding/Selecting Objects</td>
<td>1259</td>
</tr>
<tr>
<td>Using Search Tools</td>
<td>1260</td>
</tr>
<tr>
<td>Using Search Variants</td>
<td>1262</td>
</tr>
<tr>
<td>Displaying/Editing Objects</td>
<td>1263</td>
</tr>
<tr>
<td>Creating Objects</td>
<td>1266</td>
</tr>
<tr>
<td>Copying Objects</td>
<td>1268</td>
</tr>
<tr>
<td>Assigning/Moving/Repositioning Objects</td>
<td>1269</td>
</tr>
<tr>
<td>Terminating/Deleting Objects or Assignments</td>
<td>1271</td>
</tr>
<tr>
<td>Undoing/Recreating Changes</td>
<td>1272</td>
</tr>
<tr>
<td>Confirmation Prompt</td>
<td>1273</td>
</tr>
<tr>
<td>Configuring Columns</td>
<td>1274</td>
</tr>
<tr>
<td>Switching Maintenance Interfaces</td>
<td>1275</td>
</tr>
<tr>
<td>Role Resolution</td>
<td>1276</td>
</tr>
<tr>
<td>Role</td>
<td>1277</td>
</tr>
<tr>
<td>Role Container</td>
<td>1280</td>
</tr>
<tr>
<td>Role Definition</td>
<td>1281</td>
</tr>
<tr>
<td>Define Roles Using Responsibilities</td>
<td>1283</td>
</tr>
<tr>
<td>Creating a Container Definition</td>
<td>1285</td>
</tr>
<tr>
<td>Creating a Responsibility</td>
<td>1287</td>
</tr>
<tr>
<td>Assigning Agents</td>
<td>1288</td>
</tr>
<tr>
<td>Defining Roles Using Organizational Data</td>
<td>1289</td>
</tr>
</tbody>
</table>
Creation of Events via Business Transaction Events .............................................1339
Creation of Events by Calling a Function Module ..................................................1340
Definition of Event Container ..................................................................................1343
Event Receiver .........................................................................................................1344
  Receiver Function Module ....................................................................................1345
  Check Function Module .......................................................................................1347
Receiver Type ..........................................................................................................1349
  Receiver Instance ................................................................................................1350
Receiver Type Function Module .............................................................................1351
Maintenance of Linkage Tables ..............................................................................1353
  Evaluation and Maintenance of Instance Linkage Table ......................................1354
  Evaluation and Maintenance of Type Linkage Table ............................................1358
Event Simulation .....................................................................................................1363
Creation of Events for Test Purposes .................................................................1365
Event Trace ..........................................................................................................1366
Business Workplace: Workflow Functions .............................................................1368
  Work Item ...........................................................................................................1371
    Dialog Work Items (Type W) .........................................................................1372
    Status of a Dialog Work Item ........................................................................1373
    Status Transitions of a Dialog Work Item ......................................................1375
    Missed Deadline Work Item (Type D) ............................................................1377
    Status of a Missed Deadline Work Item .......................................................1378
    Status Transitions of a Missed Deadline Work Item ....................................1379
    Workflow Work Item (Type F) .......................................................................1380
    Status of a Workflow Work Item ..................................................................1382
    Status Transitions of a Workflow Work Item ..............................................1383
    Background Work Item (Type B) ..................................................................1385
    Status of a Background Work Item ..............................................................1386
    Error Handling for Background Work Items ..............................................1388
    Work Queue Work Item (Type A) .................................................................1390
    Status of a Work Queue Work Item ............................................................1391
    Wait Step Work Item (Type E) ....................................................................1392
    Work Item with Express Notification ..........................................................1393
Workflow Settings .................................................................................................1394
  Personal Settings ...............................................................................................1395
Organizational Assignment of a User ................................................................. 1397
Substitutes for Workflows ..................................................................................... 1398
Maintaining and Activating Substitutes ................................................................. 1400
Views .................................................................................................................... 1402
Maintaining Views ............................................................................................... 1403
Adopting and Exiting Views .................................................................................. 1404
Dynamic Columns for the Business Workplace................................................... 1405
Selectable Columns for the Business Workplace: Workflow .............................. 1406
Workflow Inbox ..................................................................................................... 1408
Work Item Display: Standard View ....................................................................... 1411
Tab Page Basic Data ............................................................................................. 1414
Tab Page Activities ............................................................................................... 1415
Tab Page Available Objects .................................................................................. 1416
Work Item Display: Technical View ...................................................................... 1418
Workflow Log: Standard View ............................................................................... 1420
Workflow Log: Technical View .............................................................................. 1422
Graphical Workflow Log ...................................................................................... 1424
Send Mail ............................................................................................................... 1425
Display Agents ...................................................................................................... 1427
Changing Deadlines .............................................................................................. 1428
Attachment Management ..................................................................................... 1429
Reject Execution .................................................................................................. 1430
Execute Together ................................................................................................. 1431
Grouping According to Sort Key .......................................................................... 1432
Change Priority .................................................................................................... 1433
Create Link ........................................................................................................... 1434
Forward Work Item .............................................................................................. 1435
Workflow Relationships ....................................................................................... 1436
Change Work Item ................................................................................................ 1438
Set Work Item to Done ......................................................................................... 1440
Workflow Outbox .................................................................................................. 1441
Workflow Resubmissions ...................................................................................... 1443
Work Item Preview ............................................................................................... 1445
Workflow Toolbox ................................................................................................. 1446
Starting Tasks and Workflows ............................................................................. 1448
<table>
<thead>
<tr>
<th>Topic</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>Starting Workflows in Dialog</td>
<td>1450</td>
</tr>
<tr>
<td>Starting Workflows (Test Environment)</td>
<td>1452</td>
</tr>
<tr>
<td>Advancing with Immediate Dialog</td>
<td>1453</td>
</tr>
<tr>
<td>Defining Start Transactions</td>
<td>1455</td>
</tr>
<tr>
<td>Starting Workflows or Tasks from Message Long Texts</td>
<td>1456</td>
</tr>
<tr>
<td>Start Conditions for Workflows</td>
<td>1458</td>
</tr>
<tr>
<td>Remote Worklist</td>
<td>1461</td>
</tr>
<tr>
<td>Creating a Logical RFC Destination</td>
<td>1463</td>
</tr>
<tr>
<td>Specifying Subscribers</td>
<td>1464</td>
</tr>
<tr>
<td>Scheduling a Report</td>
<td>1466</td>
</tr>
<tr>
<td>Workflow Information System</td>
<td>1467</td>
</tr>
<tr>
<td>Using the Workflow Information System (WIS)</td>
<td>1468</td>
</tr>
<tr>
<td>Key Figures Used</td>
<td>1470</td>
</tr>
<tr>
<td>Standard Analysis Organization View</td>
<td>1471</td>
</tr>
<tr>
<td>Standard Analysis Process View</td>
<td>1472</td>
</tr>
<tr>
<td>Standard Analysis Object View</td>
<td>1473</td>
</tr>
<tr>
<td>Standard Analysis Group View</td>
<td>1474</td>
</tr>
<tr>
<td>Standard Analysis Example Scenario</td>
<td>1476</td>
</tr>
<tr>
<td>Periodic Data Transfer</td>
<td>1477</td>
</tr>
<tr>
<td>Creating a Variant for the Report RMCADATA</td>
<td>1478</td>
</tr>
<tr>
<td>Scheduling a Job</td>
<td>1479</td>
</tr>
<tr>
<td>Making Customer-Specific Enhancements to the WIS</td>
<td>1480</td>
</tr>
<tr>
<td>Extending the Communication Structure</td>
<td>1481</td>
</tr>
<tr>
<td>Retrieving Data for Customer Fields</td>
<td>1482</td>
</tr>
<tr>
<td>LIS Customizing</td>
<td>1483</td>
</tr>
<tr>
<td>Work Item Analysis</td>
<td>1484</td>
</tr>
<tr>
<td>Workload Analysis</td>
<td>1485</td>
</tr>
<tr>
<td>Task Analysis</td>
<td>1488</td>
</tr>
<tr>
<td>Work Item Selection</td>
<td>1490</td>
</tr>
<tr>
<td>Error Diagnosis</td>
<td>1493</td>
</tr>
<tr>
<td>Consistency Check</td>
<td>1495</td>
</tr>
<tr>
<td>Workflow Trace</td>
<td>1496</td>
</tr>
<tr>
<td>RFC Monitor</td>
<td>1498</td>
</tr>
<tr>
<td>Demo and Test Workflows</td>
<td>1499</td>
</tr>
<tr>
<td>Administration Tools</td>
<td>1500</td>
</tr>
</tbody>
</table>
Verifying Workflow Customizing ................................................................. 1504
Customizing for Tasks and Workflows ...................................................... 1506
Diagnosis of Workflows with Errors ......................................................... 1507
Executing Work Items Without Checking Agents ...................................... 1508
Workflow Restart After Error ..................................................................... 1509
Deleting Work Items ................................................................................ 1510
Displaying the Work Item Container ......................................................... 1511
Displaying XML Documents ...................................................................... 1512
Defining the Web Server ........................................................................... 1513
Administration of Event Queue ............................................................... 1515
  Maintenance of Tab Page Basic Data ..................................................... 1517
  Maintenance of Tab Page Activation ..................................................... 1519
  Maintenance of Tab Page Background Job ............................................ 1521
  Maintenance of Tab Page Event Delivery ............................................. 1523
  Maintenance of Tab Page Linkages With Errors ................................... 1525
  Event Queue Browser ........................................................................... 1526
Transport and Client Copy ........................................................................ 1528
Authorization Management ......................................................................... 1529
Administration of Organizational Management ........................................ 1530
  Workflow-Relevant Data for Organizational Management .................. 1533
  Evaluation Paths .................................................................................. 1534
    Maintaining Evaluation Paths ......................................................... 1535
SAP Business Workflow via the Internet .................................................. 1537
  Creating a Customer Master Record on the Internet (BC-BMT-WFM SAPforms) .... 1538
    Process Flow (BC-BMT-WFM Creating a Customer Master Record) ......... 1541
Business Workplace via the Internet ........................................................ 1542
  Business Workplace ............................................................................ 1543
  Displaying and Processing Work Items .............................................. 1544
  Creating, Processing and Sending Documents .................................... 1545
Workflow Status Reports via the Internet (BC-BMT-WFM) .......................... 1546
  Execution of Selection Reports (BC-BMT-WFM Status Reports) .......... 1549
Web Form Integration ............................................................................ 1551
Starting Workflows with Web Forms ....................................................... 1552
Designing a Web Form to Start a Task ................................................... 1553
  Defining a Task to be Started with a Web Form ................................... 1554
Unit 17: Including .................................................................1631
Unit 18: Work Item Attachments ...........................................1633
  Unit 19: Fifth Test ...............................................................1635
Unit 20: Monitoring Missed Deadlines ..................................1637
Tutorial: Maintaining the Organizational Plan .........................1639
  Unit 1: Creating an Organizational Structure .........................1640
  Unit 2: Creating Staff Assignments .......................................1643
Tutorial: Event Creation During Status Changes .......................1647
  Unit 1: Testing the Existing Application Functionality ..............1649
  Unit 2: Identifying and Extending Object Types ......................1651
  Unit 3: Adding Further Attributes .......................................1654
  Unit 4: Flexible Event Creation ..........................................1657
  Unit 5: Finding Agents - Organizational Structure ................1659
  Unit 6: Defining a Workflow .............................................1661
  Unit 7: Editing Workflow Definitions ....................................1663
  Unit 8: Executing the Scenario ............................................1666
Tutorial: Workflow Programming ............................................1667
  Extending Object Types: Inheritance and Delegation ..............1669
  Unit 1: Creating an Object Type in the Business Object Repository 1671
  Unit 2: Creating Database Field Attributes ..........................1675
  Unit 3: Creating Object Database Field Attributes ................1677
  Unit 4: Creating Virtual Attributes ......................................1679
  Unit 5: Creating Object Virtual Attributes ............................1681
  Unit 6: Creating Multiline Virtual Attributes .........................1683
  Unit 7: Creating the Method Edit Without Parameters ............1686
  Unit 8: Creating the ExistenceCheck Method Without Parameters 1689
  Unit 9: Creating the Display Method Without Parameters ........1691
  Unit 10: Creating a Method with Parameters .........................1693
  Unit 11: Events and Their Creation .....................................1696
  Unit 12: Basic Data ..........................................................1698
  Unit 13: Final Steps ..........................................................1699
Appendix: Program for Object Type Z_BUS2032 .........................1700
BC - Workflow Scenarios in Applications (BC-BMT-WFM) ............1703
Aim and Structure of this Documentation ..................................1705
Settings and Preparations ......................................................1706
<table>
<thead>
<tr>
<th>Topic</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>Organizational Plan</td>
<td>1708</td>
</tr>
<tr>
<td>Customizing for Tasks and Workflows</td>
<td>1710</td>
</tr>
<tr>
<td>SAP Business Workflow</td>
<td>1711</td>
</tr>
<tr>
<td>Technical Principles of SAP Business Workflow</td>
<td>1712</td>
</tr>
<tr>
<td>General Procedure Model</td>
<td>1714</td>
</tr>
<tr>
<td>Transport Workflow</td>
<td>1716</td>
</tr>
<tr>
<td>Form Integration with SAPforms (BC-BMT-WFM)</td>
<td>1718</td>
</tr>
<tr>
<td>Forms</td>
<td>1721</td>
</tr>
<tr>
<td>Electronic Form Integration</td>
<td>1722</td>
</tr>
<tr>
<td>Installing SAPforms</td>
<td>1725</td>
</tr>
<tr>
<td>SAPforms - Administration</td>
<td>1727</td>
</tr>
<tr>
<td>Designing Electronic Forms</td>
<td>1729</td>
</tr>
<tr>
<td>Designing Forms with Visual Basic</td>
<td>1730</td>
</tr>
<tr>
<td>Designing Forms with MS Outlook</td>
<td>1731</td>
</tr>
<tr>
<td>Designing Forms with MS Word and MS Excel</td>
<td>1733</td>
</tr>
<tr>
<td>SAPforms Designer</td>
<td>1735</td>
</tr>
<tr>
<td>Getting Started</td>
<td>1739</td>
</tr>
<tr>
<td>SAPforms Helper</td>
<td>1741</td>
</tr>
<tr>
<td>SAP Trace Log Viewer</td>
<td>1742</td>
</tr>
<tr>
<td>Maintaining Plug-Ins and Opening a Structure</td>
<td>1744</td>
</tr>
<tr>
<td>Plug-ins</td>
<td>1746</td>
</tr>
<tr>
<td>Submit Control</td>
<td>1747</td>
</tr>
<tr>
<td>AfterSubmit (Method)</td>
<td>1750</td>
</tr>
<tr>
<td>BeforeSubmit (Method)</td>
<td>1751</td>
</tr>
<tr>
<td>Custom (Property)</td>
<td>1752</td>
</tr>
<tr>
<td>ErrorMessage (Property)</td>
<td>1753</td>
</tr>
<tr>
<td>SuccessMessage (Property)</td>
<td>1754</td>
</tr>
<tr>
<td>SAPForm Plug-In Type</td>
<td>1755</td>
</tr>
<tr>
<td>SAPforms Form Plug-In for Visual Basic</td>
<td>1756</td>
</tr>
<tr>
<td>Link Class</td>
<td>1757</td>
</tr>
<tr>
<td>Load (Method)</td>
<td>1758</td>
</tr>
<tr>
<td>Links Class</td>
<td>1762</td>
</tr>
</tbody>
</table>
Save (Method) .....................................................................................................................1763
SAPCtrl Class .....................................................................................................................1767
Name (Property) .................................................................................................................1768
Value (Property) ..................................................................................................................1770
SAPForm Class ..................................................................................................................1772
Execute (Method) ...............................................................................................................1774
GetData (Method) ...............................................................................................................1777
Load (Method) .....................................................................................................................1778
LogIt (Method) ....................................................................................................................1782
Options (Method) ................................................................................................................1783
PutData (Method) ................................................................................................................1785
SAPData (Property) ............................................................................................................1786
SAPXmit (Property) ............................................................................................................1787
Save (Method) .....................................................................................................................1788
Start (Method) .....................................................................................................................1792
Submit (Method) ................................................................................................................1795
SAPforms Form Plug-In for MS Outlook ..........................................................................1796
Link Class ............................................................................................................................1797
Load (Method) .....................................................................................................................1798
Links Class ..........................................................................................................................1802
Save (Method) .....................................................................................................................1803
SAPCtrl Class .....................................................................................................................1807
Name (Property) .................................................................................................................1808
Value (Property) ..................................................................................................................1810
SAPForm Class ..................................................................................................................1812
AddCbControl (Method) .................................................................................................1813
Execute (Method) ................................................................. 1814
GetData (Method ) .............................................................. 1817
Load (Method) ...................................................................... 1818
LogIt (Method) ...................................................................... 1822
Options (Method) ............................................................... 1823
PutData (Method) ............................................................... 1825
SAPData (Property) ............................................................ 1826
SAPXmit (Property) ............................................................ 1827
Save (Method) ...................................................................... 1828
Start (Method) ..................................................................... 1832
Submit (Method) ............................................................... 1835
UpdateInstance (Method) .................................................... 1836
SAPXmit Class ....................................................................... 1837
Execute (Method) ................................................................. 1838
Load (Method) ...................................................................... 1841
MailItem (Property) ............................................................ 1845
Options (Method) ............................................................... 1846
Save (Method) ...................................................................... 1848
Start (Method) ..................................................................... 1852
Subject (Property) .............................................................. 1855
SAPforms Form Plug-In for MS Office ............................. 1856
Link Class ............................................................................. 1857
Load (Method) ...................................................................... 1858
Links Class ........................................................................... 1862
Save (Method) ...................................................................... 1863
SAPCtrl Class ....................................................................... 1867
<table>
<thead>
<tr>
<th>Method/Property</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>Name (Property)</td>
<td>1868</td>
</tr>
<tr>
<td>Value (Property)</td>
<td>1870</td>
</tr>
<tr>
<td>SAPForm Class</td>
<td>1872</td>
</tr>
<tr>
<td>Execute (Method)</td>
<td>1873</td>
</tr>
<tr>
<td>GetData (Method)</td>
<td>1876</td>
</tr>
<tr>
<td>Load (Method)</td>
<td>1877</td>
</tr>
<tr>
<td>LogIt (Method)</td>
<td>1881</td>
</tr>
<tr>
<td>Options (Method)</td>
<td>1882</td>
</tr>
<tr>
<td>PutData (Method)</td>
<td>1884</td>
</tr>
<tr>
<td>SAPData (Property)</td>
<td>1885</td>
</tr>
<tr>
<td>SAPXmit (Property)</td>
<td>1886</td>
</tr>
<tr>
<td>Save (Method)</td>
<td>1887</td>
</tr>
<tr>
<td>Start (Method)</td>
<td>1891</td>
</tr>
<tr>
<td>Submit (Method)</td>
<td>1894</td>
</tr>
<tr>
<td>SAPData Plug-In Type</td>
<td>1895</td>
</tr>
<tr>
<td>SAPforms Data Plug-In for Workflow</td>
<td>1897</td>
</tr>
<tr>
<td>Field Class</td>
<td>1898</td>
</tr>
<tr>
<td>Load (Method)</td>
<td>1899</td>
</tr>
<tr>
<td>Name (Property)</td>
<td>1903</td>
</tr>
<tr>
<td>Save (Method)</td>
<td>1905</td>
</tr>
<tr>
<td>Fields Class</td>
<td>1909</td>
</tr>
<tr>
<td>Load (Method)</td>
<td>1910</td>
</tr>
<tr>
<td>Save (Method)</td>
<td>1914</td>
</tr>
<tr>
<td>SAPData Class</td>
<td>1918</td>
</tr>
<tr>
<td>Connection (Property)</td>
<td>1920</td>
</tr>
<tr>
<td>Execute (Method)</td>
<td>1921</td>
</tr>
</tbody>
</table>
Fields (Property).................................................................................................................1924
Load (Method) .....................................................................................................................1925
LoadInstance (Method) ......................................................................................................1929
LogIt (Method) ....................................................................................................................1931
Name (Property) .................................................................................................................1932
Offline (Property) ............................................................................................................. ...1934
Options (Method) ................................................................................................................1935
R3DoNot Use (Property) .....................................................................................................1937
Receiver (Property) ............................................................................................................1938
SAPXmit (Property) .............................................................................................................1939
Save (Method) .....................................................................................................................1940
SaveInstance (Method) ......................................................................................................1944
Start (Method) .....................................................................................................................1945
SuppressSystem Selection (Property)..............................................................................1948
UpdateInstance (Method) ..................................................................................................1949
Value (Property)..................................................................................................................1950
Version (Property)..............................................................................................................1952
WorkitemID (Property).......................................................................................................1953
SAPforms Data Plug-In for Function Modules .................................................................1954
Field Class...........................................................................................................................1955
Load (Method) .....................................................................................................................1956
Name (Property)..................................................................................................................1960
Save (Method) .....................................................................................................................1962
Fields Class ..........................................................................................................................1966
Load (Method) .....................................................................................................................1967
Save (Method) .....................................................................................................................1971
<table>
<thead>
<tr>
<th>Function</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>SAPData Class</td>
<td>1975</td>
</tr>
<tr>
<td>Connection (Property)</td>
<td>1977</td>
</tr>
<tr>
<td>Execute (Method)</td>
<td>1978</td>
</tr>
<tr>
<td>Fields (Property)</td>
<td>1981</td>
</tr>
<tr>
<td>Load (Method)</td>
<td>1982</td>
</tr>
<tr>
<td>LoadInstance (Method)</td>
<td>1986</td>
</tr>
<tr>
<td>Name (Property)</td>
<td>1988</td>
</tr>
<tr>
<td>Options (Method)</td>
<td>1990</td>
</tr>
<tr>
<td>R3DoNotUse (Property)</td>
<td>1992</td>
</tr>
<tr>
<td>RfcHandle (Property)</td>
<td>1993</td>
</tr>
<tr>
<td>SAPXmit (Property)</td>
<td>1994</td>
</tr>
<tr>
<td>Save (Method)</td>
<td>1995</td>
</tr>
<tr>
<td>SaveInstance (Method)</td>
<td>1999</td>
</tr>
<tr>
<td>Start (Method)</td>
<td>2000</td>
</tr>
<tr>
<td>Value (Property)</td>
<td>2003</td>
</tr>
<tr>
<td>Version (Property)</td>
<td>2005</td>
</tr>
<tr>
<td>SAPforms Data Plug-In for IDoc</td>
<td>2006</td>
</tr>
<tr>
<td>Field Class</td>
<td>2007</td>
</tr>
<tr>
<td>Load (Method)</td>
<td>2008</td>
</tr>
<tr>
<td>Name (Property)</td>
<td>2012</td>
</tr>
<tr>
<td>Save (Method)</td>
<td>2014</td>
</tr>
<tr>
<td>Fields Class</td>
<td>2018</td>
</tr>
<tr>
<td>Load (Method)</td>
<td>2019</td>
</tr>
<tr>
<td>Save (Method)</td>
<td>2023</td>
</tr>
<tr>
<td>SAPData Class</td>
<td>2027</td>
</tr>
<tr>
<td>Connection (Property)</td>
<td>2028</td>
</tr>
</tbody>
</table>
Execute (Method) ...............................................................................................................2029
Load (Method) .....................................................................................................................2032
LoadInstance (Method) ...................................................................................................... 2036
Name (Property) .................................................................................................................2038
Options (Method) ................................................................................................................2040
R3DoNotUse (Property) ..................................................................................................... 2042
SAPXmit (Property) ............................................................................................................ 2043
Save (Method) .....................................................................................................................2044
SaveInstance (Method) ...................................................................................................... 2048
Start (Method) .....................................................................................................................2049
Value (Property)...................................................................................................................2052
Version (Property) ..............................................................................................................2054
Segment Class ....................................................................................................................2055
Fields (Property).................................................................................................................2056
Load (Method) .....................................................................................................................2057
Name (Property) .................................................................................................................2061
Save (Method) .....................................................................................................................2063
Segments Class ..................................................................................................................2066
AddWithTemplate (Method)............................................................................................... 2068
Load (Method) .....................................................................................................................2069
Save (Method) .....................................................................................................................2073
SAPforms Data Plug-In for Demo ..................................................................................... 2077
SAPData Class....................................................................................................................2078
Execute (Method) ...............................................................................................................2079
Load (Method) .....................................................................................................................2082
LoadInstance (Method) ...................................................................................................... 2086
Start (Method) .....................................................................................................................2143
SAPforms Transmit Plug-In for MAPI..................................................................................2146
SAPXmit Class .....................................................................................................................2147
Execute (Method) ...............................................................................................................2148
Load (Method) ....................................................................................................................2151
MailItem (Property) ..............................................................................................................2155
Options (Method) ................................................................................................................2156
Profile Name (Property) ......................................................................................................2158
Save (Method) .....................................................................................................................2159
Start (Method) .....................................................................................................................2163
Subject (Property) ...............................................................................................................2166
SAPforms Transmit Plug-In for SAPMAPI ...........................................................................2167
SAPXmit Class .....................................................................................................................2168
Execute (Method) ...............................................................................................................2169
Load (Method) ....................................................................................................................2172
MailItem (Property) ..............................................................................................................2176
Options (Method) ................................................................................................................2177
Profile Name (Property) ......................................................................................................2179
Save (Method) .....................................................................................................................2180
Start (Method) .....................................................................................................................2184
Subject (Property) ...............................................................................................................2187
SAPforms Transmit Plug-In for SMTP via WinSock .........................................................2188
SAPXmit Class .....................................................................................................................2189
Execute (Method) ...............................................................................................................2190
Load (Method) ....................................................................................................................2193
MailServer (Property) ........................................................................................................2197
Designing an Electronic Form to Call a BAPI..................................................................2261
Maintaining Plug-Ins and Opening a Structure: BAPIs ..................................................2262
Defining and Generating a Binding: BAPIs ..................................................................2263
Example: Creating a Customer Master Record ......................................................2264
Determining a Reference Customer........................................................................2266
Using IDocs....................................................................................................................2268
Designing an Electronic Form for Use as an IDoc ..................................................2271
Generating an IDoc ........................................................................................................2273
Maintaining Plug-Ins and Opening a Structure: IDocs ............................................2275
Defining and Generating a Binding: IDocs ..............................................................2278
Example: Creating a Customer Master Record via an IDoc ...................................2280
Determining a Reference Customer........................................................................2282
Web Form Integration...........................................................................................................2284
Starting Workflows with Web Forms..............................................................................2285
Designing a Web Form to Start a Task ......................................................................2286
Defining a Task to be Started with a Web Form .....................................................2287
Generating a Web Transaction ...............................................................................2289
   Background Info on the Generated Web Transactions ....................................2291
Developing an ITS Service from a Transaction..........................................................2292
   Background Info on the Generated HTML Templates ......................................2294
Starting a Task with a Web Form............................................................................2296
Processing Work Items as a Web Form .....................................................................2297
   Web form tasks .......................................................................................................2298
Designing a Web Form to Process a Work Item.....................................................2299
Defining a Task to be Executed as a Web Form ....................................................2301
Generating a Web Transaction ...............................................................................2303
   Background Info on the Generated Web Transactions ....................................2305
Developing an ITS Service from a Transaction..........................................................2306
   Background Info on the Generated HTML Templates ......................................2308
Extending HTML Templates....................................................................................2310
Adding tasks to a workflow definition .................................................................2313
Processing a Work Item on the Internet ..................................................................2314
Creating a Customer Master Record on the Internet (BC-BMT-WFM SAPforms) ....2315
   Process Flow (BC-BMT-WFM Creating a Customer Master Record)..................2318
SAP Business Workflow (BC-BMT-WFM)

Purpose

SAP Business Workflow can be used to define business processes that are not yet mapped in the R/3 System. These may be simple release or approval procedures, or more complex business processes such as creating a material master and the associated coordination of the departments involved. SAP Business Workflow is particularly suitable for situations in which work processes have to be run through repeatedly, or situations in which the business process requires the involvement of a large number of agents in a specific sequence.

You can also use SAP Business Workflow to respond to errors and exceptions in other, existing business processes. You can start a workflow when predefined events occur, for example an event can be triggered if particular errors are found during an automatic check.

SAP provides several workflows that map predefined business processes. These workflows do not require much implementation. For an overview of these SAP workflows, refer to Workflow Scenarios in Applications [Seite 1703].

Integration

SAP Business Workflow uses the existing transactions and functions of the R/3 System and does not change the functions. You can combine the existing functions of the R/3 System to form new business processes with SAP Business Workflow. The workflow system takes over control of the business processes. If you are already using SAP Organizational Management, you can use the organizational structure created there to have the relevant agents carry out the individual activities. It is possible to have an activity carried out by a position. This ensures that the respective occupiers of the position can carry out the individual activities during execution of the workflow. This means that personnel changes in your organization are taken into account immediately in the execution of a workflow.

Features

SAP Business Workflow provides a number of tools for defining and analyzing workflows as well as for monitoring operation.

The Workflow Builder is for displaying and making changes to workflows. You can make small extensions directly to the original workflows supplied by SAP, such as carrying out your own agent assignments or changing deadline monitoring.

There are several Workflow Wizards to support you in the definition of workflows, with which you can create specific parts of a workflow. The Workflow Wizard Explorer gives you an overview of the existing Workflow Wizards.

In order to make the functions of the R/3 Systems available to a workflow, you use business objects, which you can define and analyze in the Business Object Builder. These business objects are made available to the workflow in reusable tasks. The Business Wizard Explorer gives you an overview of all existing tasks.

The end user receives information about the activities they are to carry out in their Business Workplace. This provides them with a central overview of all the activities that they are authorized to carry out. They can commence the activities from here.

Several tools are available to the workflow system administrator, with which they can control and analyze the current workflows. The workflow system administrator is notified of problems automatically by the system.
Using this Documentation

The SAP Business Workflow documentation has two parts:

- **Role documentation [Seite 93]**
  
  You use this part of the documentation if you:
  
  - Have one of the [workflow roles [Seite 94]](Seite 94) defined in the role-based procedure model
  - Have little or no experience using SAP Business Workflow
  - Want to get an overview of the main functions of SAP Business Workflow

- **Reference documentation [Seite 971]**
  
  You use this part of the documentation if you:
  
  - Require a comprehensive overview of all functions of SAP Business Workflow
  - Are interested in detailed information
  - Want to look up a particular subject
Technical Principles of SAP Business Workflow

The definition and execution of a workflow can be divided into four main areas. A user executes workflows in their Business Workplace where the work items that they can execute are displayed. A workflow must be defined to be executed. To this end, a workflow definition is created in the Workflow Builder. This definition contains steps that are executed at runtime. The step either control the workflow directly or they contain a reference to a task. The task refers to a method of an object type in the Business Object Repository (BOR) and can be executed at runtime either automatically (background task) or by a user (dialog task).

Business Workplace and work items

Work items ([Seite 1371]) are displayed to the user for execution in their Business Workplace. Work items are instances of a workflow at runtime. Their are various types of work item. Only certain types are displayed in the Business Workplace.

Workflow and workflow definition

A workflow must be defined before it can be executed. This workflow definition is made up of steps that control the workflow or refer to the tasks to be executed. You can make additional specifications about agents and deadline monitoring for a step. These specifications are evaluated at runtime by the work item manager ([Extern]). The workflow is started either manually or by the system at runtime. For the system to start a workflow, the workflow definition must contain a triggering event (for example the event "material created"). When the event occurs, the relevant workflow is started automatically.

When you activate a workflow definition, you automatically generate a runtime version. When the workflow is started (manually or automatically), the relevant runtime version is used for the execution. If the workflow definition is changed later and a new runtime version is generated, these changes do not affect workflows that are already being executed.
Tasks
Tasks describe elementary business activities. Tasks always refer to a method of an object type. Possible agents are defined for tasks. Tasks can refer to automatically executable methods (background tasks) or they can need a user to execute them (dialog tasks).

Object types and objects
An object type describes the data with which you want to work in a workflow, for example the object type Material. An object is an individual data record of an object type. Attributes are defined for an object type, which make up its data record (for example, material name or material number). Each object type has methods, in which activities are defined, which can be executed with the data (for example, create material). The transactions and functions of the R/3 System can be called in a method as can your own transactions or other applications. The last important component of an object type are its events. These describe the status changes that an object can undergo (for example, material deleted or material changed). A workflow can be started by an event of this kind being triggered.

The Business Object Repository provides an overview of all object types in the R/3 System. You can use or extend the existing object types as well as create new object types.
SAP Business Workflow Demo Examples (BC-BMT-WFM)

These workflow templates are for demonstration purposes and are particularly suitable for training. They are not intended for use in production operation.

Compare them with the workflow templates that can be used in production operation:

SAP Business Workflow - Application Scenarios [Seite 1703]

The following demo examples are available:

- Processing a Notification of Absence [Seite 52]
- Processing a Notification of Absence as a Form [Seite 72]
Demo Example: Processing a Notification of Absence (BC-BMT-WFM)

Purpose
This workflow template demonstrates how to process a notification of absence. It can be used as an example for demonstrating SAP Business Workflow functions, and is particularly suitable for training courses.

Process Flow
An employee enters a notification of absence (leave request) in the R/3 System by filling out the relevant input template. The direct superior of the employee is responsible for approving or rejecting the notification of absence. The R/3 System determines the direct superior automatically on the basis of the organizational plan maintained.

If the request is approved the creator is notified by mail:
If the request is not approved, the creator is informed and can decide whether to withdraw the notification of absence or revise it. If the superior has given reasons for the rejection in an attachment, the creator can take these into consideration.

Processing a Notification of Absence

If the creator revises the request, it is submitted to the superior for approval again. The applicant can also add an attachment, which can then be accessed by the superior. This cycle is repeated until either the superior approves the leave request or the creator withdraws it. The applicant can find out the current processing status at any time by looking in their workflow outbox.
Technical Implementation (BC-BMT-WFM)

Object types
Object technology is used to implement the interface between the R/3 functions and the workflow system. The following information is of a technical nature. You require this information if you are interested in the details of implementation or want to make your own enhancements.

Object Type FORMABSENCE (Form: Notification of Absence) [Seite 76]
Object Type SELFITEM (Work Item for Dialog Steps On Itself) [Seite 75]

Standard tasks
The standard tasks used here are available as “modules” and can also be used in other workflow scenarios without any modifications.

Standard Task AF_approve (Approve Notification of Absence) [Seite 56]
Standard Task AF_update (Revise Notification of Absence) [Seite 57]
Standard Task AF_delete (Delete Notification of Absence) [Seite 58]
Standard Task AF_Txt_Send (Send Long Text After Approval) [Seite 59]

Roles
Roles are usually defined very specifically for an application scenario. An exception to this is the role for determining an employee's superior. This role is also used in the example for processing a notification of absence and is available for general use.

Role Manager (Superior of...) [Seite 82]

Workflow template
The business process flow is implemented as a workflow definition within a workflow template. You can find this workflow template in your R/3 System.

Workflow Template AF_process (Process Notification of Absence) [Seite 62]
Workflow Definition Details [Seite 63]
Object Type FORMABSENC (Form: Notification of Absence)

Definition
A notification of absence is a business application object of the type FORMABSENC (form: notification of absence). An object of this type is identified by its number as key field [Extern]. This is defined in the Business Object Repository.

Use
In the scenario, a notification of absence is created, processed, released and possibly deleted, as an object of this type.

Structure
You can also find the attributes [Extern], methods [Extern] and events [Extern] in the object type definition in the Business Object Repository. In particular, note:

- The object method Approve, which is defined as a method with a result [Extern].
- The attributes Creator and Approver, which are defined as database field attributes [Extern] with a data type reference to the object type USR01.
Object Type SELFITEM (Work Item for Dialog Steps on Itself)

Definition
Technical object type that provides methods to operate "on itself". Objects of this type are work items that represent single-step tasks that were defined with reference to the object type SELFITEM and one of its methods.

Use
This object type is used in conjunction with the sending of a mail.

Structure
The object type SELFITEM provides the send method SendTaskDescription. This method is used to send the long text of the task that was defined with reference to the object type SELFITEM and its method SendTaskDescription as a mail.
Standard Task AF_approve (Approve Notification of Absence)

Definition
Standard task for approving or rejecting notifications of absence.

- **Standard task:** 30000016
- **Abbreviation:** AF_approve
- **Description:** Approve notification of absence

Structure

**Referenced object method, properties**

- **Object type:** FORMABSENCE (notification of absence)
- **Method:** Approve
- **Properties:** Confirm end of processing

**Agent assignment**

At runtime, this standard task is addressed to the superior of the creator of the notification of absence using role resolution.

When customizing this task, you must therefore keep the agent assignment so general that all superiors of all employees are possible agents [Extern] of the single-step task. Therefore, link the standard task Approve notification of absence with a job “head of department” (or similar), and also maintain the organizational plan in such a manner that the relevant chief positions of every single organizational unit are described with this job.
Standard Task AF_update (Revise Notification of Absence)

Definition
Standard task for revising a notification of absence.

**Standard task:** 30000017
**Abbreviation:** AF_update
**Description:** Revise notification of absence

Structure

Referenced object method, properties

**Object type:** FORMABSENC (*notification of absence*)
**Method:** Update

Agent assignment

At runtime, this standard task is executed by the creator of the notification of absence. Since this could be any employee, the single-step task must be classified as a general task [Extern] in Customizing.
Standard Task AF_delete (Delete Notification of Absence)

Definition
Standard task for deleting a notification of absence.
Standard task: 30000018
Abbreviation: AF_delete
Description: Delete notification of absence

Structure
Referenced object method, properties
Object type: FORMABSEN (notification of absence)
Method: Delete
Properties: Background processing
Terminating event: deleted
The underlying object method is an asynchronous method [Extern]. A terminating event is therefore declared for this standard task.

Agent assignment
Agent assignment does not apply because the standard task runs in the background
Standard Task AF_Txt_Send (Send Long Text After Approval)

Definition
Standard task for sending the long text.
This standard task always runs in the background. A multiline element is provided in the task container for the recipient name(s) and needs to be filled via a binding and passed on to the method.

Standard task: 30000102
Abbreviation: AF_Txt_Send
Description: Send long text after approval

Structure
Referenced object method, properties
Object type: SELFITEM (WI for method on itself)
Method: SendTaskDescription (send description)
Properties: Background processing

Agent assignment
Agent assignment does not apply because the standard task runs in the background

Other information about the standard task
The object method operates on the work item from which it is called. The object processed is therefore the work item itself and does not exist until execution time. The element _WI_Object_Id of the task container must not be assigned a value.
The referenced method has four parameters:

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Receivers</td>
<td>Recipient address as object reference, multiline</td>
</tr>
<tr>
<td>Express</td>
<td>Express flag</td>
</tr>
<tr>
<td>AddressStrings</td>
<td>Recipient name, multiline</td>
</tr>
<tr>
<td>TypeId</td>
<td>Recipient type</td>
</tr>
</tbody>
</table>

For each method parameter, there is a corresponding element in the task container and a binding definition [Seite 1218] from the task container to the method container. How the task container is "supplied" with the information required is described in the binding definition from the workflow container to the task container.

Each single-step task can only be used for the one text defined for it. If you want to send the long text in another context, you need to create a new single-step task with its own long text.
The descriptive long text contains expressions in the form &AbsenceForm.xxx& as variables. The object reference to the processed notification of absence is located in the element AbsenceForm in the task container. This element has been added to the standard elements in the task container. The expression above refers to the attribute xxx of this object. At runtime, the variables are replaced by the current attribute values.
Role Manager (Superior of...)

Definition
Role with which the superior of an agent, a position, or an organizational unit can be determined.

Role: 000000168
Abbreviation: Manager
Description: Superior of...

Structure

Role parameters
The following can be passed as role parameters:

- **ORG_OBJECT** (reference field RHOBJECTS-OBJECT)
  or
- **OType** (reference field OBJEC-OTYPE) and **ObjId** (reference field OBJEC-REALO)

All three role parameters are elements in the role container. The contents of **ORG_OBJECT** are evaluated with priority. If **ORG_OBJECT** is not passed, the role parameters **OType** and **ObjId** are evaluated.

- **ORG_OBJECT** contains either the specification of a person, a position or an organizational unit in the composite form **P <8-digit number>, S <8-digit number>, O <8-digit number>** or the specification of a user in the composite form **US<12-character user name>**.

- **OType** (reference field OBJEC-OTYPE) contains the ID of the object in a 2-character field. The following are allowed:
  - **O** organizational unit
  - **S** position
  - **US** user name
  - **P** person

- **ObjId** contains either the number of the person, position or organizational unit, or the user name in a 12-character field.

Binding definition
In the demo example, the role is used to find the superior of the initiator of the workflow starting from the initiator. The username of the initiator is stored in a 14-character field with reference to **RHOBJECTS-OBJECT** in the element **WF_Initiator** of the workflow container. The binding is therefore defined to the element **ORG_OBJECT** of the role container for the relevant step.

Integration

Preparation in the organizational plan
The company's organizational plan is evaluated during role resolution. This takes into account either direct reporting structures between positions (position "reports to" position, relationship A/B002) or the designation of chief positions (position "manages" organizational unit, relationship A/B012) in the organizational units.
The relevant relationships must have been created at the appropriate points when the organizational plan was set up.
Workflow Template AF_process (Process Notification of Absence)

Definition
Workflow template used to process a notification of absence once it has been created. Note that the notification of absence is not created in the workflow, but is "application functionality". A triggering event makes the link to the workflow. One of the main functions of workflow control is determining the correct agents.

Workflow template: 30000015
Abbreviation: AF_process
Description: Process notification of absence

Structure

Workflow container
The main two items of information that must be available during the workflow are:

- The (object) reference to the notification of absence to be processed
- The name of the creator of the notification of absence
  (required to find the superior in role resolution)

The relevant elements must be available in the workflow container to take this information:

- The element _WF_Initiator for the creator is in the workflow container as standard.
- The element AbsenceForm for the object reference was created in addition to the standard elements.

Both items of information are initially available as event parameters in the container of the triggering event and must be passed from there "via binding" to the workflow container.

Triggering event of workflow template
The event created for the object type FORMABSENC (notification of absence) is entered as the triggering event of the workflow template. This "linkage" between the event and the workflow template to be started is deactivated as standard and must first be activated for this workflow template in Customizing if the workflow template is to be started.

Binding from the event parameter container to the workflow container
The following binding is defined between the triggering event (or event container) and the workflow container:

<table>
<thead>
<tr>
<th>Workflow container</th>
<th>Event container</th>
</tr>
</thead>
<tbody>
<tr>
<td>WF_Initiator</td>
<td>Evt_Creator</td>
</tr>
<tr>
<td>AbsenceForm</td>
<td>Evt_Object</td>
</tr>
</tbody>
</table>
Details of Workflow Definition

The following information is of interest in connection with the workflow definition for the workflow template AP_process. Look at the definition in the system.

Enter notification of absence

Notification of absence created

Approve?

Approved

Rejected

Set flag

Set

Revise?

No

Yes

Revise

Delete

Sent

Deleted

Revised

Resubmit for approval?

No

Yes

Terminate workflow

Workflow terminated

Binding for the workflow definition steps

Element _WI_Object_Id of task container

In each of the steps of the workflow definition in which the notification of absence can be approved, revised and deleted, a binding is defined from the element AbsenceForm of the workflow container to the element _WI_Object_ID of the relevant task container.

In addition to the standard elements in the workflow container, the element AbsenceForm was created to contain the object reference. It was provided with the reference to the created object from the triggering event.

Element AbsenceForm of task container

In the step of the workflow definition in which the descriptive long text is sent, a binding is defined from the element AbsenceForm of the workflow container to the element AbsenceForm of the task container. This binding ensures correct text substitution in the long text of the single-step task. The element _WI_Object_ID of the task container must not be filled in this step.

Agent determination

Steps for the creator

If the creator of the notification of absence is to process a step, the element _WF_Initiator of the workflow container is entered as the agent. This element is in the workflow container as standard and was provided with the name of the creator from the triggering event.
Details of Workflow Definition

**Steps for the superior**
If the creator's superior is to process a step, they must be determined via role resolution. The role requires the name of the creator as “input”. The role 00000168 (Superior of...) is therefore entered as the agent, and a binding is defined from the element _WI_Initiator of the workflow container to the element Org_Object of the role container.

**Address determination and sending the text**
In the step in which the standard task AF_Txt_Send is called to send the mail to the creator of the notification of absence, the creator (element _WF_Initiator of the workflow container) is passed to the element AddressStrings in the task container. The element _WI_Object_Id of the container of this single-step task remains empty.

**Result of the user decision**
The result of the user decision in which the creator must decide whether to delete or revise the notification of absence is required again for the loop check to determine whether the notification of absence should be resubmitted.

The result of the user decision, which is returned in the element _WI_Result of the task container, is therefore transferred to the element Flag of the workflow container via a binding. This element Flag was created for this in the workflow container with a data type reference to the DDIC table field SWD_LINES-Returncode.
Preparation and Customizing (BC-BMT-WFM)

Purpose
In addition to general Customizing, which ensures that the workflow system functions properly, special Customizing is required for this workflow template.
If you want to execute this workflow scenario in an IDES system, you do not need to make the following settings.

Prerequisites
You have carried out general Customizing for SAP Business Workflow.

Process Flow
The following steps are required:

- Setting up the Organizational Plan [Seite 87]
- Performing Task-Specific Customizing [Seite 67]
- Administration for Shared Office Settings [Seite 89]
Setting up the Organizational Plan

The notification of absence is created by an employee and then approved or rejected by their superior.

Procedure

To set up a suitable organizational plan for this scenario, carry out the following steps in SAP Business Workflow Customizing:

1. Execute the activity *Edit organizational plan* in SAP Business Workflow Customizing.
2. Create an organizational unit [Extern] or change an existing organizational unit.
3. Create two positions [Extern] in this organizational unit:
4. Designate the position you have set up for the superior as a chief position.
Performing Task-Specific Customizing

Procedure
The following must be carried out in SAP Business Workflow Customizing:

- The standard task 30000016 (Approve notification of absence) must be assigned to its possible agents.
- The standard task 30000017 (Revise notification of absence) must be classified as a general task.

1. Execute the Customizing activity Perform task-specific Customizing.
2. In the application component SAP Business Workflow, you will find the option Assign tasks to agents.

   Carry out the two assignments described above:
   - Link the standard task 30000016 (Approve notification of absence) to the job from which you derived the position for the superior in the department.
   - Specify the job of head of department as the possible agent of the standard task 30000016 (Approve notification of absence).
   - Classify the standard task 30000017 as a general task.
Maintaining an Event Receiver Linkage

The event created of the object type FORMABSENCE is a triggering event of the workflow template 30000015 and, as such, is entered in the type linkage table as standard.

Procedure

In order for the workflow template to actually be started, the linkage between the triggering event and the workflow template as the receiver of the event must be activated in SAP Business Workflow Customizing.

1. Execute the Customizing activity Perform task-specific Customizing.

2. In the application component SAP Business Workflow, activate the event linkage for the workflow template Process notification of absence.

(Alternatively, you can activate the event receiver linkage by processing the workflow template AF_process directly.)
**Administration Required for Shared Office Settings**

After approval, a mail is sent to a recipient specified by the container element \_WF\_Initiator. This container element contains the agent in the HR-specific form with the prefix US.

**Procedure**

So that a mail can be sent correctly to the various recipient types, the relevant recipient types must be declared in *shared office settings*:

2. Choose the tab page *Addressing*.
3. Select *Organizational units*. 
Operation and Connection to Application Functionality (BC-BMT-WFM)

Purpose
You use this process if you want to test or demonstrate workflow functions.

Demo example for notification of absence in IDES
In IDES, the scenario described can switch between the users WF-MM-3 (creator of notification of absence) and WF-MM-4 (superior).
Both users hold positions that are assigned to the organizational unit US Purchasing. The position of the user WF-MM-4 is the chief position of this organizational unit. It is described by the job head of department, the activity profile of which includes releasing notifications of absence.

IDES may contain other workflow templates and workflow tasks (from the workflow tutorials), which are used to process the same notification of absence.

When you “run through” the scenario, you must be sure only to execute the work items for the workflow template WS30000015.

Prerequisites
You have made the settings described in the section Preparation and Customizing (BC-BMT-WFM) [Seite 65].

Process Flow
A user enters a notification of absence. This is carried out from the transaction accessed by following the path Tools → Business Workflow → Development → Environment → Demo: Fill out form.

Entering a notification of absence and creating an event
The action of entering (and saving) a notification of absence creates a new object of the object type notification of absence and automatically generates the event created for this object type. The event container of this event contains:

- The reference to the notification of absence in the element _Evt_Object
- The user name of the creator of the notification of absence in the form US<UserName> in the element _Evt_Creator

This event is the triggering event of the workflow template for processing a notification of absence.

Approving a notification of absence
The user you entered as the holder of the position designated as chief position finds a work item in their workflow inbox, which represents the standard task Approve notification of absence. They approve or reject the notification of absence by executing this work item. Attachments can be appended to the work item explaining any reasons.

Important: The processing of this work item must completed in the end using the function Set to 'Done'. (This procedure is defined as a property of the underlying standard task.)

The decision made by the superior determines how the workflow progresses. If the request is rejected, the creator of the notification of absence must make the next move.
Additional information on this step

- Deadline monitoring
  
  This step is subject to deadline monitoring. If the superior has not started processing the request within 5 minutes, the creator of the notification of absence is notified as the notification agent for deadline.

  This procedure is defined in the description of the step within the workflow definition.

- Selected and possible agents
  
  According to the settings you made in Customizing, all employees (= possible agents) whose activity profile is described by the job head of department (or another job you chose) can in principle approve the notification of absence. At runtime, however, this decision is only to be made by the head of department (= selected agent) "playing" the role of this employee's superior. The relevant person is determined from the organizational plan of the company via role resolution.

Employee's decision on how to continue

If the superior rejects the notification of absence, the creator of the notification of absence is asked to decide whether they want to revise or withdraw it.

Additional information on this step

- Withdraw notification of absence
  
  If the creator decides not to revise the notification of absence, the rejected notification of absence is deleted in a background step.

  This terminates the workflow.

- Revise notification of absence
  
  If the creator decides to revise the notification of absence, they are immediately asked to change it. Advancing with immediate dialog makes it possible for this functionality to be offered without the employee having to choose a work item from their workflow inbox.

What next?

If the creator has decided to revise their notification of absence, it is resubmitted to the superior for approval.
Demo Example: Processing a Notification of Absence as a Form (BC-BMT-WFM Form)

Definition
The workflow template for processing a notification of absence as a form described here can be used as an example to demonstrate SAP Business Workflow functions. It is particularly useful for training courses. This workflow template cannot be used in productive operation.

This scenario is simpler than the Demo Example: Processing a Notification of Absence (BC-BMT-WFM) [Seite 52] which is similar in content. This scenario is more about demonstrating the functionality of form processing than describing a practical business process.

To be able to perform this scenario you must install SAPforms [Seite 1718] on your PC.

Process Flow

Entering a Notification of Absence as a Form
An employee enters a notification of absence by making the appropriate entries in a form installed locally on their PC. In this demo example, the data entered can be transmitted to the R/3 System in two ways (form settings):

- The employee transmits the data entered to the R/3 System via an Internet-enabled mail system as an attachment to an Internet mail.
- The employee transmits the data via an RFC call to the R/3 System.

The user is not however concerned with technical details of the data transfer. The R/3 System uses the data from the form to start a workflow template that controls and co-ordinates further processing of the notification of absence.

Approving a Notification of Absence as a Form
The direct superior of the employee approves or rejects the notification of absence. The R/3 System determines the superior automatically using the maintained organizational plan. The superior is notified by mail. The superior fills in a form, approving or rejecting the notification. The employee is then informed by mail of the approval or rejection.

Advantages of Using a Form
Filling in a form is generally an intuitive process. It is particularly worth using forms if simple functions are to be made available to a large number of employees who do not otherwise work much in the R/3 System, if at all.

Furthermore, integration of a mail system enables the workflow to be started asynchronously. If the R/3 System is not available or the user is working offline, the mail system ensures that the workflow is started later. The mail system must be suitable for offline use.
Technical Implementation (BC-BMT-WFM)

Object types
Object technology is used to implement the interface between the R/3 functions and the workflow system. The following information is of a technical nature. You require this information if you are interested in the details of implementation or want to make your own enhancements.

Object Type FORM (Desktop Form) [Seite 74]
Object Type FORMABSENCE (Form: Notification of Absence) [Seite 76]
Object Type SELFITEM (Work Item for Dialog Steps On Itself) [Seite 75]

Standard tasks
The standard tasks provided by SAP as single steps describe the basic business activities from an organizational point of view. The standard tasks used here are available as “modules” and can also be used in other workflow scenarios without any modifications.

Standard Task WF_AbsFormCr (Create Notification of Absence without Dialog) [Seite 77]
Standard Task WF_AbsCheck (Check Vacation with Form) [Seite 78]
Standard Task AF_Txt_Send2 (Send Long Text after Approval) [Seite 80]
Standard Task WF_AbsNRep2 (Notify Rejection of Notification of Absence) [Seite 81]

Roles
Roles are usually defined very specifically for an application scenario. An exception to this is the role for determining an employee's superior. This role is also used in the example for processing a notification of absence and is available for general use.

Role Manager (Superior of...) [Seite 82]

Workflow template
The business process flow is implemented as a workflow definition within a workflow template. You can find this workflow template in your R/3 System.

Workflow Template WF_SAPFORMS1 (Process Notification of Absence) [Seite 84]
Workflow Definition Details [Seite 85]
Object Type FORM (Desktop Form)

Definition
The object type FORM serves as a “hanger” for the methods Process and HTMLProcess. It is a technical object type without key fields. There are therefore no objects of this type in the R/3 System.

Use
The object type FORM with its methods Process and HTMLProcess is used in defining form tasks and/or HTML form tasks.

Method: Process
The method Process enables you to process the container of the work item, via which the method is executed, using an electronic form. This method has the following import parameters:

- R3FForm
  Form name of a PC form.
- R3FFormType
  Form type of a PC form.
- Workitem
  Object reference to the work item, in which the method is executed. (The method can access the container of the referenced work item via this object reference.)

The functionality is technically implemented in such a way, that the values are exported from the work item container to the PC in a structured manner (as “R3F attachment”), and the form specified by form name and form type is started there.

Method: HTMLprocess
The method HTMLprocess calls the Web transaction defined for the HTML form task.
Object Type SELFITEM (Work Item for Dialog Steps on Itself)

Definition
Technical object type that provides methods to operate "on itself". Objects of this type are work items that represent single-step tasks that were defined with reference to the object type SELFITEM and one of its methods.

Use
This object type is used in conjunction with the sending of a mail.

Structure
The object type SELFITEM provides the send method SendTaskDescription. This method is used to send the long text of the task that was defined with reference to the object type SELFITEM and its method SendTaskDescription as a mail.
Object Type FORMABSENC (Form: Notification of Absence)

Definition
A notification of absence is a business application object of the type FORMABSENC (form: notification of absence). An object of this type is identified by its number as key field [Extern]. This is defined in the Business Object Repository.

Use
In the scenario, a notification of absence is created, processed, released and possibly deleted, as an object of this type.

Structure
You can also find the attributes [Extern], methods [Extern] and events [Extern] in the object type definition in the Business Object Repository. In particular, note:

- The object method Approve, which is defined as a method with a result [Extern].
- The attributes Creator and Approver, which are defined as database field attributes [Extern] with a data type reference to the object type USR01.
Standard Task WF_AbsFormCr (Create Notification of Absence without Dialog)

Definition
Standard task for creating a notification of absence with the data from the task container. As the data must be available in full, no more dialog is carried out.

Standard task: 30000365
Abbreviation: WF_AbsFormCr
Name: Create notification of absence without dialog

Structure

Referenced object method, properties
Object type: FORMABSENC (notification of absence)
Method: CreateWithoutDialog (create form without dialog)
Properties: Background processing

Task container, binding definition
The task container contains, amongst other things, all method parameters of the method CreateWithoutDialog. These method parameters are required for creating a notification of absence without dialog.
The binding between the corresponding elements from the task container and the method container is defined.
The method creates a notification of absence with the data from the task container/method container. After execution of the method, the reference to the notification of absence created is in the element _WI_Object_Id of the task container.

Agent assignment
Agent assignment does not apply because the standard task runs in the background.
Standard Task WF_AbsCheck (Check Vacation (with Form))

Definition
Standard task used to transfer all the data of a notification of absence located in the task container to a form for execution.

Standard task: 30000366
Abbreviation: WF_AbsCheck
Name: Check vacation (with form)

Structure

Referenced object method, properties
Object type: FORM (desktop form)
Method: Process (process)

Task container, binding definition
As the method Process is a generic method which can operate on the data of any form, the method container cannot be a 1:1 mapping of the task container.

The method Process has three method parameters which each have correspondence in the task container.

The task container contains additional elements which describe the fields of a notification of absence. The last-named container elements do not have correspondence in the method container.

The binding from the task container to the method container is defined as follows:

- R3FForm <= &R3FForm&
- R3FFormType <= &R3FFormType&
- Workitem <= &Workitem&

The method container contains the element Workitem which is filled with the reference to the work item in which the method is executed. The method can access the container of the referenced work item via this object reference. (In contrast, the method container of the method described above CreateWithoutDialog contains all elements of the task container.)

Initial Value Assignment
As the single-step task was defined specifically for a particular form, form name and type are assigned to the relevant elements of the task container in an initial value assignment:

- R3FForm <= APPROVE;IPM.Note.SAP.Demo.Absence.Approval
- R3FFormType <= EXE;Outlook

When executing the methods, the values are exported from the work item container to the PC in a structured manner (as “R3F attachment”), and the form specified by form name and form type is started there.

File APPROVE.EXE is installed on your PC together with SAPforms [Seite 1718].
Agent assignment

At runtime, this standard task is addressed to the superior of the creator of the notification of absence using role resolution.

When customizing this task, you must therefore keep the agent assignment so general that all superiors of all employees are possible agents [Extern] of the single-step task. Therefore, link the standard task Approve notification of absence with a job “head of department” (or similar), and also maintain the organizational plan in a such a manner that the relevant chief positions of every single organizational unit are described with this job.
Standard Task AF_Txt_Send2 (Send Long Text after Approval)

Definition
Standard task for sending the long text.
This standard task always runs in the background. A multiline element is provided in the task container for the recipient name(s) and needs to be filled via a binding and passed to the method.

Standard task: 30000382
Abbreviation: WF_Txt_Send2
Name: Send long text after approval

Use
This standard task is used as a step in the workflow when a mail is sent in the case of approval.

Structure

Referenced object method, properties
Object type: SELFITEM (WI for method on itself)
Method: SendTaskDescription (send description)
Properties: Background processing

Agent assignment
Agent assignment does not apply because the standard task runs in the background

Other information about the standard task
The object method operates on the work item from which it is called. The object processed is therefore the work item itself and does not exist until execution time. The element _WI_Object_Id of the task container must not be assigned a value.
The referenced method has four parameters:

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Receivers</td>
<td>Recipient address as object reference, multiline</td>
</tr>
<tr>
<td>Express</td>
<td>Express flag</td>
</tr>
<tr>
<td>AddressStrings</td>
<td>Recipient name, multiline</td>
</tr>
<tr>
<td>TypeId</td>
<td>Recipient type</td>
</tr>
</tbody>
</table>

For each method parameter, there is a corresponding element in the task container and a binding definition [Seite 1218] from the task container to the method container. How the task container is “supplied” with the information required is described in the binding definition from the workflow container to the task container.

Each single-step task can only be used for the one text defined for it. If you want to send the long text in another context, you need to create a new single-step task with its own long text.
The descriptive long text contains expressions in the form &AbsenceForm.xxx& as variables. The object reference to the processed notification of absence is located in the element AbsenceForm in the task container. This element has been added to the standard elements in the task container. The expression above refers to the attribute xxx of this object. At runtime, the variables are replaced by the current attribute values.
Standard Task WF_AbsNRep2 (Notify Rejection of Notification of Absence)

Definition
Standard task for sending the long text.
This standard task always runs in the background. A multiline element is provided in the task container for
the recipient name(s) and needs to be filled via a binding and passed to the method.

Standard task: 30000381
Abbreviation: WF_AbsNRep2
Name: Notify rejection of notification of absence

Use
This standard task is used as a step in the workflow when a mail is sent in the case of rejection.

Structure

Referenced object method, properties
Object type: SELFITEM (WI for method on itself)
Method: SendTaskDescription (send description)
Properties: Background processing

Agent assignment
Agent assignment does not apply because the standard task runs in the background

Other information about the standard task
The object method operates on the work item from which it is called. The object processed is therefore the
work item itself and does not exist until execution time. The element _WI_Object_Id of the task
container must not be assigned a value.

The referenced method has four parameters:

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Receivers</td>
<td>Recipient address as object reference, multiline</td>
</tr>
<tr>
<td>Express</td>
<td>Express flag</td>
</tr>
<tr>
<td>AddressStrings</td>
<td>Recipient name, multiline</td>
</tr>
<tr>
<td>TypeId</td>
<td>Recipient type</td>
</tr>
</tbody>
</table>

For each method parameter, there is a corresponding element in the task container and a binding definition
[Seite 1218] from the task container to the method container. How the task container is “supplied” with
the information required is described in the binding definition from the workflow container to the task
container.

Each single-step task can only be used for the one text defined for it. If you want to send the long text in
another context, you need to create a new single-step task with its own long text.

The descriptive long text contains expressions in the form &AbsenceForm.xxx& as variables. The
object reference to the processed notification of absence is located in the element AbsenceForm in the
task container. This element has been added to the standard elements in the task container. The expression
above refers to the attribute xxx of this object. At runtime, the variables are replaced by the current
attribute values.
Role Manager (Superior of...)

Definition
Role with which the superior of an agent, a position, or an organizational unit can be determined.

Role: 00000168
Abbreviation: Manager
Description: Superior of...

Structure

Role parameters
The following can be passed as role parameters:

- ORG_OBJECT (reference field RHOBJECTS-OBJECT)
  or

- OType (reference field OBJEC-OTYPE) and ObjId (reference field OBJEC-REALO)

All three role parameters are elements in the role container. The contents of ORG_OBJECT are evaluated with priority. If ORG_OBJECT is not passed, the role parameters OType and ObjId are evaluated.

- ORG_OBJECT contains either the specification of a person, a position or an organizational unit in the composite form P <8-digit number>, S <8-digit number> or O <8-digit number>, or the specification of a user in the composite form US<12-character user name>.

- OType (reference field OBJEC-OTYPE) contains the ID of the object in a 2-character field. The following are allowed:
  O   organizational unit
  S   position
  US  user name
  P   person

- ObjId contains either the number of the person, position or organizational unit, or the user name in a 12-character field.

Binding definition
In the demo example, the role is used to find the superior of the initiator of the workflow starting from the initiator. The user name of the initiator is stored in a 14-character field with reference to RHOBJECTS-OBJECT in the element _WF_Initiator of the workflow container. The binding is therefore defined to the element ORG_OBJECT of the role container for the relevant step.

Integration

Preparation in the organizational plan
The company's organizational plan is evaluated during role resolution. This takes into account either direct reporting structures between positions (position "reports to" position, relationship A/B002) or the designation of chief positions (position "manages" organizational unit, relationship A/B012) in the organizational units.
The relevant relationships must have been created at the appropriate points when the organizational plan was set up.
Workflow Template WF_SAPFORMS1 (Process Notification of Absence)

Use
This workflow template is used to process a notification of absence including its creation. Actual data entry for the notification of absence occurs in an external form. The data entered there is then put into the workflow container when the workflow template is started. A notification of absence is created with this data in the first step of the workflow.

Workflow template: 30000161
Abbreviation: WF_SAPFORMS1
Description: Process notification of absence

Structure

Workflow container
The following information must be in the workflow container:
- The fields of a notification of absence:
  These fields are filled with the data from the form as mandatory import elements when the workflow template is started.
- The (object) reference to the notification of absence to be processed:
  This object reference comes into existence after the notification of absence was created in the R/3 System in the first step of the workflow.
- The name of the person who created the notification of absence:
  This applicant is required in order to find the superior in role resolution.

The relevant elements for this information are in the workflow container:
- The elements PersonnelNumber, Department, ... for the notification of absence fields were created in addition to the standard elements.
- The element AbsenceForm for the object reference was created in addition to the standard elements.
- The element _WF_Initiator for the creator is in the workflow container as standard.

Start of workflow
The workflow is started after execution of form APPROVE.EXE.

⚠️
For the scenario to work, SAPforms [Seite 1718] must be installed on your PC.

In the R/3 System, the data entered in the form is transferred to the workflow container of the workflow template started. The task is to be started and which form fields are mapped on which container elements were established in the definition of the form.
Details of Workflow Definition

The following details are of interest in connection with the workflow definition for the workflow template WF_SAPFORMS1. Look at the definition in the system.

Binding for the workflow definition steps

Element _WI_Object_Id of task container / Element AbsenceForm of the workflow container

This element of the task container contains the reference to the notification of absence that is to be processed.

In the first step of the workflow definition, in which the notification of absence is created in the R/3 System, the binding is defined from the _WI_Object_ID of the task container to the element AbsenceForm of the workflow container. The element AbsenceForm for the object reference was created in addition to the standard elements in the workflow container.

In the following step of the workflow definition, in which the notification of absence can be approved, a binding is defined from the element AbsenceForm of the workflow container to the element _WI_Object_Id of the task container.

Element AbsenceForm of task container /

Element AbsenceForm of the workflow container

In the steps of the workflow definition in which the descriptive long text is sent, a binding is defined from the element AbsenceForm of the workflow container to the element AbsenceForm of the task container. This binding ensures correct text substitution in the long text of the single-step task.

The element _WI_Object_Id of the task container must not be filled in this step.

Agent determination

Steps for the creator

If the creator of the notification of absence is to process a step, the element _WF_Initiator of the workflow container is entered as the agent. This element is in the workflow container as standard and was supplied from the form.

Steps for the superior

If the creator's superior is to process a step, they must be determined via role resolution. The role requires the name of the creator as “input”. The role 00000168 (Superior of...) is therefore entered as the agent, and a binding is defined from the element _WI_Initiator of the workflow container to the element Org_Object of the role container.

Address determination and sending the text

In the step in which the standard task AF_Txt_Send is called to send the mail to the creator of the notification of absence, the creator (element _WF_Initiator of the workflow container) is passed to the element AddressStrings in the task container. The element _WI_Object_Id of the container of this single-step task remains empty.
Preparation and Customizing (BC-BMT-WFM)

Purpose
In addition to general Customizing, which ensures that the workflow system functions properly, special Customizing is required for this workflow template. If you want to execute this workflow scenario in an IDES system, you do not need to make the following settings.

Prerequisites
You have carried out general Customizing for SAP Business Workflow.

Process Flow
The following steps are required:
- Setting up the Organizational Plan [Seite 87]
- Performing Task-Specific Customizing [Seite 88]
- Administration for Shared Office Settings [Seite 89]
- SAPforms Administration [Seite 1727]
Setting up the Organizational Plan

The notification of absence is created by an employee and then approved or rejected by their superior.

Procedure

To set up a suitable organizational plan for this scenario, carry out the following steps in SAP Business Workflow Customizing:

5. Execute the activity *Edit organizational plan* in SAP Business Workflow Customizing.
6. Create an organizational unit [Extern] or change an existing organizational unit.
7. Create two positions [Extern] in this organizational unit:
8. Designate the position you have set up for the superior as a chief position.
Performing Task-Specific Customizing

Procedure
In SAP Business Workflow Customizing, the standard task 30000366 (Check vacation (with form)) must be assigned to its possible agents.
1. Execute the Customizing activity Perform task-specific Customizing.
2. In the application component SAP Business Workflow, you will find the option Assign tasks to agents.

   Link the standard task 30000366 (Check vacation (with forms)) to the job from which you have derived the position for the departmental superior.

   ![Alert Icon]

   Enter the job of head of department as the possible agent of the standard task 30000366 (Check vacation (with forms)).
Administration Required for Shared Office Settings

After approval, a mail is sent to a recipient specified by the container element _WF_Initiator. This container element contains the agent in the HR-specific form with the prefix US.

Procedure

So that a mail can be sent correctly to the various recipient types, the relevant recipient types must be declared in shared office settings:

5. Choose the tab page Addressing.
6. Select Organizational units.
Operation and Connection to Application Functionality (BC-BMT-WFM)

Purpose
You use this process if you want to test or demonstrate workflow functions.

Process Flow

Entering a notification of absence using a form
A user enters a notification of absence. This is carried out using a form which must be available locally on the user's PC together with the relevant form application.
The following are established in form definition:
- Which task is started in the R/3 System.
- How the communication to the R/3 System is structured.
- Which form fields are mapped on which container elements of the task to be started.
The form is not defined by the administrator but by a central group and made available across the enterprise.
The form used for this example (APPROVE.EXE) is part of the SAPforms Installation [Seite 1725].

Creating a notification of absence
A notification of absence (that is, a new object for the object type notification of absence) is created in the R/3 System from the values in the container elements of the workflow container.

Creating a notification of absence always generates the event created for this object type automatically.

If, for demonstration purposes, you have already established the “other” scenario for processing a notification of absence (workflow template AF_process, WS30000015), this workflow template is also started as a reaction to the event.

You should cancel the event receiver linkage for the workflow template WS30000015, in order to avoid double work items with similar functionality.

Approving a notification of absence
The user you have entered as the holder of the position designated as a chief position finds a work item in his workflow inbox which represents the standard task Check vacation (with form).
Executing this work item exports the form data to the superior's PC and starts the form application for “approving” or “not approving” the notification of absence there.

Executing work items via the Internet
If automatic forwarding is activated for the superior, and the report for periodic transmission is scheduled, an Internet mail is generated for every work item which refers to the method Process for the object type FORM. This Internet mail contains the data of the work item container as an “R3F attachment”.
This attachment can be executed in the external mail system with the form there. Returning the form to the R/3 System terminates the work item there and continues the workflow.
General Procedure Model

Purpose
With SAP Business Workflow, you can map business processes in the R/3 System and process them (several times if required) under the control of the workflow system. A workflow management system can process and monitor structured processes that:

- Contain a sequence of activities
- Reoccur in the same or similar forms
- Involve several people or groups of people
- Require a lot of coordination

In addition to this general procedure model for working with SAP Business Workflow, there are also role-specific procedure models. For descriptions of these procedure models, refer to Workflow Roles [Seite 94].

Prerequisites
Before working with SAP Business Workflow, you should have a clear idea of the areas in question and the extent to which you want to change and adapt business processes, and pass them over to electronic process control.

Process Flow
Use the following procedure model as orientation when working with SAP Business Workflow.

Organizational plan
The enterprise-specific organizational plan describes the organizational assignment of the employees. The aim of this is to determine the responsibility of employees for the execution of individual business activities in the form of activity profiles.

You maintain the organizational plan on a client-specific basis. You can use an organizational plan that was created (or is still being created) for HR purposes in SAP Business Workflow as well without making any changes, as long as the workflow functionality and the HR application are used in the same client.

Generally, however, in each client you (only) map those sub-areas and organizational structures of your enterprise, in which you also coordinate business processes using SAP Business Workflow.

For further information, refer to Enterprise-Specific Organizational Plan [Seite 1708].

Objects and object type definition
You identify all objects involved in your business process. You sort out which business function you want to map in your scenario and which attributes you want to access for control purposes.

You check whether the relevant object types with their methods, attributes and events are defined in the Business Object Repository. The grouping of object types in the application component hierarchy and the option of searching generically for parts of a name help when looking for object types.

- If you find an object type whose definition meets your requirements, you can use it without making any modifications.
- If you find an object type whose definition does not quite meet your requirements, you extend its definition.
- If you do not find a suitable object type, you define your own object type.

For further information, refer to Extending and Adapting Object Types [Seite 1157].
Single-step tasks
You identify the single-step tasks involved in your business process. Establish which object method is to be executed with the single-step task and who is responsible for executing it. You then define the single-step tasks by specifying object type and method, and determine the possible agents of the single-step tasks.

In the test and development stage, you should specify all single-step tasks used as general tasks [Extern].

You check whether single-step tasks are already defined.
The single-step tasks available are not usually sufficient, meaning that you define additional single-step tasks to meet your requirements.
For further information, refer to Single-Step Task [Seite 1175].

Standard roles
You identify possible agent roles in your business process. You use roles when agents are to be found using specific, business, functionally-oriented criteria.

Specifying roles is always optional.
You can initially do without roles in the test and development stage.
You check whether you can use any existing roles. If you cannot use any existing roles, you define your own.
For further information, refer to Role [Seite 1277].

Events
You identify the events required to initiate and control the workflow and check whether these events are defined for the relevant object types.
If you require an event that is not incorporated in the standard version, you must add the event to the relevant object type definition and ensure that the event is created.
For further information, refer to Event [Seite 1321].

Multistep tasks
To describe a business process, which normally comprises several steps, you first create a workflow task and then reference a workflow definition in it.
You can use these workflow templates as a basis for your own developments, helping you to make a start.
You can, of course, also use the definition tools to define new multistep tasks from scratch.
For further information, refer to Multistep Task [Seite 1194].

Demo example and tutorials
You can learn the basic principles of workflow definition and execution in the demo example for processing a notification of absence.
For further information, refer to Demo Example: Processing a Notification of Absence [Seite 52].
There are also self-teach tutorials that introduce you step-by-step to selected functions. See SAP Business Workflow - Tutorials [Seite 1581].
Role Documentation

Definition
Role-based view on SAP Business Workflow reference documentation.

Purpose
The role documentation is used when you want a role-based and hence task-oriented view on the workflow documentation. You have to identify one of the predefined workflow roles as representative for your task area.
The purpose of the role documentation is to make access to SAP Business Workflow as easy as possible.

Structure
The following workflow roles [Seite 94] are defined:
- Process consultant [Seite 96]
- Workflow developer [Seite 220]
- Workflow system administrator [Seite 683]
- Workflow agent [Seite 805]
- Process controller [Seite 884]

Integration
The SAP Business Workflow reference documentation consists of four structures:
- BC - SAP Business Workflow - Navigation [Seite 972]
  This structure contains a full description of the functions of the SAP Business Workflow development and administration environment.
- BC - SAP Business Workflow - Tutorials [Seite 1581]
  This structure contains tutorials for the SAP Business Workflow development and administration environment.
- BC - SAP Business Workflow - Scenarios in Applications [Seite 1703]
  This structure contains the documentation for the R/3 application component tasks and workflows supplied by SAP.
- BC - SAPforms [Seite 1718]
  This structure contains the documentation for integrating Web forms and electronic forms with the R/3 System.

There are links in the individual roles of the role documentation to various parts of these structures.

Only the reference documentation contains a description of all functions.
Roles in Workflow

Use

SAP Business Workflow can be used on the basis of roles and a number of roles are supplied for this purpose. The roles supplied cover the activity fields of the groups of people that work with SAP Business Workflow. There is role-specific documentation for each of these roles. This documentation contains only the sections that are relevant for the holders of the role in question.

The term "workflow role" should not be confused with the term "role", which is used in connection with Organizational Management. For further information, refer to the documentation under Role [Seite 1277].

Features

The workflow roles in this part of the documentation are based on the assumption of a particular way of working with SAP Business Workflow. It is assumed that IMG activities are already complete. The workflow roles used are stylized. But they can also be used in your organization if you work in a different way.

The role-based procedure model

The process consultant [Seite 96] identifies the business processes in an organization, which are to be mapped with workflow technology. The workflow developer [Seite 220] then implements these processes using the definition tools of SAP Business Workflow.

At runtime, the workflow is evaluated and managed from a technical perspective by the workflow system administrator [Seite 683]. The process controller [Seite 884] caters for this from a business perspective. The end user in the workflow context is the workflow agent [Seite 805], who processes the tasks assigned to them in their workflow inbox.
A role-specific procedure model is also described for each role, meaning that the overall role-based procedure model is derived from the overview model and the individual models. The documentation also contains a general procedure model [Seite 1714] for working with SAP Business Workflow.

**Relation to reference documentation**

The workflow roles are intended for specific users of SAP Business Workflow. Specific users mostly only require some of the information available. The role documentation is an application-specific and hence also a limited view of the reference documentation. The reference documentation is intended for those persons who require a complete and detailed overview of the functions available.

**Activities**

Read the documentation for your role. If you require further information on a topic, you can refer to it in the reference documentation.
Process Consultant

Definition
Person within the system of workflow roles who analyses the existing business processes, identifies those that are suitable for SAP Business Workflow and maps them onto a business process model. The business process model is implemented by the workflow developer. The process consultant is also responsible for the maintenance of the organizational plan required for the workflow.

The workflow scenarios provided by SAP can be used as reference when identifying suitable business processes.

Use
This part of the documentation is to be used in conjunction with the user menu for the process consultant. This user can access this user menu on the screen SAP Easy Access via .

Assign the following roles to the user:
- SAP_BC_SRV_USER
- SAP_BC_BMT_WFM_PROCESS

Structure
The documentation for this role consists of:
- Tutorial: Maintaining the Organizational Plan
  Here you will find the tutorial for maintaining an organizational plan. The complete set of tutorials can be found under SAP Business Workflow -Tutorials.

- Creation of Business Process Models
  Here you will find vital documentation for process consultants. In particular, you will find a typical procedure model for this role. The complete documentation on workflow scenarios can be found in the reference documentation on SAP Business Workflow. The complete documentation on maintaining an organizational plan can be found under BC Organizational Management [Extern].

Integration
This role is part of the role-based procedure model containing the following other roles:
Tutorial: Maintaining the Organizational Plan

Purpose

This tutorial shows you how to create the framework for an organizational plan quickly and effectively. An organizational plan describes how an employee is assigned within the organization. You can store the employees who are responsible for carrying out individual business activities in your organizational plan. The organizational plan is maintained for specific clients in the Organizational Management component. An organizational plan created for HR purposes can also be used in SAP Business Workflow as long as the workflow functionality and the HR application are in the same client. Creating your organizational plan also creates the prerequisites for the Workflow System to determine the appropriate agents of a work item at runtime.

Process Flow

The individual units of this tutorial contain step-by-step instructions on how to create the basic framework of your organizational plan in order to reflect the structure and HR environment of your enterprise. The following items, each of which represents a separate unit, provide you with an overview of the tutorial procedure.

1. You create a root organizational unit. Once you have done so, you create the subordinate organizational units.

2. You create jobs, provided that they do not already exist in your job index. You also create positions, which represent the specific instances of jobs in your enterprise. You assign R/3 users as position holders. You can also indicate that a position is a chief position for an organizational unit.

3. You assign tasks to the objects you have created. These objects can be jobs, positions, or organizational units. You can use tasks to describe jobs and positions.

Examples are included to help you work through the individual units. These are intended to illustrate the step-by-step procedure for creating your organizational plan. For further information, see Organizational Management.
Unit 1: Creating an Organizational Structure

An organizational plan consists of the organizational units that exist in the company. The organizational units are related one to the other in a hierarchical reporting structure. However, they can also be created independently of each other.

To create a new organizational plan, you create a root organizational unit. This is the highest unit within the organizational structure, such as the Executive Board. You then set up the organizational structure, starting at the root organizational unit and working downwards.

Procedure

Creating a Root Organizational Unit


2. Confirm the validity period proposed in the dialog box Creating a Root Organizational Unit.

   This takes you to the Create Organization and Staffing (Workflow) screen. This user interface is divided into four screen areas:

   3. On the Basic Data tab in the details area, enter an abbreviation and a name in the Organizational unit input fields for your root organizational unit.

      Abbreviation: <ini_org>

      Name: <Organizational plan(ini)>

4. Choose .

   The root organizational unit you have just created is then selected in the overview area.
Creating further organizational units
Starting from the root organizational unit, you create subordinate organizational units. In this unit, you will create two further organizational units.

1. Choose 

   A new organizational unit is now created below the root organizational unit.

2. Open the details view of the new organizational unit by double-clicking the corresponding line in the display area.

3. In the details area, enter an abbreviation and a description for the Organizational unit in the input fields.

   Abbreviation: <ini_org_sa>
   Name: <Sales (ini)>

   When determining the validity period of objects and relationships, the system chooses the validity period of the superordinate object as standard.

   If you want to change the validity period for the organizational units to be created and their relationships, change the values in the corresponding input fields in the details area.

4. Select your root organizational unit.

5. Choose 

   A new organizational unit is now created below the root organizational unit.

6. In the details area, enter an abbreviation and a description for the Organizational unit in the input fields.

   Abbreviation: <ini_org_rd>
   Name: <Research and development (ini)>

7. Choose 

   You can also include existing organizational units in your organizational plan. To do so, drag them from the selection area to your organizational plan.

Reassigning organizational units
If changes are made to the organizational structure of your company, you can reassign the organizational units involved.

If you reassign organizational units, please note the following:

- If you change the assignment of an organizational unit, you also change the associated relationship records.
- If the validity period of the new relationship overlaps with the validity period of the original relationship, the system delimits the validity period of the original relationship accordingly.


- Root objects cannot be reassigned. In your example, the root object has the abbreviation \textit{ini\_org}.

1. Select the organizational unit that you want to reassign in the selection area.
2. Drag it to the organizational unit under which it is to be assigned.

**Delimiting organizational units**

You delimit organizational units if you want to change their validity period, that is, bring forward the validity end date.

\begin{itemize}
\item When you delimit objects (in this case, organizational units), all of the associated links are automatically delimited at the same time.
\end{itemize}

1. Double-click the organizational unit to open the details view.
2. Switch to display with periods. Choose 
3. Change the entry in the \textit{Valid to} field.

\begin{itemize}
\item You can only change a period if you also change a different entry of the organizational unit at the same time. For example, change the description.
\end{itemize}
Unit 2: Creating Staff Assignments

The staff assignments [Extern] for each organizational unit are maintained. The organizational unit is assigned positions [Extern] in these staff assignments. A position is derived from a descriptive job [Extern] and assigned to one or more users in your company.

All of the positions must be linked to jobs. The positions inherit the attributes and properties of the job.

In this unit, you will create the staff assignments for each of your organizational units. To do so, you will:

1. Create positions
2. Classify a position as a chief position.
3. Assign jobs to positions
4. Assign users to positions

Procedure

Choose Tools → Business Workflow → Development → Definition Tools → Organizational Management → Organizational Plan → Change. The Change Organization and Staffing (Workflow) screen is then displayed.

Create jobs

You must assign every position a job that contains general functions and tasks and which passes on the position. Jobs are normally defined centrally for an organizational plan in the job index. When positions are created, the corresponding job must only be assigned. For this tutorial, you create the positions yourself.

1. Choose Edit → Create job.

   The dialog box Create jobs is displayed. The lower area contains a list of existing jobs and the upper area contains an input table in which you can create new jobs by entering abbreviations and names.

1. In the input table, enter an abbreviation and a name for each of the new jobs.

   Job - head of department:

   Abbreviation: <ini_hd_C>

   Name: <Head of department job (ini)>

   Job administrator:

   Abbreviation: <ini_ad_C>

   Name: <Administrator job (ini)>

   All new jobs receive a validity period from the current date to the 31.12.9999 as standard.

2. Choose ☑.
Unit 2: Creating Staff Assignments

Opening staff assignments for an organizational unit

1. If the root organizational unit (ini_org) that you created in unit 1 is not open for processing, choose organizational unit in the search area, enter a search term and select your organizational unit from the organizational units found by double-clicking in the selection area.

2. In order to open the staff assignments for an organizational unit, select the organizational unit in the overview area and select the arrow on the right next to the Staff assignments (list).

   The staff assignments for the organizational unit selected are now displayed in the overview screen.

Create positions

You create positions for your organizational unit in the associated staff assignments.

1. Open the staff assignments for the organizational unit Research and development (ini).

   In this organizational unit, assign a position for an administrator and a position for a head of department.

2. Choose .

   The new position is now displayed in the staff assignments. You can edit all relevant data for the position in the details area.

   The details view for positions is available in the key date mode and the periods mode. You can toggle between the two modes with the and buttons.

3. In the key date mode, enter the abbreviation and name of the new position in the Position fields on the Basic data tab:

   Abbreviation: <ini_ad_S_rd>

   Name: <Administrator position: (ini) R&D>

   You can change the validity period of the position in periods mode.

4. Repeat steps 2 and 3. Use the following data:

   Abbreviation: <ini_hd_S_rd>

   Name: <Head of department job (ini) R&D>

5. Set the Head of own organizational unit indicator. Confirm the information using the adaptations of the relationship period.

   Chief positions [Extern] are marked in the staff assignments.

   You must define chief positions if you want to use roles [Extern] to determine recipients in workflows that determine a user’s superior.

6. Choose and then the arrow on the right next to and Organizational Structure.
The organizational unit **Research and Development (ini)** is then displayed.

7. Choose to switch to editing of your root organizational unit.

8. Repeat steps 1 - 7 for the organizational unit **sales (ini)**. Use the following abbreviations and names for the positions:

   - **Abbreviation**: `<ini_adsa>`
   - **Name**: `<Administrator position: (ini) Sales>`

   - **Abbreviation**: `<ini_hdsa>`
   - **Name**: `<Head of department job (ini) Sales>`

   Set the **Head of own organizational unit** indicator for the head of department job sales.

**Assign jobs to positions**

You assign a job to every position.

1. Display your created position in the selection area. Choose the entry **Position** in the search area and enter the abbreviation you used (**ini**) as the search term.

   All positions created in this unit are displayed in the selection area.

2. Select the **Administrator position (ini) R&D** for editing by double-clicking.

   The position and the organizational objects assigned to you (job, organizational unit, user) are displayed in the overview area.

3. Display your created job in the selection area. Choose the entry **job** in the search area and enter the abbreviation you used (**ini**) as the search term.

   All jobs created in this unit are displayed in the selection area.

4. Select the job **Administrator job (ini)** in the selection area.

5. Drag the job to the position in the overview area.

6. Choose.

   The job **administrator job (ini)** is assigned the position **administrator position (ini) R&D**.

7. Repeat steps 1 to 6 for the **head of department job (ini) R&D** and assign it the job **head of department job (ini)**.

8. Repeat steps 1 to 6 for the **head of department job (ini) Sales** and assign it the job **head of department job (ini)**.

9. Repeat steps 1 to 6 for the **administrator position (ini) sales** and assign it the job **administrator job (ini)**.

**Assigning users to positions**

When you assign users to positions, you define the **holders [Extern]** of these positions.

This assignment is necessary for workflow applications if you use roles or objects in the Organizational Management to determine recipients.
If you use Personnel Administration (PA), the system recognizes employees who are assigned directly to positions. To ensure in this tutorial that the R/3 user can be established as the agent starting from the employee, employees must have a relationship to R/3 users.

If you do not use Personnel Administration, users are assigned directly to positions. At workflow runtime, they are established directly as the agents of particular single-step tasks.

You can assign the user to the position in the staff assignments or in the single processing of a position. In this tutorial you assign the users using the single processing of a position.

1. Display your created position in the selection area. Choose the entry Position in the search area and enter the abbreviation you used (ini) as the search term.

   All positions created in this unit are displayed in the selection area.

2. Select the position in the selection area for which you want to assign a user by double-clicking.

   The position and the organizational objects assigned to you (job, organizational unit, user) are displayed in the overview area.

3. Display the user assigned to in the selection area. Choose the entry user in the search area and enter an appropriate search term.

   All of the users that match your search criteria are displayed.

4. Select the user whom you want to assign in the selection area.

5. Drag the user to the position in the overview area.

   Users can occupy a position either in full or in part. This depends on the working hours assigned to the position, and on the working capacity of the person or user.

   The staffing percentage refers to the working capacity of a person or user assigned to a position.

6. Choose }.
Creation of Business Process Models

Purpose
The typical process flow depicted here is a suggestion for including *SAP Business Workflow* functions in the modeling of processes. The creation of a business process model is the first step for working with SAP Business Workflow.

Process Flow

The process consultant analyzes the real-life **business processes** of a company and tries to map these processes into an optimized, workflow-compatible **business process model**. This may involve reengineering. The application scenarios [Seite 1703] supplied by SAP should be used as reference for this.

- **Organizational Plan [Seite 1708]**
  - If no company-specific organizational plan exists or if there is one but it cannot be accessed from the system on which the workflow runs, the process consultant must create one.

- **Business Workflow Explorer [Seite 975]**
  - All tasks and workflows in the system can be accessed via the Business Workflow Explorer. This includes the tasks used in the workflow scenarios. Takes the user to the documentation on the scenarios.

Result
This results in a business process model, which is used by the workflow developer to implement a workflow. The work of the process consultant must produce a full organizational plan.
Ad Hoc Workflow

Purpose
There are two types of ad hoc workflow:

Ad hoc agent assignment
A workflow of this type enables agents to be assigned to workflow steps when the workflow is started. During workflow execution the agent assignment can be changed by authorized users at any time for steps not yet completed.

Ad hoc definition
Once you have selected a work item in the Business Workplace, you can start workflows that use the application object of the work item. If the workflow required does not yet exist, you can create it ad hoc.

Prerequisites
To execute an ad hoc definition effectively, tasks groups must be assigned to the application object types. For further information, refer to Defining Task Groups for Ad Hoc Workflows [Seite 1099].

Process Flow

Ad hoc agent assignment
You can convert an existing workflow into an ad hoc workflow in the Workflow Builder. No responsible agents should be entered in the step definitions for all steps to which ad hoc users are to be assigned during execution. Container elements are created automatically by the system in the workflow container and the relevant task containers during conversion into an ad hoc workflow. An expression [Extern] is assigned to the input fields for responsible agents. For further information, refer to Defining Ad Hoc Agent Assignment [Seite 1096].
If this ad hoc workflow is started by a user, this workflow initiator defines the recipients [Extern] who are to execute the individual steps. Throughout the execution of the workflow the workflow initiator can change this assignment of recipients for steps not yet completed at any time. For further information, refer to Executing Workflows with Ad Hoc Agent Assignment [Seite 1097].

Ad hoc definition
If you select a work item in the Business Workplace, you can choose Environment → Start Workflows. This displays a dialog box for starting workflows, in which only the workflows that work with the application object of the selected work item are offered. Here you can define a workflow ad hoc if the workflow you require does not exist yet. The Workflow Builder (with restricted functionality) can be used to define the ad hoc workflow. Tasks are available in the tray My workflows and tasks, which you can use in the workflow definition. For further information, refer to Defining Workflows Ad Hoc [Seite 1098].
Organizational Plan

Definition
Representation of the task-related, functional structure of your enterprise, created using tools from the Organizational Management component.
This functional organizational plan differs from the administrative enterprise structure and the personnel structure whose elements are relevant to Payroll Accounting (company code, personnel subarea or employee group, for example). These structures are found in their corresponding components.

Use
You can create several organizational plans in different plan versions, this provides you with the following options in Organizational Management:

- in one plan version, you depict your current valid organizational plan which you use for your current business processes (evaluations, Workflow, personnel planning, for example).
- In additional plan versions, you can depict organizational plans as planning scenarios (for Business Process Re-Engineering, for example).
You can compare the current organizational plan with the planning scenarios and transfer data from the simulated structures into the current organizational plan.

Structure
Organizational plans are normally created by assigning objects of the following types to each other:

- Organizational Unit
- Position
- Job
- Task
If you are using your organizational plan for Workflow, the following object types are also available:

- Standard task
- Workflow template
- Task group
The main elements of an organizational plan are

- an organizational structure [Extern], with which the reporting structure and task distribution are created using organizational units (departments, for example)
- Staff assignments [Extern] for each organizational unit, in which the current persons (employees), users and vacancies are listed

Integration

Organizational plans are generally related to objects from other components.

- If Integration with Personnel Administration [Extern] is active, the personal data for the staff assignments comes from the Personnel Administration component.

  These functional (organizational plan) and administrative (personnel and enterprise structures) structures come into contact if a person is assigned to an organizational plan (as the holder of a position) as well as an enterprise or personnel structure (that is, to a personnel subarea etc.).

- If the enterprise structure [Extern] is active, account assignment data can be obtained from Controlling.
Working with Screen Areas

Purpose
You use the user interface in the Organization and Staffing or Organization and Staffing (Workflow) view to create, display and edit organizational plans.
The user interface is divided into various areas, each of it which fulfills specific functions.

Together, the search area and the selection area make up the Objektmanager [Seite 1257].
In the search area you can choose between different object types. Depending on whether you are in the Organization and Staffing or Organization and Staffing (Workflow) view, you are offered different object types to choose from.

Each time you access the Organization and Staffing or Organization and Staffing (Workflow) views, the objects that you edited last are automatically called up, though (if appropriate) with a new date. You can continue editing the objects directly. In addition, the last user-specific settings relating to the size of the screen, and in some cases the preview period, are available again.

You can reset user-specific settings in the object manager using the report RH_DELETE_OM_USER_SETTINGS. For more information, see the report documentation.

You can reset the last object selection and the preview period using the report RH_DELETE_NF_USER_SETTINGS. For more information, see the report documentation.

Note that the reset takes place across all applications.
Prerequisites
Before you work with screen areas in the Organization and Staffing or Organization and Staffing (Workflow) views, you must be familiar with Validity [Seite 1251].
You have decided to use either the Organization and Staffing or the Organization and Staffing (Workflow) view. If necessary, choose the required view over Goto → Change view.

Process Flow
A typical work process could look like the following:

1. In the Search area, find [Seite 1259] one or more objects that you want to display or edit, for example:
   - a complete organizational structure
   - all objects of a certain object type, positions, for example
   - one or more objects of a certain object type, for example, a particular organizational unit

2. The objects found are listed in the Selection area. Select one of these objects
   - by double clicking on it, if you want to display [Seite 1263] the object itself including its environment in the overview area and its characteristics in the detail area
   - by clicking on it, if you want to assign [Seite 1269] it to another object using drag & drop; you can assign a position to an organizational unit, for example

3. In the Overview area
   - the selected object including its environment is displayed.
     According to the object type the environment can be displayed as
     - an organizational structure
     - a report hierarchy
     - staff assignments in the form of a list or a tree structure
     - task assignments
     - organizational assignments
     - job assignments
     - a task hierarchy
     - an agent
     - account assignments
   - you can, if required
     - switch between these representations of the object’s environment using.
     - create [Seite 1266], copy [Seite 1268], assign/move/reposition [Seite 1269], or delimit/delete [Seite 1271] new objects.
     - select another object

4. In the Detail area, characteristics of the selected object are displayed [Seite 1263] on tab pages. You can edit [Seite 1263] the characteristics of this object or add new ones, as required.
Using [image], you can hide the detail area, so that you can enlarge the overview area. You can display the detail area again using [image].

In customizing you can define which tab pages should be displayed. For further information, see the Implementation Guide (IMG) under Personnel Management → Organizational Management → Hierarchy Framework.
Validity

Use
An enterprise’s organizational plan is constantly undergoing change. For this reason, *Organizational Management* allows you to edit the organizational structure, staff assignments as well as individual objects according to key dates.
You must be familiar with the concept of validity as it is relevant to every step in the *Organization and Staffing* and *Organization and Staffing (Workflow)* views.

Prerequisites
So that you can use validity completely, make sure
- that you know the period for which you want to find and select objects
- when you are creating objects and object characteristics, that you specify the period in which they are to be valid
- when you are assigning objects and object characteristics that you specify the period in which the assignments are to be valid

Features
You determine the validity of an object or assignment when you create it. You control the display of objects by entering key dates and preview periods. For more information, see *Validity of Objects and Assignments [Seite 1252]*
You determine the validity of object characteristics when you create them. Changes to data are presented in periods. For more information, see *Validity of Object Characteristics [Seite 1256]*.
Validity of Objects and Assignments

Use
So that the presentation of your organizational plan can be exact, objects and assignments between objects exist for a specific period of time. You determine the validity period of objects and assignments when you create them, as required.

Features

Key Date and Preview Period
A key date and a preview period are always set in the Organization and Staffing and the Organization and Staffing (Workflow) views.

- Every time you log on, the current date is set as the **key date**. You can change the key date. Data valid on the date you have selected is displayed.
- When you logon initially, a **preview period** of 3 months is set, that is, all changes to data that happen in this period are displayed. You can change this preview period. Next time you log on, the **preview period** which you selected is set.

System Response
When you create an organizational unit or position in the **overview area**,

- the validity of the object begins with the key date set, this can be moved forward in the detail area
- the assignment begins with the key date set

If you want the validity of the organizational unit, position or assignment to be restricted, you can terminate them. For more information, see Terminating Organizational Units [Extern], or Terminating Positions [Extern].

If you are creating a job or a task, specify the validity date in the dialog box. For more information, see Creating Jobs [Extern] or Creating Tasks [Extern]. Task and job assignments begin on the key date set.

If you are finding objects in the **search area**, the system selects all the objects that exist on the key date and preview period you set, and presents them for selection in the **selection area**. Objects whose validity begins only in the preview period set, are indicated by a in the selection area.

Objects, whose validity ends in the preview period, are indicated by a .

If you have selected an object to be displayed in the **overview area**, according to object type

- the relevant organizational structure and the organizational units assigned to it on the key date or in the preview period is displayed.
- the relevant staff assignments and the positions, persons (employees) or users assigned to it on the key date or in the preview period are displayed.
- the relevant staff assignments and the positions, persons (employees) or users from an organizational unit assigned to the selected job on the key date or in the preview period are displayed.
- the relevant task assignment and the objects assigned to it on the key date or in the preview period is displayed.

Objects whose assignment to another object begins only in the preview period set, are indicated by a .

Objects whose assignment to another object ends in the preview period, are indicated by a .
If you assign an object to another object in the overview area or detail area, the new assignment applies from the key date to the end of the validity of an object. If you have activated Query time period for organizational changes, assignments of organizational units and positions are valid in the period you entered in the Query time period for organizational changes dialog box.

Assignments can only be made in the detail area in key date mode. Change the mode by choosing 

**Activities**

Before you perform a step, check the key date and preview period that are set. Change the settings as required. For more information, see Selecting a Date and Preview Period. Activate/deactivate Query Time Period for Organizational Changes.
Selecting a Date and Preview Period

Use
To display and edit data according to key dates and time periods, select a key date and preview period.

Procedure
1. Choose Settings → Date and Preview period (or).
   The Define Date and Preview Period dialog box appears.
2. Enter the key date in the Start Date field.
3. In the Preview period field, enter the preview period.
   When you confirm your entries with Return, the preview period you selected is displayed.
   Correct your entries as required.
4. Confirm your entries by choosing.

Result
You have set a date and a preview period. You can change these settings at any time.
Activating/Deactivating Query Time Period

Use
The validity of assignments normally begins on the key date you have specified. Alternatively, you can activate a dialog box, in which you can enter any validity date when assigning objects (this does not apply when you are assigning persons (employees) or users to positions).

Procedure
1. Choose Settings → Query time for organizational changes
   The Query time for organizational changes dialog box appears.
2. Select the Query time for organizational changes field, if you want to enter a validity date when you are assigning objects. Otherwise, you can deselect this field.
3. Confirm your entries by choosing 🔄.

Result
If you have activated the time period query, a dialog box will appear when you move or assign objects, you can enter the validity of the assignment in this dialog box.
If you have deactivated the time period query, the validity of the assignment is determined according to the key date.
Validity of Object Characteristics

Use
You store the characteristics of an object on tab pages in the detail area. The characteristics of an object can change with time. For this reason, an exact representation of your organizational plan is available according to a specific time period. You determine the validity of object characteristics, according to your requirements, when you create them.

Features

Key Date and Periods
In the Organization and Staffing and the Organization and Staffing (Workflow) views:
- **period mode** is always set, which displays all existing periods
- **key date mode** is set, this displays the key date and the current period

System Response
Each time data is changed using a tab page in the detail area, a new period is created for this tab page. It is valid from the date in the Valid from field and until the date in the Valid to field. The previous period is automatically terminated on the day before the new period.
A new period is also created, if you make a change outside the detail area, which affects the tab pages, for example, if you create a new position in the overview area.
In key date mode, if more than one period exists for a tab page, this is indicated by a 📅. This indicated that assignments which you are making are valid alongside the current period. If there is only one period, this is indicated by a 📅. In this case, the duration of the new assignment includes the duration of the current period.

Activities
Scroll through the existing periods for object characteristics using ⬅ and ➤ or ⌛ and ⌜. Alternatively, you can use 📅 to select a period directly.
When you are changing data, enter the date from when the data is to be valid in the Valid to field, and the date until it is to be valid in the Valid to field.
By selecting 📅 switch to key date mode, to assign objects in the detail area.
Object Manager

Use
With the object manager you can search for and select objects that you want to display or edit.

Prerequisites
You are familiar with the validity concept of the application. The validity concept determines which objects you can find during a search.

Features
The object manager consists of the search area and the selection area.

1. In the search area are one or more search functions for each object type, for example the Search Term and Structure Search functions. These search functions are marked with . In addition, the object type itself can contain a search function. The object types are marked with the respective object type-specific symbol.

   If necessary, you can add more object types and search functions in customizing. You can also change the sequence of the search functions. For further information see the Implementation Guide (IMG) under Personnel Management → Global Settings in Personnel Management → Settings for Object Manager or the IMG for Organizational Management.

2. In the search area you can create search variants, so that you can reuse search criteria you have grouped together, or hits. These search variants are marked with .
3. In the **selection area** the system displays the objects that you searched for and actually found. According to the search function, this can be either a hit list or a structure.

4. You can scroll through search results in the selection area using ⬅️ and ➡️.

5. You can completely hide or display the **object manager**, so that the other screen areas get correspondingly bigger or smaller. To do that, choose **Settings → Show Object Manager** or **Hide Object Manager**.

6. With ⬅️ you can increase or reduce the size of the selection area, in order to show more hits. As you do that, the search area is hidden or displayed accordingly.

   ![Increase or decrease selection area size](image)

   The system saves the last settings relating to screen size and the last object selection user-specifically, and they are available next time you call up the transaction.

   You can reset these settings and the last object selection using the report RH_DELETE_OM_USER_SETTINGS. Note that the reset takes place across all applications. For more information, see the report documentation.

- You can call up **generic object services** for a selected object using the right mouse button menu. To do that, select an object in the selection area with the right mouse button and choose **Generic Object Services**. For further information about generic object services, see [List of Object Services](Extern).
Finding/Selecting Objects

Use
In the Search area you can search for one or more objects that you want to display or edit. For each object type there are various search functions, for example:

- Search Term
- Structural Search
- Free Search

Prerequisites
Objects you are searching for must already exist. Create new objects as required. The required object types and search functions are set up.

Procedure
1. In the search area select one of the available search functions and if necessary enter the required selection criteria.
   In the selection area the system displays all found objects with the corresponding object type, either as a list or as a structure.
2. Double-click on the required object.
Using Search Tools

Use
With the search tools for each object type you can search for objects in various object type-specific ways. These search tools are marked with \[\text{icon}\]. In addition, the object type itself can contain a search tool. The object types are marked with the respective object type-specific symbol.

Prerequisites
You are familiar with how the search tools are assigned.

1. In the search area, the object type itself can contain a search tool. The object types are marked with \[\text{icon}\].

2. Search tools can be positioned under the corresponding object type. In that case it is marked with \[\text{icon}\].

Procedure

Searching for Objects Using a Search Term
1. Choose \textit{Search Term} (or the required object type, if it has this search function).
   The \textit{Search for \textless object type\textgreater} dialog box appears.

2. Enter a name. This can be a name, abbreviation or numeric ID. You can also search using the entry *.

3. Restrict the number of hits, if required. Enter whether the object you are looking for is \textbf{directly} or \textbf{indirectly} assigned to another object.

4. Choose \textbullet Search.
   The results of the search are displayed in the \textbf{selection area}. The display in the overview and detail areas does not change.

5. If necessary, you can start another search for the same object type and then choose \textbullet Insert to add the new hits to the first results in the selection area.

6. Double-click on the required object in the selection area.

Searching for Objects Using Free Search
The \textit{Free Search} search tool uses the \textit{InfoSet Query}.

1. Choose \textit{Free Search} (or the required object type, if it has this search function).
   The \textit{Find Objects of Type <Object type>} dialog box appears. For further information, see \textit{HR in the InfoSet Query [Extern]}. After the search, the system displays the search results in the \textbf{selection area}.

2. Double-click on the required object in the selection area.

Searching for Objects Using Structure Search
1. Choose \textit{Structure Search} (or the required object type, if it has this search function).
   In the \textbf{selection area} the system displays all found objects of the relevant object type in a tree structure, ordered according to their assignment in the organizational plan.
If necessary, you can refresh the hits displayed in the selection area using 🔄. This is recommended, for example, if you have created new objects shortly before.

2. Expand the structure until the required object is revealed.

To display unrelated objects, choose 🔄.

3. Double-click on the required object in the selection area.
Using Search Variants

Use
In some search functions, for example Search Term, you can restrict the number of hits by using a combination of selection criteria. You can then save such a combination as a search variant so that you can use it again. You can also delete a search variant again, if necessary.

Procedure

Creating Search Variants
1. Use one of the search functions to search for objects.
   The system displays the hits in the selection area.
2. Choose and enter a name. Choose .
   The system saves the search criteria as a search variant and assigns them to the corresponding object type in the search area. The search variant is marked with .
3. Select the search variant you created and choose . You can check your search criteria.

Searching for Objects Using a Search Variant
1. Select a search variant.
   The hits are displayed in the selection area.
2. Double-click on the required object.

Deleting Search Variants
1. Select the search variant that you want to delete.
2. Choose .
   The search variant is deleted.
Displaying/Editing Objects

Use
You can allocate numerous characteristics to the organizational objects in an organizational plan. Select an object in the selection area or overview area so that you can display the characteristics of the object in the detail area and, if required, edit them.

Prerequisites
You are familiar with Validity [Seite 1251]. Select a key date and preview period. It can be that the detail area is hidden. In order to display and edit object characteristics, you may need to redisplay the detail area using ...

Procedure

Selecting and Displaying Objects
You can select objects either in the selection area or in the overview area:

- In the overview area, select an object by double-clicking on it.

You can only select an object from the selection area if you have already searched for one or more objects in the search area [Seite 1259].

The object itself is displayed in the overview area. The following display types are possible:

- **Organizational Unit**
  - Organizational structure: structural display showing the assignment of an organizational unit to other organizational units.
  - Task assignment: structural display showing the assignment of an organizational unit to tasks and activity groups.
  - Report hierarchy: structural display showing the assignment of an organizational unit to a chief position and its holder (person or user).
  - Staff assignments (list): list display showing the assignment of an organizational unit to the subordinate positions and their holders.
  - Staff assignments (structure): structural display showing an organizational structure and the assignment of each organizational unit to the subordinate positions and their holders.

- **Position**
  - Task assignment: structural display showing the assignment of positions to tasks and activity groups.
  - Report hierarchy: structural display showing the higher- and lower-level assignments of positions.
  - Organizational assignment: structural display showing the assignment of a position to higher-level organizational units.
**Job**
- *Task assignment*: structural display showing the assignment of jobs to tasks and activity groups.
- *Job usage*: structural display showing the assignment of a job to positions.

**Person**
- *Organizational assignment*: structural display showing the assignment of a person over positions to higher-level organizational units.
- *Task assignment*: structural display showing the assignment of persons to tasks and activity groups.
- *Report hierarchy*: structural display showing the higher- and lower-level assignments of positions and their holders.

**User**
- *Organizational assignment*: structural display showing the assignment of a user over positions to higher-level organizational units.
- *Task assignment*: structural display showing the assignment of users to tasks and activity groups.

**Task, Standard Task, Workflow Template, and Task Group**
- *Task hierarchy*: structural display showing the assignment of a task to other tasks and activity groups.
- *Agent*: structural display showing the assignment of a task to other organizational objects.

You can display the exact evaluation path for the current display using 

To switch between displays, select the required object and choose . The system proposes the various display types available.

To move up a level within a displayed structure, choose .

- In the **overview area**, double-click on an object that is displayed in a structure or list. The characteristics of the selected object are displayed in the **detail area**.

**Scrolling**

In the **overview area** you can scroll through the displays using ‹ and ‡: If you have already made changes to data but have not yet saved them, you can use ‹ and ‡ to undo or recreate the last change. In the **detail area**, you can use ‹ and ‡ or † and ‡ to scroll through the periods that exist for an object.

**Editing the Characteristics of an Object**

In the **detail area** you can edit the characteristics of the object you selected.

1. Select a tab page.
2. Make the appropriate entries. For more information, see
   - Editing Organizational Units [Exter]
   - Editing Positions [Exter]
3. Save your entries by choosing 📊.

   On tab pages containing characteristics there is a ✅.
Creating Objects

Planstelle bearbeiten [Extern]
Stelle bearbeiten [Extern]
Aufgabe bearbeiten [Extern]

Use
You depict the functional structure of your enterprise as an organizational plan, by creating organizational objects (organizational units, positions etc.) in the overview area and assigning them to objects that already exist.

Prerequisites
To create a root organizational unit for a new organizational plan, you must be in Create mode (Organization and Staffing Create or Organization and Staffing (Workflow) Create).
Create additional organizational objects
• in Create mode, if you are going to create additional objects straight away once you have created the root organizational object, without leaving the mode.
• in Change mode (Organization and Staffing Change or Organization and Staffing (Workflow) Change), if you have left create mode.

You are familiar with Validity [Seite 1251]. Select a key date and preview period. Which objects you can create depends on which display type is currently in the overview area. If necessary, check which assignments can be displayed in your particular case, using . If necessary, use to switch to a display type in which you can carry out the required assignment.

For more information on the display types, see Displaying/Editing Objects [Seite 1263]

Procedure

Creating Root Organizational Units
You have chosen create mode.
1. In the Create a root organizational unit dialog box, enter the required validity period of the root organizational unit and confirm with .
   The system creates a root organizational unit with the provisional name “New Organizational Unit”.
2. In the detail area, replace the preset entries for object name and object description on the Basic data tab page with the name of your choice.
3. Save your entries by choosing .

You have created your root organizational unit.

Creating Organizational Objects in the Structural Display
1. In the overview area, select the object to which you want to assign the object being created.
2. Choose .

The Choose Relationship dialog box appears.
3. Select an object type.
   The system creates the object.
4. Over the tab pages in the **detail area**, edit the characteristics of the new object.
   For further information according to object type, see:
   - Editing Organizational Units [Extern]
   - Editing Positions [Extern]
   - Editing Jobs [Extern]
   - Editing Tasks [Extern]

**Creating Positions in Staff Assignments (List)**

1. Choose 📝.
   The system creates the new position.
2. Over the tab pages in the **detail area**, edit the characteristics of the new object.

   You can also create new objects by copying. For more information, see [Copying Objects [Seite 1268]].
   You can always assign objects to other objects – not just when you create them. For more information, see [Assigning Objects [Seite 1269]].
Copying Objects

Use
You can create a new object by copying an object which already exists. When you do this, all the characteristics of the object are copied.

Prerequisites
You are familiar with Validity [Seite 1251]. Select a key date and preview period.

Procedure
1. In the overview area, select the object which you want to copy.
2. Choose \[\text{Copy Object}\].
   The Copy Object dialog box appears.
3. Specify the number of copies you wish to make and the validity period which is to apply to the new objects.
4. Choose \[\text{Create}\].
   The new object is created. The object’s tab pages will be filled with the characteristics of the template. The characteristics can be changed or enhanced as required.
5. Save your entries by choosing \[\text{Save}\].
Assigning/Moving/Repositioning Objects

Use
An organizational plan depicts the functional relationships in your enterprise. To depict such relationships (the hierarchy of organizational units or the staffing of positions by persons, for example), you assign objects to each other. Objects can be
- newly assigned if they are not yet assigned.
- moved, that is, you end the object’s current assignment in a structure and create a new assignment within the same structure.
- repositioned within a hierarchy level in a structure.

The following can be target positions:

- An organizational unit can be assigned to objects of the following types:
  - Organizational Unit, Task, Standard Task, Workflow Template, Task Group

- A position can be assigned to objects of the following types:
  - Organizational Unit, Task, Standard Task, Workflow Template, Task Group

- A person (employee) or user can be assigned to objects of the following types:
  - Position, Task, Standard Task, Workflow Template, Task Group

- A position can be assigned to objects of the following types:
  - Position, Task, Standard Task, Workflow Template, Task Group

- A task, a standard task, a workflow template or a task group can be assigned to objects of the following types:
  - Organizational Unit, Position, Job, Person, User, Task, Standard Task, Workflow Template, Task Group.

Prerequisites
You are familiar with Validity [Seite 1251]. Select a key date and preview period.

- You can only assign standard tasks, workflow templates and task groups in the Organization and Staffing (Workflow) view.
- Which objects you can assign/move/reposition depends on which display type is currently in the overview area. If necessary, check which assignments can be displayed in your particular case, using . If necessary, use to switch to a display type in which you can carry out the required assignment.

For more information on the display types, see Displaying/Editing Objects [Seite 1263]

Procedure
Assigning Objects
1. Search for an object that already exists, but does not yet belong to the current structure. For more information, see Finding Objects [Seite 1259].
2. In the selection area, select the object that you want to assign to another object.
Assigning/Moving/Repositioning Objects

3. Holding down the left mouse button, drag the object to the target position. The target position can be an object in a structural display or a field in the staff assignments (list) display.
   
   The system assigns the object.

4. Save your entries by choosing .

Moving Objects

1. In the overview area, select the object that you want to move.
2. Hold down the left mouse button and drag the object to the object to which you want to assign it.

   The object has moved and is therefore newly-assigned.

3. Save your entries by choosing .

Repositioning Objects

1. In the overview area, select the object that you want to reposition.
2. Choose or to reposition the object in a higher or lower position within the hierarchy level.

   The object has been repositioned. Therefore, the sequence of objects on a hierarchy level has changed.

3. Save your entries by choosing .

Assigning/Moving Using a Pushbutton

As an alternative to assigning or moving with drag & drop, you can also use the pushbutton.

1. In the overview area, select the object that you want to assign to another object.

2. Choose .

   The Restrict allowed values dialog box appears.

3. Search for the object that you want to assign, select it and then choose .

4. Save your entries by choosing .
Terminating/Deleting Objects or Assignments

Use
You can:

- Terminate organizational objects if you want to limit their validity.
- Terminate the assignment of an organizational object to another organizational object if you want to limit the validity of this assignment.
- Delete an organizational objects, if you want to delete them completely, including their history.
- Delete the assignment of an organizational object to another organizational object, if you want to delete this assignment and its history completely.

Prerequisites
You are familiar with Validity [Seite 1251]. Select a key date and preview period. Choose key date mode in the detail area.
Which objects/assignments you can terminate/delete depends on what is displayed in the overview area. Check which assignments can be depicted in a particular case by choosing . If necessary, switch to a display in which you can terminate/delete the object or assignment by choosing .

For more information on the displays, see Displaying/Editing Objects [Seite 1263].

Procedure
1. In the overview area, select the object which you want to terminate/delete or whose assignment you wish to delete.
2. Choose if you want to terminate, or if you want to delete.
3. Choose whether you want to terminate/delete the object or the assignment.

   If you want to terminate, enter the date from which the object/assignment is to be terminated in the dialog box, and choose .

   The object / the assignment is terminated/deleted.

4. Save your entries by choosing .
Undoing/Recreating Changes

Use
If you have already made changes to data in the overview area, but have not yet saved them, you can use and to undo or recreate them.

Prerequisites
You have changed data but you have not yet saved.
A change (no assignment) in the detail area: You have confirmed the change with Return, or you carried out another activity after the change, chosen another tab page, for example.

Procedure
1. Choose to undo the last change.
   The change is undone.
2. Choose to recreate the last change.
   The change is recreated.
3. Save your entries by choosing when you want to finally confirm your changes.
Confirmation Prompt

Use
Once you have carried out a specified number of changes to data (specified in Customizing), a dialog box appears asking whether you want to save these changes. Thus you can avoid losing data.

Prerequisites
You define the number of steps that are carried out before the confirmation prompt appears in Customizing. For further information, see the Implementation Guide (IMG) under Personnel Management → Organizational Management → Framework → Set Up Confirmation Prompt.

Activities
Save data as required.
Configuring Columns

Use
You can decide which columns should be displayed in different screen areas.

Prerequisites
The columns that are available are determined in Customizing. For further information see the Implementation Guide (IMG) under Personnel Management → Global Settings in Personnel Management → Column Framework or the IMG for Organizational Management.

Procedure
1. Choose ☐
   The Column configuration dialog box appears.
2. Select the columns that you want to display.
   Some entries represent column groups, that is, more than one column will be displayed if you select one of these entries.
3. Confirm your entries by choosing ☑.

Result
The columns you selected are displayed.

You can reset all of a user’s column configurations using the report RH_DELETE_COL_USER_SETTINGS. Note that the reset takes place across all applications. For more information, see the report documentation.
Switching Maintenance Interfaces

Use
You can replace the Organization and Staffing and the Organization and Staffing (Workflow) views with Simple Maintenance. You may want to switch if, for example, you have already used Simple Maintenance and you want to work with the maintenance interface you are used to. For more information on Simple Maintenance, see Simple Maintenance [Extern]. In addition, you can jump to Infotype maintenance from a selected object. For more information, see Infotype Maintenance [Extern].

Procedure

Situation A: You are in Organization and Staffing or Organization and Staffing (Workflow) view and want to switch to Simple Maintenance.
Choose Settings → Maintenance Interface. You access Simple Maintenance. If you do not change this setting, you will automatically access Simple Maintenance when you call up a maintenance interface in the future.

Situation B: You want to reverse this setting. You are in Simple Maintenance.
Choose Settings → Maintenance Interface. The Organization and Staffing interface appears. When you call up a maintenance interface in future, the Organization and Staffing view will appear.

Situation C: You are in neither of the two maintenance interfaces; you want to select an interface.
In the SAP menu choose Human resources → Organizational management → Settings → Set maintenance interface. The Set maintenance interface dialog box appears. Select a maintenance interface and confirm by choosing ✔. If you do not change this setting, you will automatically access the interface you select when you call up a maintenance interface in the future.

Situation D: You have selected an object in the Organization and Staffing maintenance interface and displayed its characteristics in the detail area. You want to go to the infotype maintenance for this object.
Choose Goto → Detail object → Enhanced object description. To get back to the Organization and Staffing view, choose 🔄.
Business Workflow Explorer

Use
You use the Business Workflow Explorer [Extern] to display an overview of all tasks for a selected search range. All the tasks displayed can also be processed from the Business Workflow Explorer.

Prerequisites
You have to define a search range when you first call the Business Workflow Explorer. When the Business Workflow Explorer is called thereafter, the last active search range is used. The Business Workflow Explorer is only available on a 32-bit platform.

Features
In the left-hand screen area, the Business Workflow Explorer displays all the tasks assigned to the selected search range.
In the right-hand screen area, the Business Workflow Explorer displays the workflow (with associated tasks) or the task, which you selected in the left-hand screen area.

Activities
You can call the Business Workflow Explorer by choosing Tools → Business Workflow → Development → Definition tools → Business Workflow Explorer.

Task-related functions
The functions available on the tasks displayed are shown in the context menu.

Quick view
With tasks an overview of the most important properties is displayed and with workflows the workflow definition is displayed.

Display or change
A new session is opened to display or change.

Create
You can create new tasks and workflows. If you have selected a workflow you create a new workflow template, and if you have selected a task you create a new standard task.

Workflows of a definition
You go to the work item selection in which the task ID is already entered. Further information on carrying out the selection, refer to Work Item Selection [Seite 1490].

Where-used list
All workflow definitions in which the task is used are displayed.

Choose another search range
This enables you to change the Business Workflow Explorer search range. The search range can be limited according to specific criteria. These criteria are:
- Task groups [Extern]
- Application components
- Your own or all tasks and workflows saved locally
- Tasks and workflows with the SAPphone property
The last ten workflows processed
Workflows defined as demo workflows or as verification and test workflows.
Tasks that refer to a particular object type and possibly a particular method

If you define a new search range, it replaces the last active range.

Search / extended search
You can use this to search for tasks and workflows. The standard search function only searches for the term in your current search range. The extended search covers all existing tasks and workflows.

Copy
You select a task or a workflow in the right-hand tree and choose Copy. The selected task/workflow is copied to form a new task/workflow. You only see the new task/workflow in the current session of the Business Workflow Explorer as well if it is in the search range displayed.

If you copy a task of the type T (customer task) or a workflow of the type WF (workflow task), the copy will be defined as a standard task or workflow template since customer tasks and workflow tasks will not be supported in the future.

Detailed view
You can activate a detailed view in the right-hand tree by choosing View → Details. This displays attributes of the task or workflow. The business object type used is also displayed with tasks.
Business Workplace: Workflow Functions

Use
You use this part of the Business Workplace if you want to use the functions of SAP Business Workflow.

The Business Workplace is the main interface between an end user and the workflow system. All dialog and missed deadline work items to which the user is assigned as a recipient are displayed in the user’s workflow inbox.

As a head of department, you are responsible for approving leave requests. The relevant approval process is implemented using a workflow in your enterprise.

The requests (in the form of work items) appear in your worklist (workflow inbox) and must be rejected or approved there.

The rejected or approved requests (executed work items) are not only returned to the applicants after processing, but are also put into your workflow outbox (under Work items executed by me). You can therefore check the requests you have processed.

Features

Workflow settings
You can configure the workflow functions in the Business Workplace using the personal workflow settings.

Business Workplace screen areas
The Business Workplace has three screen areas, which are used in the following manner by SAP Business Workflow:
Overview tree

The following workflow functions are available under the *Inbox* node, which is under the initial node *Workplace*:

- **Workflows**
  - Grouped according to task
  - Grouped according to content
  - Grouped according to content type
  - Grouped according to sort key
- Overdue entries
- Deadline messages
- Incorrect entries

For information on these functions, refer to *Workflow Inbox [Seite 1408]*.

The following functions are available under the *Outbox* node, which is under the initial node *Workplace*:

- Started workflows
- Work items executed by me
- Forwarded work items

For information on these functions, refer to *Workflow Outbox [Seite 1441]*.

The *Resubmissions* node is located under the initial node *Workplace* and contains the:

- *Workflow resubmissions [Seite 1443]*
Worklist
The worklist [Extern] is displayed in the upper right corner of the Business Workplace screen. Depending on whether you are in the workflow inbox, the workflow outbox or the workflow resubmissions, you have various functions available to you, which are described at the respective locations.

Work item preview
In the lower right corner of the Business Workplace screen, a work item selected in the worklist is displayed in a preview [Seite 1445]. Not all the functions of the work item display or the workflow log are available. A user exit [Extern] can be used to configure the work item preview to suit your individual requirements.

Support for context menus
All workflow functions can be called using the relevant context menu.

Workflow Toolbox
SAP Business Workflow's Workflow Toolbox [Seite 1446] enables the user to access workflow functions even during a workflow-driven application transaction.

E-mail notification for new work items
The report RSWUWFML can be used to inform an employee by mail that there is a new work item in their Business Workplace inbox. This function is therefore beneficial to all employees who do not work with their Business Workplace on a daily basis.

Activities
To access the Business Workplace from the SAP Easy Access screen, choose one of the following options:

-  
- Menu → Business Workplace
Work Item

Definition
Object that represents a task or action in the workflow system at runtime.

Use
Work items are subdivided into a specific work item type according to their assignments. The internal processing procedures are controlled via this work item type. The work item type determines which statuses and transitions are valid.
Depending on the work item type, some of these work items are displayed in a user's work list. Other work items, on the other hand, are only used and processed internally.

Structure

Work item types displayed in the Business Workplace

<table>
<thead>
<tr>
<th>Work item type</th>
<th>Short text</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>W</td>
<td>Work item with dialog</td>
<td>Runtime representation of single-step tasks that require interaction with the user. Runtime representation of user decisions</td>
</tr>
<tr>
<td>D</td>
<td>Missed deadline</td>
<td>Work item for notification of missed deadline</td>
</tr>
<tr>
<td>A</td>
<td>Work Queue</td>
<td>A work queue is a list of objects to be processed once and together in a limited time frame.</td>
</tr>
</tbody>
</table>

Other work item types

<table>
<thead>
<tr>
<th>Work item type</th>
<th>Short text</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>F</td>
<td>Workflow</td>
<td>Runtime representation of a multistep task</td>
</tr>
<tr>
<td>B</td>
<td>Batch item</td>
<td>Runtime presentation of a single-step task that runs in the background</td>
</tr>
<tr>
<td>E</td>
<td>Wait step work item</td>
<td>Runtime representation of a wait step in the workflow definition</td>
</tr>
<tr>
<td>C</td>
<td>Container anchor</td>
<td>This type of work item is required as a special development in the EDI environment. It does not normally appear in the workflow environment. It is recommended that work items of this type be regularly deleted or archived.</td>
</tr>
</tbody>
</table>
Dialog Work Items (Type W)

Definition
Work item that represents a task at runtime that requires interaction with the user.

Since the user decision is also represented internally by a task, a dialog work item can also represent a user decision.

When a dialog work item is executed, the underlying object method of the task is called. The deadlines for executing dialog work items are monitored.

Use
A dialog work item is displayed with ready status in the workflow inbox of the Business Workplace. It is removed from the integrated inboxes of the other agents when the recipient reserves, executes, or processes this work item with other functions. The database oriented approach used in SAP Business Workflow allows a work item to be seen by several recipients equally authorized in organizational terms in their inboxes and executed from there. However, only one recipient can actually reserve this work item for processing and execute it. The work item is then no longer available to any other recipients.

Integration
A task represented by a dialog work item can be

- a step in a workflow definition:
  In the workflow definition, reference is made to tasks in the activity and user decision steps.

- started as single steps via an event or in dialog:
  Tasks can be started as elementary activities directly in dialog or via a triggering event. These tasks are then also represented by a dialog work item in the workflow inbox.
## Status of a Dialog Work Item

The valid statuses for dialog work items (type W) are listed in the following table.

<table>
<thead>
<tr>
<th>Work item status</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>waiting</td>
<td>The work item has been scheduled for its requested start. A work item has this status:</td>
</tr>
<tr>
<td></td>
<td>• if it already exists but the requested start specified in the workflow definition has not been reached yet.</td>
</tr>
<tr>
<td></td>
<td>• if it has been set to resubmission</td>
</tr>
<tr>
<td></td>
<td>Work items in the waiting status are not displayed in the workflow inbox.</td>
</tr>
<tr>
<td>ready</td>
<td>The work item has been released for execution and appears in the workflow inbox of all recipients.</td>
</tr>
<tr>
<td>reserved</td>
<td>The work item has been received by one of its recipients with the result that its status has changed from ready to reserved. A work item in the reserved status is then displayed to this recipient only. It is no longer displayed in the workflow inboxes of the other recipients.</td>
</tr>
<tr>
<td>In process</td>
<td>The work item is currently being processed by a recipient or in a different mode. A work item also has this status:</td>
</tr>
<tr>
<td></td>
<td>• if the work item is waiting for its terminating event.</td>
</tr>
<tr>
<td></td>
<td>• if the user cancelled the method.</td>
</tr>
<tr>
<td></td>
<td>• if the method was ended with a temporary exception for which no subsequent steps have been modeled.</td>
</tr>
<tr>
<td></td>
<td>The point at which processing is completed cannot be detected by the workflow system in this status. As long as the status of the work item is set to in process, database changes have not been made.</td>
</tr>
<tr>
<td>Executed</td>
<td>The work item is awaiting explicit confirmation of its completion. The work item only has this status if it is necessary to confirm that it has been completed. A work item with executed status can be executed or forwarded several times until it is set to the status done in the Business Workplace. In this way, groupware components are realized in SAP Business Workflow.</td>
</tr>
<tr>
<td>completed</td>
<td>The execution of the work item is completed. The result of the task represented by the work item is correct, i.e. the result modeled in the workflow definition. Work items in the completed status are not displayed in the workflow inbox of the Business Workplace.</td>
</tr>
<tr>
<td>Logically deleted</td>
<td>Execution of the work item is no longer meaningful or required by the workflow logic. A work item changes to the logically deleted status in the following way:</td>
</tr>
<tr>
<td></td>
<td>• Termination in parallel processing branches</td>
</tr>
<tr>
<td></td>
<td>When the required number of processing paths has been executed in a fork, the work items in the other paths that have not yet reached the completed status are automatically set to the logically deleted status.</td>
</tr>
<tr>
<td></td>
<td>• Intervention by an administrator</td>
</tr>
<tr>
<td></td>
<td>An administrator can only set a work item to the logically deleted status if it has not yet reached the completed status and is not part of a higher-level workflow.</td>
</tr>
</tbody>
</table>
## Status of a Dialog Work Item

<table>
<thead>
<tr>
<th>Status</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Work items in the <strong>logically deleted</strong> status are not displayed in the workflow inbox of the Business Workplace.</td>
<td></td>
</tr>
<tr>
<td>A work item with the <strong>logically deleted</strong> status may have caused database changes or other actions (generate event, send notification). These changes are not canceled automatically (compensated).</td>
<td></td>
</tr>
<tr>
<td>Error</td>
<td>Execution of the work item was terminated with an error.</td>
</tr>
</tbody>
</table>
Status Transitions of a Dialog Work Item

The diagram below shows the possible status transitions that a dialog work item (type W) can undergo:

The arrows are labeled with the functions used by a user to trigger the respective status transition.

**Comments**

For further information, refer to [Status of a Dialog Work Item](#).  

**Transition from status waiting**

The transition from the status waiting to the status ready is performed automatically by the work item manager when the requested start is reached. A workflow system administrator can set a work item to the status ready manually.

**Transition from status ready**

From this status, the work item passes either to the status reserved or via the status in process to the status completed.

**Transition from status reserved**

A work item with the status reserved can be reset to the status ready.

**Transition to and from status in process**

A work item with the status in process can be reset to the status ready. A workflow system administrator can reset a work item manually. This function is available when changing the work item.

**Transition from status executed**

After confirmation of end of processing, the work item assumes the status completed.
Status Transitions of a Dialog Work Item

Transition from status completed
Work items with the status completed can no longer be set to another status even if a workflow system administrator intervenes.

Transition from status error
A workflow system administrator can intervene and set work items with errors to the status in process or the status logically deleted (possibly after eliminating the error).

Transition from status logically deleted
Work items with the status logically deleted can no longer be set to another status even if a workflow system administrator intervenes.
**Missed Deadline Work Item (Type D)**

**Definition**
Notification of a deadline recipient if the runtime system detects that the deadline for a certain work item has been exceeded.

**Use**
This work item informs its recipients that a deadline (start or end deadline) of a monitored work item has been exceeded. The recipients are informed by means of a missed deadline work item (type D) in the workflow inbox of the Business Workplace. When it is executed, this work item displays information on the monitored (and now late) work item. The text for notifying the recipient is set by default. When a deadline is monitored for an activity [Seite 1024] or user decision [Seite 1074], it is also possible to enter an individual text in the respective task definition.
Status of a Missed Deadline Work Item

The valid statuses for missed deadline work items (type D) are listed in the following table:

<table>
<thead>
<tr>
<th>Work item status</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>✅ Ready</td>
<td>The work item appears in the Business Workplace of the recipients.</td>
</tr>
<tr>
<td>🔄 In process</td>
<td>One of the recipients has executed the work item</td>
</tr>
<tr>
<td>🏷 Completed</td>
<td>The execution of the work item is completed. Work items in the <em>completed</em> status are not displayed in the workflow inbox of the Business Workplace.</td>
</tr>
</tbody>
</table>
Status Transitions of a Missed Deadline Work Item

The following status transitions are possible for missed deadline work items (type D):

**Transition to and from status in process**

The status of a missed deadline work item changes from *ready* to *in process* when it is executed by a user. Execution of a missed deadline work item displays the most important information on the monitored work item. The missed deadline work item remains in the *in process* status until end of processing is confirmed explicitly. Until then, the work item can be executed or forwarded several times.

**Transition to completed status**

The status of the missed deadline work item changes to the *completed* status when completion of processing has been confirmed explicitly.
Work Queue Work Item (Type A)

Definition
Work item that represents a work queue.
A work queue comprises a list of objects that are to be processed once together within a specified period.
The work queue serves as a framework for the individual entries to be processed and manages the list of
objects to be processed including their statuses and the tasks to be performed on them.
Work queue work items are displayed in the Business Workplace. The work item status indicates the
overall processing status of the work queue.

Use
Once you have created the work queue work item and you know its work item ID, you have the following
options:
• Process the work queue within a workflow.
• Control the release, processing and status evaluation of the work queue with function
  modules.
• Control the release and processing of the work queue by processing the work queue work
  item directly.

Integration
To create the work queue work item from the list, you call the function module SWZ_AI_CREATE.
## Status of a Work Queue Work Item

A work queue work item can have the following statuses:

<table>
<thead>
<tr>
<th>Work item status</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>✚ waiting</td>
<td>The work queue has been created but cannot be released yet.</td>
</tr>
<tr>
<td>📩 ready</td>
<td>The work queue work item (type A) is displayed in the Business Workplace of the recipients passed in the table AGENTS of the function module SWZ_AI_CREATE.</td>
</tr>
<tr>
<td></td>
<td>If the work queue is to be processed via a workflow and therefore not appear as a separate work item in any Business Workplace, a non-existent user must be specified in the table when the work queue is created.</td>
</tr>
<tr>
<td></td>
<td>The work queue must be reserved for further processing.</td>
</tr>
<tr>
<td>🎯 reserved</td>
<td>A person with release authorization has reserved the work queue.</td>
</tr>
<tr>
<td>🌟 In process</td>
<td>Work queue processing has begun.</td>
</tr>
<tr>
<td></td>
<td>– Dialog work items (type W) have been created for entries with dialog.</td>
</tr>
<tr>
<td></td>
<td>– For entries without dialog marked accordingly, the methods have been executed directly.</td>
</tr>
<tr>
<td>🎯 completed</td>
<td>All lines in the work queue have the status COMPLETED or CANCELLED and have reported back accordingly to the work queue.</td>
</tr>
<tr>
<td></td>
<td>A work queue work item (type A) that assumes this status automatically creates the event created.</td>
</tr>
<tr>
<td>🕰 Logically deleted</td>
<td>Further processing of the work queue is invalid (and therefore no longer possible).</td>
</tr>
<tr>
<td>✊ error</td>
<td>At least one line of the work queue has the status error.</td>
</tr>
</tbody>
</table>
Workflow Work Item (Type F)

Definition

Work item that represents a multistep task at runtime.

Use

For every multistep task started there is one type F work item. The workflow log and the workflow container can be accessed:

- For error diagnosis and error correction if no work items are displayed
- For information on steps of a workflow (including their current agents, notes, and ad hoc objects) already processed
- For modifying an ongoing workflow by changing the workflow container
- For reporting on completed processes

Type F work items are not displayed in the Business Workplace but can be found using the work item selection [Seite 1490].

Structure

A workflow consists of a sequence of work items that are executed by agents or the system. The work items represent the steps in the workflow definition that refer to a particular task. These are steps of the types activity [Seite 1024] or user decision [Seite 1074].

Integration

Workflow Execution

The Work Item Manager manages the processing of work items and monitors deadlines. To automate workflow processes, activities in the workflow can also refer to object methods which run in the
background. If this is the case, the work item manager initiates the calling of the background processes. The work items whose execution requires dialog can be accessed by the selected agents (determined from the organizational model and role resolution) from their worklists in order to select them for processing. This worklist is displayed and managed in the workflow inbox of the Business Workplace [Seite 1368].
Status of a Workflow Work Item

The valid statuses for workflow work items (type F) are listed in the following table:

<table>
<thead>
<tr>
<th>Work item status</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Waiting</td>
<td>The (sub) workflow is scheduled but its requested start has not yet been reached.</td>
</tr>
<tr>
<td>Ready</td>
<td>Temporary interim status</td>
</tr>
<tr>
<td>In process</td>
<td>Execution of the (sub) workflow has begun.</td>
</tr>
<tr>
<td>Completed</td>
<td>The end of the (sub) workflow has been reached.</td>
</tr>
<tr>
<td>Logically deleted</td>
<td>Execution of the (sub) workflow is no longer required or meaningful. A status of a workflow changes to logically deleted in the following way:</td>
</tr>
<tr>
<td></td>
<td>• Intervention by an administrator. An administrator can only set a workflow item to the logically deleted status if it has not yet reached the completed status.</td>
</tr>
<tr>
<td></td>
<td>• Termination in parallel processing branches When the required number of processing paths has been executed in a fork, the work items in the other paths that have not yet reached the completed status are set to the logically deleted status.</td>
</tr>
<tr>
<td></td>
<td>A workflow set to the logically deleted status is recursively scanned for dialog and workflow work items (type W or F) that do not yet have the status &quot;completed&quot;. These work items are then also set to the status logically deleted.</td>
</tr>
<tr>
<td></td>
<td>A work item changed from the executed status to the logically deleted status may already have caused database changes or other actions (generate event, send notification). These changes are not canceled automatically (compensated), but are recorded in the log.</td>
</tr>
<tr>
<td>Errors</td>
<td>An error occurred in workflow control.</td>
</tr>
</tbody>
</table>

Workflows and their statuses are not displayed in the Business Workplace. Information about the workflow can be accessed via the subordinate dialog work items.
Status Transitions of a Workflow Work Item

The diagram shows the possible status transitions.

Status and Status Transitions of Work Items (F)

The statuses of a workflow work item and the permissible status transitions always concern an entire workflow.
A step (activity) in the workflow definition can reference both a task and another workflow. A workflow can therefore also contain subordinate subworkflow items.

Comments

Transition from status waiting
The work item manager automatically carries out the transition from the waiting status to the ready status when the requested start date/time of the workflow has been reached.
A workflow system administrator can set a work item to the status ready manually.

Transition from status ready
The ready status is a temporary interim status because it only exists until the first work item of the workflow has been created.

Transition to and from status in process
The status of the workflow changes to in process as soon as the first work item of this workflow has been created. The workflow remains in this status until the entire workflow definition has been processed.

Transition to incorrect status
An error occurs during workflow control or coordination.

Role resolution for determining an agent does not return a result which can be used.
A workflow system administrator can intervene and set workflows with errors to the status *in process* or the status *logically deleted* (possibly after eliminating the error). If a workflow is incorrect, the responsible workflow system administrator specified either globally in Customizing or in the basic data of each workflow definition is notified by mail.

This status does *not* mean that a dialog work item of this workflow has the *incorrect* status.

**Transition to completed status**

A workflow is set to the status *completed* when the last step of the relevant workflow has been completed.
Background Work Item (Type B)

Definition
Work item that represents a single-step task at runtime whose execution does not require a dialog and, therefore, can be controlled automatically by the system.

Integration
Type B work items are not displayed in the Business Workplace.
Status of a Background Work Item

The valid statuses for background work items (type B) are listed in the following table.

<table>
<thead>
<tr>
<th>Work item status</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>🔄 Waiting</td>
<td>The work item has been scheduled for its <em>requested start</em>. A work item has this status after it has been created until the <em>requested start</em> specified in the workflow definition is reached.</td>
</tr>
<tr>
<td>🔄 Ready</td>
<td>Temporary intermediate status of a background work item. The system calls the associated object method as soon as a background work item can be processed. The status of the background work item then changes to <em>in process</em> immediately.</td>
</tr>
</tbody>
</table>
| 🔄 In process     | The work item is currently being processed. A work item also has this status.  
• if the method was left with a temporary exception. In this case, special Error Handling for Background Work Items with Temporary Errors [Seite 1388] is carried out.  
• if the work item is waiting for its terminating event. The point at which processing is completed cannot be detected by the workflow system in this status. |
| 🔄 Completed      | The execution of the work item is completed. The result of the task represented by the work item is correct, i.e. the result modeled in the workflow definition. |
| 🔄 Logically deleted | Execution of the work item with regard to the process logic is no longer meaningful or necessary for the process to continue. A work item changes to the *logically deleted* status in the following way:  
• Termination in parallel processing branches  
  When the required number of processing paths has been executed in a fork, the work items in the other paths that have not yet reached the *completed* status are automatically set to the *logically deleted* status.  
• Intervention by an administrator  
  The administrator can only set a work item to the *logically deleted* status if it has not yet reached the *completed* status and is not part of a higher-level workflow.  
  A work item with the *logically deleted* status may have caused database changes or other actions (generate event, send notification). These changes are not canceled automatically (compensated). |
| 🔄 error          | Execution of the work item was terminated with an error. |

In addition to the statuses given above, a work item can be **locked against execution**. This is possible in any status and involves an administrative function which is selected for a work item via the Change option.
Error Handling for Background Work Items

Use

Error handling for background work items must be carried out by the workflow system:

- because background work items (with errors) are not displayed in the Business Workplace and are therefore detected too late, if at all.
- because background work items that are temporarily incorrect, for which another execution attempt can (theoretically) be successful, cannot be restarted by a user.

(Temporarily incorrect work items are work items whose method was terminated with a temporary exception for which no subsequent step has been defined.)

Features

In Customizing for the workflow system, you can define how often the workflow system attempts to restart a temporarily incorrect work item. You can also define the interval between two repetition attempts and activate the monitoring function.

Monitoring is also activated as part of automatic Customizing [Extern] (Tools → Business Workflow → Development → Utilities → Customizing).

You can also specify the maximum number of repetition attempts (“repetition counter”) separately for each individual background step in the respective workflow definition. This setting overrides the value in Customizing if it is greater than zero.

If an error (method exception) occurs when background work items are executed by the workflow system, the system responds as follows depending on the type of exception and error modeling settings:

<table>
<thead>
<tr>
<th>In the workflow definition...</th>
<th>The exception is defined for the object method as a temporary error system or application error</th>
</tr>
</thead>
<tbody>
<tr>
<td>a subsequent step is modeled for the exception.</td>
<td>The step has status completed and the modeled subsequent step is executed. The step has status completed and the modeled subsequent step is executed.</td>
</tr>
<tr>
<td>Background work items for which a repetition counter is explicitly modeled in the workflow definition are restarted first by the system. If all of the attempts are unsuccessful, the status of the work item changes to completed and the modeled subsequent step is executed.</td>
<td>The modeled subsequent step is executed.</td>
</tr>
</tbody>
</table>
Error Handling for Background Work Items

| no subsequent step is modeled for the exception | The step is not yet completed. The respective work item retains the status in process. Background work items are restarted by the system. The number of repetition attempts is determined either by the repetition counter in the step definition or - if this is equal to zero - by the repetition counter set in Customizing. If all of the attempts are unsuccessful, the work item status changes to incorrect. | Workflow and work item assume the error status. |

**Processing Incorrect Work Items**

The workflow system sends a mail to the relevant system administrator for every background work item with the status incorrect.

**Processing Work Items That Have Been Started**

The workflow system determines all of the background work items that have been in process for longer than 30 minutes. An error message is then sent to the workflow system administrator for all of these background work items, since the system assumes that processing has been cancelled. However, this does not necessarily mean that an error has occurred.
Wait Step Work Item (Type E)

**Definition**
Work item that represents a wait step or a workflow at runtime, which is waiting for an event to occur.

**Use**
Type E work items are not displayed in the Business Workplace.
Work Item with Express Notification

Use
When the system creates a work item with priority 1, each of its recipients [Extern] receives an express notification (dialog box with appropriate text) on the screen. The recipient can call the Business Workplace directly from the express message. Excluded agents do not receive a message.

Constraints
The system does not create an express message
- if the user processes the work item immediately due to advance with immediate dialog [Seite 1453].
- if the work item represents a general task [Extern] that is not restricted to certain agents.
- if the work item was forwarded.

The system only creates express notifications when a work item is created. An express notification is not created for the new recipients of priority 1 work items that are forwarded.

If one of the selected agents processes the work item, the other agents still receive an express message.

Features
Express messages are only sent for dialog steps as soon as the system has created them with the ready status. If the work item is created first with the waiting status because its requested start has not been reached yet, the express message is not sent until the status changes from waiting to ready.

Activities
How is the priority set?
The priority of a work item can be determined for steps that require a dialog with the user. It is defined in the tab page Miscellaneous of the step definition.
Workflow Settings

Use
You use the workflow settings to maintain the special workflow functions in the Business Workplace.

Features
The following functions are available:

- Personal settings [Seite 1395]
- Display organizational assignment [Seite 1397]
- Refresh organizational environment [Seite 1397]
- Adopt substitution [Seite 1400]
- End substitution [Seite 1400]
- Maintain substitute [Seite 1400]
- Activate substitute [Seite 1400]
- Adopt view [Seite 1404]
- Exit view [Seite 1404]

Activities
You can access the workflow settings within the Business Workplace [Seite 1368] via Settings → Workflow settings.
**Personal Settings**

**Use**
The personal settings for workflow enable you to adapt the runtime system to suit your requirements.

**Features**

**Work item display**

<table>
<thead>
<tr>
<th>Setting</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>User view with ActiveX (32-bit)</td>
<td>Both of these settings define the [work item display][Seite 1411] as the default. Note that the ActiveX variant is only available on 32-bit platforms.</td>
</tr>
<tr>
<td>User view without ActiveX</td>
<td>This setting defines the [technical work item display][Seite 1418] as the default.</td>
</tr>
<tr>
<td>Technical view</td>
<td>This setting defines the technical work item display [Seite 1418] as the default.</td>
</tr>
</tbody>
</table>

**Workflow log**

<table>
<thead>
<tr>
<th>Setting</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>User view with ActiveX (32-bit)</td>
<td>User view of the [workflow log][Seite 1420] without any technical aspects. This view uses ActiveX controls and is therefore only available on 32-bit platforms.</td>
</tr>
<tr>
<td>User view without ActiveX</td>
<td>User view of the workflow log without any technical aspects. This view uses the hierarchical list display of the ABAP List Viewer [Extern].</td>
</tr>
<tr>
<td>Technical view</td>
<td>In addition to the main semantic information, the technical view of the workflow log also displays technical numbers and texts that may not be available in the logon language of the user. The technical view is intended primarily for system administrators.</td>
</tr>
</tbody>
</table>

**Further settings**

<table>
<thead>
<tr>
<th>Setting</th>
<th>Description</th>
</tr>
</thead>
</table>
| Display work item texts in logon language    | All work item texts in the Business Workplace are always displayed in the user's logon language.  
If the user's logon language is different to the language in which the work item was created, additional database accesses are required that can delay display of the Business Workplace. |
| Enable forwarding of work items to several users | The work item can be forwarded to several agents.  
The term “agent” is used in a broad sense in this context. You specify the agent as an organizational object (organizational unit, job, position, user) when forwarding the work item. This organizational object can consist of several persons.  
For further information, refer to [Forwarding][Seite 1435]. |
| (Double)-clicking on an object displays the object in the same window. | You can choose an object in the work item display, which is then displayed in the current window. |
### Personal Settings

<table>
<thead>
<tr>
<th>(Double)-clicking on an object displays the object in the same window.</th>
<th>You can choose an object in the work item display, which is then displayed in the current session window and replaces the work item display. When you exit the object display, you return to the work item display. If you do not set the indicator, a new session is created to display the object.</th>
</tr>
</thead>
<tbody>
<tr>
<td>This setting does not apply if you work with the SAP GUI for HTML.</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>No tips &amp; tricks in workplace</th>
<th>The Business Workplace [Seite 1368] includes tips and tricks for working with work items and workflows in the work item preview. If you set this indicator, the tips and tricks are not displayed.</th>
</tr>
</thead>
</table>

<table>
<thead>
<tr>
<th>No HTML in execution of decision tasks.</th>
<th>You use this indicator to decide whether the execution of decision tasks is displayed in HTML or not. Deactivating the HTML display may make sense, for example, if you have problems displaying HTML documents generally because of the settings in your local HTML browser.</th>
</tr>
</thead>
</table>

### Activities

To change the personal settings for workflow, you choose Settings → Workflow settings → Personal settings in the Business Workplace.

The settings are saved as user parameters and take effect the next time the work item display or workflow log is called.

You can change the standard values for these personal workflow settings in Customizing for SAP Business Workflow. These standard values are always used if users have not maintained settings of their own.
Organizational Assignment of a User

Use
The worklist displayed is formatted specifically for the user who is logged on. You can query this user's organizational assignment.

Prerequisites
The functionality described depends on a setting in the PD system table [Seite 1533].

Features
The organizational assignment of a user is buffered and read at each new logon or once a day (but not each time the Business Workplace is called). If the organizational assignment of a user changes while the user is logged on and must be updated, you choose Settings → Workflow settings → Refresh organizational environment in the Business Workplace.

Activities
To view current information on the assignment of the user logged on in the organizational plan of the enterprise, choose Settings → Workflow settings → Display organizational assignment in the Business Workplace.

The following information is displayed for the user logged on:
- The name maintained in the user address
- The organizational unit
- The position the user occupies
- The job describing the position

By double-clicking on an entry, you can display the staff assignments or the job description.
Substitutes for Workflows

Use

*SAP Business Workflow* implements a substitution arrangement to deal with a user's duties in their absence.

Prerequisites

The automatic appearance of work items in the substitute's worklist is dependent on a setting in the PD system table [Seite 1533]. It is possible to define which objects can be entered as position-related substitutes in Administration of Organizational Management [Seite 1530].

Features

Who is involved?

Substitution always involves two users:

- One user specifies a substitute: User B
- The other user (the substitute) 'adopts' the substitution: User A

The substitute can process “foreign” work items for the duration of the substitution. It is irrelevant whether they are assigned to the underlying single-step tasks as a possible agent.

How does substitution work?

Substitution works in two ways:

1. User A adopts substitution for user B and for the duration of the substitution sees only the work items seen previously by B in their workflow inbox.

   User B must have entered user A as their substitute for a particular period without activating the substitution. During this period, A can adopt substitution for B at any time without having to confer with B.

   Instead of their own worklist, user A is displayed user B's worklist. B's name is in the column Substitution for in the worklist, as long as this column is included in the current configuration of B's Business Workplace.

2. User A automatically sees not only the work items previously seen by employee B, but also their own work items in their Business Workplace. This also applies to all work items generated for B in the future ("automatic forwarding").

   User B must enter user A as substitute and activate the substitution. No further action on the part of user A is required for this kind of substitution.

   A can continue working as usual. They see a Business Workplace to which B's work items are added dynamically. A can recognize these work items by the fact that B's name is in the column Substitution for, as long as this column is included in the current configuration.

In both cases, user A can perform operations on these "foreign" work items within the scope of the substitute profile assigned to them by B. It is irrelevant whether A is assigned to the underlying single-step tasks as a possible agent.

User B can continue working without any restrictions in both cases.
Activities
You can access substitute maintenance in the Business Workplace [Seite 1368] by choosing Settings → Workflow settings. For information on the individual functions, refer to Maintaining and Activating Substitutes [Seite 1400]
Maintaining and Activating Substitutes

Maintaining substitutes

1. In the Business Workplace [Seite 1368], choose Settings → Workflow settings → Maintain substitute.

   The Maintain substitute dialog box appears.

   To facilitate maintenance of the substitution, the Personal substitutes entry and the positions you occupy are displayed in the hierarchy on the Maintain substitutes screen.

   You specify your substitute either as a personal substitute or as a position-related substitute.

   - **Personal substitute**
     
     A personal substitute can see and execute all your work items, including those assigned to you via a personal agent assignment.
     
     You can only specify one other user as a personal substitute.

   - **Position-related substitute**
     
     A position-related substitute can only see and execute the work items you have received on the basis of agent assignment at the level of position, job or organizational unit.
     
     You can enter another position or a user as a position-related substitute.

2. Position the cursor either on the entry Personal substitutes or on the relevant position, and select Create substitute.

   You can maintain existing entries by double-clicking on the relevant entry.

   If you want to maintain a position-related substitution, decide whether you want another position or another user as a substitute. Select either position or user as the substitute type and specify the number or the user name of the substitute.

3. Specify the validity period for the substitution on the detail screen for substitution.

   Only within this period can the substitute adopt the substitution.

4. Specify a substitute profile in the dialog box Detail screen substitution.

   Irrespective of whether you have created your substitute as a personal or position-related substitute, you can limit the scope of the work items displayed to your substitute by specifying a substitute profile.

5. Select the field Substitution active if applicable.

   You must activate the substitution if you want work items to be visible for the substitute automatically from now on. If you do not activate the substitution, the substitute must adopt the substitution explicitly to see your work items.

6. Save your entries and exit substitute maintenance.
Activating substitutes

1. In the Business Workplace, choose Settings → Workflow settings → Activate substitute.
   The Activate substitute dialog box appears.
2. Select the substitutes that you want to activate, and choose the function Activate.

Deactivating substitutes

1. In the Business Workplace, choose Settings → Workflow settings → Activate substitute.
   The Activate substitute dialog box appears.
2. Select the substitutes that you want to deactivate, and choose the function Deactivate.

Adopting substitution

1. In the Business Workplace, choose Settings → Workflow settings → Adopt substitution.
   The Choose substitution dialog box appears.
2. Select the user(s) for which you want to adopt substitution.
3. Exit the dialog box.

Ending substitution

1. In the Business Workplace, choose Settings → Workflow settings → End substitution.
   The substitution is ended.
Views

Use
The Business Workplace [Seite 1368] provides various views on the work items displayed in the workflow inbox.

Using an appropriate view, a superior can "see" and process the work items of their employees.

Choosing a particular view also gives you the opportunity to see work items of other users in your workflow inbox and process them with full functionality, although the underlying tasks are not organizationally assigned to you.

Prerequisites
Views only require action by an employee: This employee chooses a view from a catalog of defined views. The employee must have the authorization required to choose a particular view.

Authorizations
To choose a particular view, you require a corresponding authorization based on authorization object S_WF_LVIEW.
To maintain a particular view, you must have the relevant authorization. This is an authorization based on the authorization object S_TABU_DIS to maintain tables for authorization group SWES.

Features
Views are always based on an evaluation path [Seite 1534] starting from the employee who wishes to adopt a view and leading to the employees whose inboxes could be viewed. The employee adopts a view by selecting another employee from the result list of the evaluation.

Activities
The activities associated with this function include:

- Adopting and Exiting Views [Seite 1404]
- Maintaining Views [Seite 1403]
Maintaining Views

Prerequisites
A view [Seite 1402] is always based on an evaluation path [Seite 1534]. This evaluation path describes which relationships are traced from the user who wants to adopt the view to the users whose Business Workplaces can be viewed.
You can use one of the evaluation paths available in the system. If there are no suitable evaluation paths, you can define an evaluation path [Seite 1535] of your own.

⚠️
Different views can only be maintained as a customer setting.

Procedure
1. To call table maintenance for views, choose Tools → Business Workflow → Development → Definition tools → Worklist client → Maintain views.
   This displays the screen Change View "View for Maintaining Views": Overview".

   Maintaining views is an activity that is described in the Implementation Guide and can be performed in Customizing.

2. Create a new view. To do this, choose the function New entries.
3. Assign a unique name to the view.
4. Specify an evaluation path.
5. Describe the view with a long text.
6. Specify a start evaluation path.
   This selection is optional. The start evaluation path is used to get an initial selection of objects, which is then evaluated further via the first evaluation path.
7. Select Save.
Adopting and Exiting Views

Procedure

Adopting views

1. To adopt a view as the standard view (view on your own work items), choose Settings → Workflow settings → Adopt view in the Business Workplace.

   You are now on the dialog box View: Choose Agent. Only the views for which you have an authorization are available.

2. Choose a view.

   On the basis of the evaluation path defined for the view, the system selects the positions, organizational units or users connected to you.

3. From the result list of this selection, choose the object whose workflow inbox you want to view.

Exiting views

A view is only active while the Business Workplace is displayed. The next time you call the Business Workplace, you are asked if you wish to adopt the view previously set.

To return to the standard view when working in the workflow inbox of the Business Workplace, choose Settings → Workflow settings → Exit view.
Dynamic Columns for the Business Workplace

Use
Up to 6 columns in the workflow inbox of the Business Workplace [Seite 1368] can be filled on a task-specific basis with contents that are determined dynamically at runtime. The standard functions for filtering, sorting, and grouping are available for these columns. Please compare with Selectable Columns for the Business Workplace [Seite 1406].

Features
If you want to include one of the “dynamic columns” into the workflow inbox display, you must specify an element from the task container for each task, from which the content of the column is established at runtime.

Work items that belong to different tasks are then also displayed with different information. Work items that belong to tasks for which this functionality is not used are displayed with a blank entry.

Activities
The column contents are maintained via Tools → Business Workflow → Development → Definition tools → Worklist client → Dynamic columns for worklist.

You can also define the column headings. These headings are displayed if all of the work items displayed in the Business Workplace refer to the same task.
Selectable Columns for the Business Workplace: Workflow

The columns displayed essentially determine the appearance and information content of the workflow inbox of the Business Workplace. Detailed knowledge of the columns is also required to make full use of the filter and grouping criteria.

You can determine the selection of columns via display variants. The following columns are available:

<table>
<thead>
<tr>
<th>Column</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>Work item is executable</td>
<td>Indicator denoting that the work item is executable. (เครื่องหมาย)</td>
</tr>
<tr>
<td>Work item title</td>
<td>Title of work item.</td>
</tr>
<tr>
<td>Status indicator</td>
<td>Status of work item. In symbol form.</td>
</tr>
<tr>
<td>Creation date</td>
<td>Date when the work item was created with the status ready or waiting for the first time. A work item is only created with status waiting if a requested start was declared for the work item and the work item is created before the requested start.</td>
</tr>
<tr>
<td>Creation time</td>
<td>Creation time of a work item.</td>
</tr>
<tr>
<td>Priority</td>
<td>Priority of the work item.</td>
</tr>
<tr>
<td>Attachments exist</td>
<td>Attachments are shown with the symbol.</td>
</tr>
<tr>
<td>End of processing must be confirmed</td>
<td>Indicator denoting that the end of processing must be confirmed explicitly. (เครื่องหมาย)</td>
</tr>
<tr>
<td>Work item overdue</td>
<td>Indicator denoting that a deadline has been missed for the work item. (เครื่องหมาย)</td>
</tr>
<tr>
<td>ID</td>
<td>Unique number of a work item, which is assigned internally by the system.</td>
</tr>
<tr>
<td>Type</td>
<td>Type of work item. In abbreviation form.</td>
</tr>
<tr>
<td>Task</td>
<td>Identification for the task represented by the work item (for example TS00008323).</td>
</tr>
<tr>
<td>Technical status</td>
<td>Status of work item. Technical name.</td>
</tr>
<tr>
<td>Work item type</td>
<td>Type of work item. Name</td>
</tr>
<tr>
<td>Task name</td>
<td>Name of task</td>
</tr>
<tr>
<td>Work item status text</td>
<td>Status of work item. Name</td>
</tr>
<tr>
<td>Deadline status</td>
<td>The deadline status specifies whether one of the deadlines has been missed. The possible values in this column are therefore:</td>
</tr>
<tr>
<td></td>
<td>- None</td>
</tr>
<tr>
<td></td>
<td>- Latest start</td>
</tr>
<tr>
<td></td>
<td>- Requested end</td>
</tr>
<tr>
<td></td>
<td>- Latest end</td>
</tr>
<tr>
<td></td>
<td>- Other</td>
</tr>
<tr>
<td>Current agent</td>
<td>Name of the user who last reserved or processed the work item.</td>
</tr>
<tr>
<td>Field</td>
<td>Description</td>
</tr>
<tr>
<td>------------------------</td>
<td>-----------------------------------------------------------------------------</td>
</tr>
<tr>
<td><strong>Latest end date</strong></td>
<td>Latest end of work item. The end is reached when the work item assumes the</td>
</tr>
<tr>
<td></td>
<td>status <em>completed</em>.</td>
</tr>
<tr>
<td><strong>Latest end time</strong></td>
<td>Latest end time of a work item</td>
</tr>
<tr>
<td><strong>Forwarder</strong></td>
<td>Name of the party who forwarded the work item.</td>
</tr>
<tr>
<td><strong>Substitution for</strong></td>
<td>Name of the substituted party whose work item is displayed</td>
</tr>
<tr>
<td><strong>Work item content</strong></td>
<td>Column in which the default attribute of the object referenced in the</td>
</tr>
<tr>
<td></td>
<td>container element _W1_Object_ID is displayed.</td>
</tr>
<tr>
<td><strong>Group object</strong></td>
<td>Column in which the default attribute of the object referenced in the</td>
</tr>
<tr>
<td></td>
<td>container element _W1_Group_ID is displayed.</td>
</tr>
<tr>
<td><strong>Execution can be rejected</strong></td>
<td>Indicator denoting whether execution of the work item can be rejected (x)</td>
</tr>
<tr>
<td><strong>Dynamic columns</strong></td>
<td>Refer to Dynamic Columns for the Business Workplace [Seite 1405].</td>
</tr>
</tbody>
</table>
Workflow Inbox

Use

The worklist of the user currently logged on to the Business Workplace is displayed in the workflow inbox.

Integration

As is the case for the workflow resubmissions [Seite 1443] and the workflow outbox [Seite 1441], the workflow inbox is an integral part of the Business Workplace.

Features

Views in the workflow inbox

A user's worklist can be displayed as an overview or according to the following grouping criteria:

<table>
<thead>
<tr>
<th>Grouping Criteria</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Grouped according to task</td>
<td>The work items are grouped according to the tasks to which they belong.</td>
</tr>
<tr>
<td>Grouped according to content</td>
<td>The work items are grouped according to the object instances to which they belong.</td>
</tr>
<tr>
<td>Grouped according to content type</td>
<td>The work items are grouped according to the object types to which they belong.</td>
</tr>
<tr>
<td>Grouped according to sort key</td>
<td>The work items are grouped according to sort keys. Please refer to Grouping According to Sort Keys [Seite 1432].</td>
</tr>
</tbody>
</table>

You can also choose from the following views:

- Overdue entries
- Deadline messages
- Incorrect entries

Workflow Functions

The screen area in which the worklist is displayed has a toolbar containing the workflow functions and functions for configuring the list. For information about these functions, refer to the documentation on the ABAP List Viewer [Extern].

The workflow functions can be accessed in the Business Workplace via the toolbar or the relevant context menu (right-hand mouse button). The following functions can be executed on work items:

- Update
  This function updates the worklist [Extern] of the user currently logged on to the Business Workplace.

- Execute
  To be executed, a work item must have either the status ready or the status in process.
  When a dialog work item [Extern] is executed, the object method to which the single-step task for this work item refers is carried out.
  When a missed deadline work item [Extern] is executed, the most important information on the monitored work item is displayed.

- Display work item
  This function goes directly to the work item display [Seite 1411]
• **Reserve** (dialog work items only)
  This reserves a work item for execution by the end user in question. The work item must have the status *ready*. This work item is then no longer visible to the other recipients who could previously see it in status *ready*. The status of the work item changes from *ready* to *reserved*.

• **Replace** (dialog work items only)
  Replacing releases the work item from its reserved status and makes it available again to all of the original recipients. The status of the work item changes from *reserved* back to *ready*. The status of the work item must be *reserved* before it can be replaced.

• **Forward**
  Forwarding [Seite 1435] passes a work item on to another employee for execution.

• **Resubmit**
  If a user chooses this function, the selected work item is placed in workflow resubmissions [Seite 1443].

• **Display workflow log**
  This function displays the workflow log [Seite 1420].

• **Manage attachments** ... (Functions for attachment management [Seite 1429], dialog work items only)
  - Display attachments
  - Create attachments
  - Change attachments
  - Delete attachments

• **More functions**
  - Set to 'Done' [Seite 1440]
  - Reject execution [Seite 1430]
  - Execute together [Seite 1431]
  - Change priority [Seite 1433]
  - Send mail [Seite 1425]
  - Change work item [Seite 1438]

• **Environment**
  - Display objects in workflow (dialog work items only)
  - Display workflow relationships [Seite 1436]
  - Start Workflow [Seite 1450]
  - Create link [Seite 1434]
Activities
You can access the workflow inbox in the tree on the left in the Business Workplace via *Workplace → Inbox*. 
Work Item Display: Standard View

Use
The objective of the work item display is to display all the information and activities that are relevant to an end user in the environment of the displayed work item in a clear and concise manner. The work item display supports the agent not only in the processing of the current work item but also in the compilation of an activity list, which then functions as the agent's personal worklist.

Integration
A technical work item display [Seite 1418] is available for work items of all other types. You can make this technical work item display standard for dialog work items as well. The standard view of the work item display can be shown with or without ActiveX. You make this setting for the work item display variant in your personal workflow settings [Seite 1395].

Features
The work item display is primarily designed for displaying dialog work items [Extern]. It contains details about deadlines, statuses, agents, attachments and linked objects for a work item. The work item display also enables an end user - providing they have the relevant authorization - to compile an activity list as their personal worklist. The work item display has three tab pages (Basic data, Activities, and Available objects).

- Tab page Basic data [Seite 1414]
- Tab page Activities [Seite 1415]
- Tab page Available objects [Seite 1416]
- Customer-defined tab page for work item display [Extern]
  You can define another tab page, which is then displayed as the first tab page when the work item display is called.

Application toolbar functions

- Execute
  To be executed, a work item must have either the status ready or the status in process.
  When a dialog work item is executed, the object method to which the single-step task for this work item refers is carried out.
  When missed deadline work items [Extern] are executed, the most important information on the monitored work item is displayed.

- Display last message
  The return code that was returned to the workflow system after the object method was executed can be retrieved for processed work items using the Messages function.

- Forward
  Forwarding [Seite 1435] passes a work item on to another employee for execution.

- Resubmit
If a user chooses this function, the selected work item is placed in workflow resubmissions [Seite 1443].

- **Change priority**
  Refer to Changing Priorities [Seite 1433].

- **Change deadlines**
  Refer to Changing Deadlines [Seite 1428].

- **Display/create/change attachments**
  For attachments, refer to Attachment Management [Seite 1429].

- **Reserve** (dialog work items only)
  This reserves a work item for execution by the end user in question. The work item must have the status *ready*. This work item is then no longer visible to the other recipients who could previously see it in status *ready*. The status of the work item changes from *ready* to *reserved*.

- **Replace** (dialog work items only)
  Replacing releases the work item from its reserved status and makes it available again to all of the original recipients. The status of the work item changes from *reserved* back to *ready*. The status of the work item must be *reserved* before it can be replaced.

- **Mail**
  Refer to Send Mail [Seite 1425].

- **Display workflow log**
  This function displays the workflow log [Seite 1420].

- **Display graphical workflow log**
  This function displays the graphical workflow log [Seite 1424].

**Additional functions in the menus**

**Work Item**

- **End resubmission**
  The work item is put back into the workflow inbox. It has the status *reserved*.

- **Create link** [Seite 1434]
- **Reject execution** [Seite 1430]
- **Set to ‘Done’** [Seite 1440]

**Goto**

- **Methods**
  You can use this function to execute the defined secondary methods [Extern] of the work item displayed from the work item display.
  
  A prerequisite for this is that the work item represents an activity that is part of a workflow. At least one secondary method must be defined for this activity.
Work Item Display: Standard View

- **Workflow description**
  The description text of the superordinate multistep task, the “process description”, is displayed.
  The work item must be part of a workflow and a description text must be maintained for the workflow.

- **Technical work item display [Seite 1418]**

**Extras**

- **Displaying tasks**
  This function can be used to display the definition of the task represented by the work item. Refer to Definition of a Single-Step Task [Seite 1175] and Definition of a Multistep Task [Seite 1194].

- **Technical data**
  Technical data about the work item, for example its ID, its texts or the ID of the superordinate work items are displayed.

- **Organizational Assignment [Seite 1397]**
- **Displaying Agents [Seite 1427]**

**Activities**

You can access the work item display by:

- Selecting a work item in the Business Workplace and choosing 📝.
- Double-clicking one of the entries for a step in the workflow log.
- Selecting an entry that does not represent a workflow in the work item selection [Seite 1490] hit list displayed. (If you choose a workflow, the workflow log is displayed.)
Tab Page Basic Data

Use
The information on this tab page of the work item display [Seite 1411] is mostly self-explanatory. Except for the priority, you cannot make any changes here.

Features

Deadlines
These are the deadlines monitored by the runtime system. (Deadlines that are not set are displayed without a date.)
- Start by (latest start)
- End by (latest end)
Depending on whether the work item represents a step in the workflow or a task, the deadlines were either specified when this step was defined in the workflow definition or when the task was started online.
To display all of the deadlines of the work item, choose Work item → Deadlines.

Further Information
- Forwarded by
  If the work item was forwarded to you, the name of the forwarder is entered here.

- Priority
  The priority of a work item is derived from the definition of this step in the workflow definition. The priority is used as a sort criterion for positioning the work item in the Business Workplace.
  The priority can be changed here.

- Status
  The current processing status is expressed by the work item status [Seite 1373].

- Creation date (created on) and processing date (processed from)
  These are the actual dates and times (when the work item was created and when processing was started).
Using the Messages function, you can display the return codes for processed work items, which were returned to the workflow system after execution of the object method.

Work Item Description
A description of the work item to be executed is provided at the bottom left of this tab page.
The task description is entered in the task definition. It is used for information purposes and generally contains instructions and recommendations on processing the work item displayed.

Attachments
The titles of all the attachments added to this work item or, if the work item is part of a workflow, to the preceding work items are shown in the lower right of this tab page.
Tab Page Activities

Use
The Activities (not yet processed) list contains all the activities that are relevant for processing this work item.
The tab page Activities is part of the work item display [Seite 1411].

Features
The work item text of the task represented by the work item is generally at the start of the list ("main activity"). Once this activity has been processed (and completion of processing has been confirmed, if necessary), the status of the work item changes to completed. No other actions can then be carried out in the work item display.

Activities
You can extend the activity list and in this way create a worklist. The activities added represent your "personal worklist" as end user (agent).

<table>
<thead>
<tr>
<th>Function</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>✔️ Done</td>
<td>An activity selected from the list is reported to be done. This functionality cannot be executed on the leading activity (first line) of this work item. The agent is responsible for reporting that an activity is done. The actual processing is not checked.</td>
</tr>
<tr>
<td>![Execute]</td>
<td>An activity selected from the list is executed. The main activity (first line) of the work item can also be executed via the menu path Work item → Execute.</td>
</tr>
<tr>
<td>🗂️ Create</td>
<td>Another activity is added to the list. This activity can entail</td>
</tr>
<tr>
<td></td>
<td>• Executing a method on an existing object (created invoice, created material, etc.)</td>
</tr>
<tr>
<td></td>
<td>• Creating a new object (text, etc.)</td>
</tr>
<tr>
<td></td>
<td>When you create an activity, you are given step-by-step support in dialog by a &quot;wizard&quot;.</td>
</tr>
<tr>
<td>![Delete]</td>
<td>An activity is deleted from the list.</td>
</tr>
</tbody>
</table>

Scrolling in the activity list: The activity list contains extensive information on each activity, which you can view by scrolling to the right. This information includes the following:

- Description
- Creator of the activity with date and time
- Actual agent of the activity with date and time
- Method and object type
Tab Page Available Objects

Use
All objects that are related to the work item are displayed in the list of available objects. These objects are stored in container elements of the task container as object references. You can display these objects or add new objects to the container element. There are:

- **Ad hoc objects**
  - Objects added to a work item in this or one of the preceding steps of the workflow (container element _Adhoc_Objects of the task container)

- **Attachment objects**
  - Documents added to a work item in this or one of the preceding steps of the workflow (container element _Attach_Objects of the task container).

- **Process objects**
  - The object currently being processed (container element _WI_Object_ID of the task container)
  - The object added for grouping purposes (container element _WI_Group_ID of the task container)
  - Objects that are referenced in other elements of the task container

The tab page Available objects is part of the work item display [Seite 1411].

Features

Displaying objects with their default attributes
Each object referenced in the work item container is displayed with its default attribute [Extern]. The default method [Extern] of each object can be executed upon request. If no default attribute was defined for the object type, the key fields of the object are displayed.

Adding objects
You can extend and process the list of objects. The main purpose of this is to make the relevant information available to the agents of the subsequent steps in the workflow as well. Only object types [Extern] that support the IFFIND interface can be selected. You identify an actual object [Extern] of this type by specifying its key fields [Extern].

Activities
To execute the functions displayed, proceed as follows:

<table>
<thead>
<tr>
<th>Function</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Create</td>
<td>An object is added to the list. When you add an object, you are given step-by-step support in dialog by a &quot;wizard&quot;.</td>
</tr>
<tr>
<td>Display</td>
<td>The default method of an object selected from the list is executed.</td>
</tr>
<tr>
<td>Remove</td>
<td>An object is deleted from the list.</td>
</tr>
</tbody>
</table>
Work Item Display: Technical View

Use
The technical work item display shows all information from the environment of the work item displayed. The technical work item display is intended in particular for workflow system administrators.

The data is always formatted and displayed in a similar way irrespective of the work item type. You should note the work item type shown on the screen.

Features

General

Work item information
- Information derived from the definition of the relevant task: Work item type, work item text, way in which processing is completed.
- Status of the work item as current processing information
- Actual agent (available after dialog work items have been executed).

In the case of processed work items, the Messages function can be used to display the return codes returned to the workflow system after execution of the object method.

Deadline data
Here you can find the current dates/times [Extern] (creation date/time of work item and date/time processing started and ended) as well as the deadlines that are monitored by the runtime system (requested and latest start and end deadlines).

A monitored deadline shown with red is in the past. If the symbol is displayed as well, a defined action was triggered.

Description
This description displayed here is taken from the task definition. It is used to inform the selected agents and contains instructions and recommendations on processing the work item displayed.

If you have selected a work item using the work item selection [Seite 1490] or the workflow outbox rather than your workflow inbox, you can execute it provided you are one of the possible agents.

You do not need to be one of the recipients in this case.

Additional functions
As well as the functions available in the standard view of the work item display [Seite 1411], the following additional functions are also available:
- Display/create/delete object

Each object referenced in the work item container is displayed with its default attribute [Extern]. The default method [Extern] of each object can be executed upon request. If no default attribute is defined for the object type, the key fields of the object are displayed instead.
You can extend and process the list of objects. The main purpose of this is to make the relevant information available to the agents of the subsequent steps in the workflow as well.

Only object types [Extern] that support the IFFIND interface can be selected. You identify an actual object [Extern] of this type by specifying its key fields [Extern].

- **Type-specific data**
  Type-specific data only applies to work queue and wait step work items.
  For work queue work items (type A), the objects and tasks contained in the work queue are listed.
  For wait step work items (type E), the number of events expected (information taken from the workflow definition) and the number of events that have already occurred are specified.

- **Execute for testing**
  In order to make it possible to check the runtime system's program execution (work item manager and workflow manager) in debugging mode after execution of an object method, internal communication within the workflow system cannot take place asynchronously. To achieve this, execute the work item for test purposes. Enter /h in the command field first to go to the debugging mode.

- **Monitored work item**
  The work item (type F or W) whose deadlines or completion are monitored by the workflow system is displayed.
  This function is only possible and active from missed deadline work items [Extern]. This function displays the work item. Full display functionality, including all navigation and change options, is also available here for this work item.

- **Instance linkage**
  You go to the relevant line of the instance linkage table, in which the expected event (identified by object type and event ID) and the object (identified by object reference) are specified.
  This function is only possible and active for those work items that wait for an event that completes them. This may apply to dialog work items (type W) and background work items (type B). This does apply to wait step work items (type E).

- **Container**
  This function displays the content of the task container.

- **Change work item [Seite 1438]**

**Activities**
You can go to the technical view of the work item display by choosing **Goto → Technical work item display** in the work item display or by having this display variant as a presetting in your personal workflow settings [Seite 1395].
Workflow Log: Standard View

Use
The workflow log [Extern] enables you to format all the information that is generated or collected during the execution of one active business process in one central location. The standard view described here is intended, in particular, for agents and process controllers who want to get an overview of the steps processed so far.

Prerequisites
To use the view described here, you must have set a view with ActiveX in your personal workflow settings [Seite 1395].

Integration
Other options for displaying the workflow log are:
- Workflow Log: Technical View [Seite 1422]
- Graphical Workflow Log [Seite 1424]

Features

Application toolbar functions
The following functions are available on the application toolbar:
- Update
- List with technical details
  This function takes you to the technical view of the workflow log.
- Graphical workflow log
- Personal workflow settings
  You can use this function to maintain your personal workflow settings.

Tab pages
The system shows the various views on the workflow log on different tab pages.

Tab page Workflow Chronicle (what was processed when?)
The tab page Workflow Chronicle shows a hierarchical display of all steps in the workflow, which have been processed so far or are due to be processed. If the workflow has a subworkflow structure, the subworkflows are also displayed.
The Details function (symbol) lists the following information about each step in the lower part of the screen:
- Who carried out what detailed actions for these work items and with what results.
- When these actions were carried out.
- Which objects were involved.
The Agents function (symbol) displays the selected and possible agents of a step.
The Graphic function (symbol) displays the graphical workflow log.
**Tab page Workflow Agents ( documento, who processed what?)**
The tab page **Workflow Agents** shows the employees involved in this workflow up to now. The following is displayed for each employee:
- What action was carried out in what step.
- When this action was carried out.
- Which objects were involved.
This view shows how an employee was involved in the execution of a workflow.

**Tab page Workflow Objects ( documento, what was processed?)**
The tab page **Workflow Objects** lists the objects related to the workflow or addressed up to now in the execution of the workflow. These objects include:
- The “leading” object of the workflow.
- Any attachments and objects added in the individual steps of the workflow.
The following is displayed for each object:
- Who carried out what detailed action for what task.
- When this action was carried out.
This view shows what information was generated and processed, and how.

**Information at the click of a mouse**
You can view all the information provided in the workflow log using the mouse.
You can also go to the [work item display](#) for each dialog step. You can display address data for agents as well as the contents of work item attachments or the result of actions that have been executed.

**Activities**
You can access the workflow log from the work item display or the [Business Workplace](#) via the icon.
Workflow Log: Technical View

Use
The workflow log [Extern] enables you to format all the information that is generated or collected during the execution of one active business process in one central location.
The view described here, in particular the technical view described below, is intended for workflow system administrators.

Integration
Other options for displaying the workflow log are:
- Workflow Log: Standard View [Seite 1420]
- Graphical Workflow Log [Seite 1424]

Prerequisites
To use the view described here, you must have set the view without ActiveX in your personal workflow settings [Seite 1395].
If you have chosen technical view in your settings, the standard view with technical details is displayed (see below). Otherwise, the two workflow log views are identical.

Features
The system uses a two-level, hierarchical list from the ABAP List Viewer [Extern] to display the various workflow log views. You can adapt the appearance of the list to suit your requirements using display variants.
By single-clicking on an entry or a symbol, you can then branch to the workflow container or the work item display [Seite 1411], for example.
The following views are available:
- Workflow chronicle
  The first level of the Workflow chronicle view shows all the workflow steps that have already been processed or are currently pending. If the workflow has a subworkflow structure, the subworkflows are also displayed.
  The second level (detail view) shows the following for each step:
  - Who carried out what detailed actions for these work items and with what results.
  - When these actions were carried out.
  - Which objects were involved.
  This view is used to determine what activities were carried out in a workflow and in what order.
- Workflow agents
  The first level of the Workflow agents view shows the employees involved in this workflow up to now.
  The second level (detail view) shows the following for each employee:
  - What action was carried out in what step.
  - When this action was carried out.
Workflow Log: Technical View

- Which objects were involved.
  This view shows how an employee was involved in the execution of a workflow.

- **Workflow objects**
  The first level of the Workflow objects view lists the objects that are related to the workflow or that have been addressed up to now during execution of the workflow. These objects include:
  - The “leading” object of the workflow.
  - Attachments and objects added in the individual workflow steps.
  The second level shows the following for each object:
  - Who carried out what detailed action for what task.
  - When this action was carried out.
  This view shows what information was generated and processed, and how.

- **... With technical details (technical view)**
  The technical view shows technical control information for execution of a workflow, as required by workflow administrators, for instance.
  Based on the workflow chronicle, the technical view shows technical nodes and control structures, and makes additional data available, such as container elements (●), employee data (●), and workflow data (●).
  The status of the work item is also displayed.

- **... With subworkflow structure**
  Here you can choose whether or not to display any subworkflows and their structure.

- **... With error indicators**
  If you activate the function View with error indicators, errors are marked in the log with the symbol. The standard indicator is ●●●.

**Activities**
You can access the workflow log from the workflow inbox of the Business Workplace [Seite 1368] via the symbol, or using the context menu (click the right-hand mouse button).
You can maintain the individual views (such as the technical view) within the workflow log via Views or Views → Other views.
Graphical Workflow Log

Use
The workflow log [Extern] enables you to format all the information that is generated or collected during the execution of one active business process in one central location.

Integration
Other options for displaying the workflow log are:
- Workflow Log: Technical View [Seite 1422]
- Workflow Log: Standard View [Seite 1420]

Features
The graphical workflow log adds to the textual information. The workflow steps already processed are shown with 🖥 in a graphical representation of the workflow definition. You can see at a glance which “route” a workflow instance has taken and which activities are processed in parallel to your own within a business process. Unlike the text version of the workflow log, the graphical workflow log also shows the subsequent flow of a workflow instance.

The screen of the graphical workflow log is split into the workflow area (left) and the overview area (right).

The following functions are available:
- Refresh
  Refreshes the display.
- Display node
  If you select an executed step and choose this function, the system goes to the technical work item display.
  If the step has not yet been executed, the relevant activity is displayed.
- Align...
  The workflow is centered within the workflow area.
- Zoom in
  The size of the workflow is increased within the workflow area.
- Zoom out
  The size of the workflow is decreased within the workflow area.
- Complete view
  The whole of the workflow is shown within the workflow area.

For other functions, refer to the documentation on the Workflow Builder [Seite 1002].

Activities
You can call the graphical workflow log from within the workflow log [Seite 1420] via 🖥.
Send Mail

Purpose
You want to send a mail to another user in connection with a work item that requires processing. The work item being processed is therefore also to be made accessible to this user for information purposes. The Send mail function enables you to send mails to any recipients with a text which can be entered freely. These mails are automatically linked to your work item in such a way that when the mail is executed by the recipient the work item is displayed.

Process Flow

Procedure as sender of mail
1. In the Business Workplace's workflow inbox, position the cursor on a work item and choose Other functions → Send mail.
   If you are already in the work item display, choose Work item → Send mail.
2. Enter the text of the mail.
3. Send the mail to any recipients who are available as mail addressees
   The mail text is added to the work item as an attachment. The work item remains in your Business Workplace with the same status.
   You can still complete the work item despite the fact that you have sent a mail. Since this makes the mail which has been sent superfluous, the recipient of the mail receives a second mail informing him/her that the (first) mail no longer needs to be dealt with.

Reply ends resubmission
Optional: If you do not want to process the work item until the mail has been replied to, define a resubmission for the work item. To do this, position the cursor on the entry, choose Resubmit and enter a date in the future.
The work item is resubmitted irrespective of this date and appears in your workflow inbox when the reply to the mail is received.

Procedure as recipient of a mail
You receive a mail that can be executed.
1. Read the mail.
2. Execute the mail, if applicable. The work item referred to by the sender in their mail is displayed. For further information on the object to be processed, choose Goto → Object to display the default attribute or execute the default method (generally Display) for the processed object.
   You can execute this work item if you are one of the possible agents of this task.
3. Reply to the mail. To do this, choose Document → Reply or Document → Reply w/reference when the mail is displayed. Then enter your reply and save your entries.
4. Choose Document → Send. On the send screen, the work item to which the mail referred is entered as the "recipient". Do not change this entry. Send the reply to the proposed recipient.
Result

The reply sent ends a resubmission which may have been defined for the recipient work item. This work item is appended as an (additional) attachment and can be read as such by the original user and taken into account.
Display Agents

Use
This function displays information about the agents of the work item.

Features
The system displays the following agents for the work item:

- Recipient [Extern]
- Possible agents [Extern]
- Excluded agents [Extern]

You can choose a compressed display - containing only the user names (users only function) - or a complete display with additional information on the relationships used by the system to determine the agents (overall view function). You can also display the organizational assignment of each user.

Note that this information is only available for work items of types for which an agent is logical and necessary.

Activities
You can display the agents from the work item display [Seite 1411] by choosing Goto → Agents → ...
Changing Deadlines

Use
You use this function if you want to change the deadlines for a work item at workflow runtime. This dialog box displays all the deadlines of work item processing. Please also refer to Current Dates/Times of a Work Item [Extern].

Features
Depending on whether the work item represents a step in the workflow or a single-step task, the deadlines were either specified when this step was defined in the workflow definition or when the single-step task was started in dialog.

The dialog box is split into the following sections:

Deadlines
- Start by (latest start)
- End by (latest end)

Planned deadlines
- Start by (requested start)
- End by (requested end)

A monitored deadline shown in blue is in the past. If a checkmark is also displayed, the appropriate action has been initiated. This generally involves informing the deadline recipient.

Actual dates/times
- Created on: Creation date/time of the work item
  (Technically: The work item is created with the status ready or - if the requested start date has not yet been reached - with the status waiting).
- Processed from: Start of processing
  (Technically: Transition of the work item to the status in process).
- Completed on: End of processing or date when set to ‘done’).
  (Technically: Transition of the work item to the status completed.)

Activities
To execute this function, choose in the work item display [Seite 1411].
Attachment Management

Use
One or more attachments can be assigned to each work item that appears in the Business Workplace's workflow inbox.
Attachments are documents written either with a SAPscript editor (document classes RAW, SCR) or with a PC application (document classes DOC, URL, PPT, XLS, PDF, ...) and then imported. You can enter new documents as attachments or create attachments from existing files.

Features

General
The attachment is automatically
- Added to the work item container
- Added to the container of the superordinate workflow
- Added to the containers of the subsequent work items in the workflow

You can define default documents for the individual document classes. For further information, refer to Default Documents [Extern].

Attachments can be displayed by the recipients of the subsequent steps. But they cannot be changed and, therefore, have a document character.

A superior who is to make a decision on releasing a budget can enter an attachment justifying their decision. The selected agents of the subsequent steps can display this attachment.

If a work item has attachments, this is indicated by a symbol in the Attachments column in the Business Workplace. You can also execute the function for processing an attachment by double-clicking in this column (column header AT). If an attachment already exists, it is displayed.

If a work item has attachments, this is indicated by the symbol in the work item display.

Functions on attachments

<table>
<thead>
<tr>
<th>Display</th>
<th>Displays the attachment selected in the dialog box Existing attachments.</th>
</tr>
</thead>
<tbody>
<tr>
<td>Create</td>
<td>Creates a new attachment. If you want to import an existing document from your PC to the attachment, choose Import.</td>
</tr>
<tr>
<td>Change</td>
<td>Displays the selection of Changeable attachments.</td>
</tr>
<tr>
<td>Delete</td>
<td>Displays the selection of Deletable attachments.</td>
</tr>
</tbody>
</table>

Activities
You can access attachment maintenance in the Business Workplace by choosing or the relevant context menu (right-hand mouse button).
Reject Execution

Use
You can use this function if you need to reject execution of a work item for business or technical reasons. This function is only available for work items of type W.

The table entry to be processed already exists or the material whose master data is to be changed is no longer used.

Prerequisites
This function is only available if the property *processing rejectable* has been selected for the related activity in the workflow definition.

Features
Processing of the work item is terminated with the *reject execution* function. The subsequent steps defined in the workflow definition are executed.

Do not use this function if you do not want to or cannot process the work item for personal reasons (not responsible, not competent). In this case, replace the reserved work item or forward it.

Activities
You can access this function in the *Business Workplace* [Seite 1368] by choosing Other functions or the work item display (Work item → Other functions → Reject processing).
**Execute Together**

**Use**
You can select various work items and then execute them together. This function is only available for work items of type W.

The work items do not necessarily have to belong to the same workflow.

**Prerequisites**
The work items must refer to the same single-step task.
For information about other prerequisites for this function, in particular with regard to the underlying method, refer to [Creating an "Execute Together" Method](#).

**Features**
Only the first of the selected work items is proposed for execution. The entries you make here also apply to the other work items to be executed together.

**Activities**
Select the work items that you want to execute together. Then choose Execute together via Other functions... in the Business Workplace [Seite 1368].
Grouping According to Sort Key

Use
Each work item carries the two container elements _WI_Object_ID and _WI_Group_ID in its container. Both elements have been defined to hold an object reference.

- The container element _WI_Object_ID automatically contains the reference to the object to be processed in the work item.
- The container element _WI_Group_ID can contain an object reference, which must be assigned to this container element in a binding or via initial value assignment.

The object reference assigned via _WI_Group_ID is generally not identical to _WI_Object_ID, but is derived from the work item execution environment. It is used to group work items that refer to different objects or object types but are nevertheless connected.

Features

Display default attributes in workflow inbox
The default attributes [Extern] of the objects referenced in _WI_Object_ID and _WI_Group_ID are available in the Business Workplace's workflow inbox [Seite 1408] under the column headers Object or Group for grouping, sorting and filtering purposes.

Material master data is processed in several instances of a workflow. The container element _WI_Object_ID in the containers of the individual work items therefore always contains the reference to the object of type BUS1001 (material) to be processed.

The container element _WI_Group_ID is assigned the expression &Material.Labor& (laboratory/drawing office for material as object reference) in the relevant steps of the workflow definition. The default attribute of the laboratory is therefore available as a group.

Default methods
The default method [Extern] of the objects referenced in _WI_Object_ID and _WI_Group_ID is executed by double-clicking in the Object or Group column.
Change Priority

Use
The priority of a work item is a measure of its urgency. It can be used as a sort criterion for organizing the workflow inbox. This function is only available for dialog work items.

Features
End users can display and change the priority of a work item. The priority is between 1 (highest) and 9 (lowest).
The change in priority can also be passed on automatically to the superordinate workflow and then to all work items created subsequently. The Pass on priority to subsequent steps indicator must be selected for this to take effect.

If a higher priority (lower number) is defined in the workflow definition for one of the subsequent steps, it is not changed.

Activities
You can access the Change priority function in the Business Workplace by choosing Other functions...
Create Link

Use
By storing links to work items in folders [Extern], you can organize your work effectively using a personal folder hierarchy.

Features
The work item is added to a personal or shared folder as a link. This work item can be displayed from this folder and executed by its recipients. Work items can have an unlimited number of links.

Activities
You can access this function in the Business Workplace [Seite 1368] by choosing Environment...
Forward Work Item

Use
Forwarding passes a work item on to another user for execution.

Integration
The range of users to whom a work item can be forwarded is determined from the task definition as follows:

<table>
<thead>
<tr>
<th>Task definition:</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Task is a general task [Extern]</td>
<td>Work item may be forwarded to all users.</td>
</tr>
<tr>
<td>The indicator General forwarding allowed is set for the task</td>
<td>Work item may be forwarded to all users.</td>
</tr>
<tr>
<td>The indicator General forwarding not allowed is set for the task</td>
<td>Work item may only be forwarded to the recipients [Extern] of the task.</td>
</tr>
</tbody>
</table>

Features
The user has the following search options to select the new recipient:

- The F4 input help can be used to determine the new recipient by performing a generic search for name components.
- The structure search can be used to determine the new recipient using the graphical display of the organizational plan.

The new recipient does not have to be an actual user. (The work item can also be forwarded to an organizational unit or a job.)

Activities
You can access this function in the Business Workplace [Seite 1368] by choosing [ ].
Workflow Relationships

Use
You can use this function to establish the work items in which a particular object is being processed. The following work item types are taken into account:
- Dialog work items (type W)
- Background work items (type B)
- Workflow work items (type F)

The list displayed is purely runtime information.

Prerequisites

Authorization
You require appropriate authorization for the functions shown. This is an authorization based on the authorization object S_PROGRAM for executing a program for authorization group 'SWI_OA'.

Features
You can call the function for displaying linked workflows from several starting points.

Calling via SAP Easy Access
Two functions are available via SAP Easy Access:
- Workflows for object
  This function displays all workflows linked to a particular object (for example a specific notification of absence). You must specify this object beforehand.

- Workflows for object type
  This function displays all workflows linked to a particular object type (for example FORMABSENCE). You must specify this object type beforehand.

You can only specify objects whose object type definition supports the interface IFFIND.

Calling from the Business Workplace
In this case, you do not specify the object. The workflow relationships are displayed for the object processed in the selected work item.

Calling as a generic object service
In this case, you do not specify the object. The workflow relationships are displayed for the object being processed.

What is displayed?
The display of the workflow relationships has two parts. The upper part displays the workflows identified for the object or object type. The lower part displays data on the currently selected work item in the form of a simplified workflow log:
- Steps so far
The workflow steps processed so far are listed under Step name. The steps are linked to the work item preview of the relevant work item. The current agents of the work item are listed under Agents. Click once to go into detailed display of the user data.

- **Information objects addressed so far**

  All the objects and attachments belonging to the workflow are displayed here. Click once to display the information objects.

### Activities

<table>
<thead>
<tr>
<th>Calling...</th>
<th>Path</th>
</tr>
</thead>
<tbody>
<tr>
<td>... Via SAP Easy Access</td>
<td>Tools → Business Workflow → Development → Runtime Tools → Workflows For Object</td>
</tr>
<tr>
<td>... From the Business Workplace [Seite 1368]</td>
<td>Display workflow relationships</td>
</tr>
<tr>
<td>... As a generic object service</td>
<td>System → Workflow → Workflow overview</td>
</tr>
</tbody>
</table>
Change Work Item

Use
The function enables you to change information and data associated with the selected work item.

Features

Changes in the initial screen
In general, the changes that are allowed depend on the type and status of the work item. The following changes are possible:

- **Work item text**
- **Priority**
- **Deadline data**
  
  You can set and change deadline data for the start and end deadlines of the work item. You can only make specifications for deadlines that are still being monitored and are in the future. For example, dialog work items with the *ready* status can no longer be assigned a requested start.

Manual intervention into the processing of work items
If you are an administrator with the appropriate authorization, you can intervene manually into the processing and therefore into the process flow of a workflow. This is particularly helpful for dealing with errors. The following changes are possible:

- **Set to ‘ready’ manually**
  
  The work item status is changed from *waiting* to *ready*.
  
  The work item therefore appears in the workflow inboxes of the selected agents.

- **Complete manually**
  
  The work item status is changed to *completed*.
  
  In the case of dialog work items, this status change (together with the current work item container) is passed on to the workflow system for evaluation. If the object method to be executed is a synchronous method with a result, the possible result values are displayed for selection.

- **Replace manually**
  
  The status of the work item is reset to *ready*.
  
  The work item is therefore displayed again in the workflow inboxes of all selected agents.

- **Restart after error**
  
  Work items and, in particular, workflows with errors can be restarted after the errors have been corrected.
  
  The information written by the workflow system in a log when the error occurred is now taken into account.

- **Delete logically**
  
  The status of the work item is changed to *logically deleted*. 
In the case of dialog work items, this status change is passed on to the workflow system.

**Locking/unlocking execution**
An administrator can lock or unlock the current work item for execution. To do this, you select the appropriate function from the *Edit* menu.

**Deleting deadlines**
To delete a deadline you have changed or entered and reset it to its initial values, choose *Edit → Delete deadline*.

**Changing work item containers**
Choose *Edit → Change container* to go to the editor for changing the work item container. The contents of the container for the relevant work item are displayed. You can change the current, runtime-specific data for the particular work item. The container may still have elements that do not currently have a value. These elements are hidden as standard. The presence of elements that are not displayed is shown by an indicator. To display these elements, choose *Edit → Show elements*. To add more lines to container elements defined as multiline, choose *Edit → Additional line*.

**Activities**
You can access this function in the [Business Workplace](#) by choosing Other functions, the relevant context menu or the [technical work item display](#).
Set Work Item to Done

Use
This function is used by the recipient of a work item to confirm explicitly that processing of this work item has been completed. As long as explicit confirmation has not been provided, the work item has executed status and can be executed again or forwarded.

The status of the work item changes from executed → completed.

Prerequisites
This function is only possible on a work item if a setting was made in the definition of the associated single-step task stating that the end of processing must be confirmed explicitly.

Activities
You use this Business Workplace function if the status of the work item has been changed to executed by a terminating event.
In general, however, the work item is not set to done via the Business Workplace. This function is usually provided as a dialog box directly after a work item has been processed.
Missed deadline work items [Extern] must always be confirmed explicitly. After execution, they remain in the status in process until they are set to set to done.
You can access this function in the Business Workplace [Seite 1368] by choosing Other functions.
Workflow Outbox

Use
The workflow outbox lists the workflows started and the work items forwarded and executed by the current user.

Integration
As is the case for the workflow resubmissions [Seite 1443] and the workflow inbox [Seite 1408], the workflow outbox is an integral part of the Business Workplace.

Features
Views in the workflow outbox

Started workflows
This view shows work items for the tasks started by you as a user in dialog or by a triggering event whose event container contains your user name as _Evt_Creator.

Work items executed by me
The work items executed by you are displayed in this view.

Forwarded work items
The work items forwarded by you are displayed in this view.

Workflow Functions
The screen area in which the worklist is displayed has a toolbar containing the workflow functions and functions for configuring the list. For information about these functions, refer to the documentation on the ABAP List Viewer [Extern].

The following functions are available for the different workflow outbox views:

- ![Update]
  You use this function to update the workflow outbox.

- ![Display work item] (not in Started workflows view)
  This function goes directly to the work item display [Seite 1411]

- ![Change agent]
  You use this function to perform an ad-hoc agent assignment [Seite 1097]. [Seite 1411]

- ![Display workflow log]
  This function displays the workflow log [Seite 1420].

- ![Display attachments] (refer to attachment management [Seite 1429])

- ![Environment]
  - Display objects in workflow (dialog work items only)
  - Display workflow relationships [Seite 1436]
  - Start Workflow [Seite 1450]
• Create link [Seite 1434]

• Change selection period
  • Today only
  • Last 7 days
  • Last 30 days
  • User-defined …

The functions described above can be called using the relevant context menu.

Activities
You can access the workflow outbox in the overview tree in the Business Workplace [Seite 1368] via Workplace → Outbox.
Workflow Resubmissions

Use
The workflow resubmissions [Extern] function can be used to resubmit work items for processing at a later point in time or periodically and to display these work items in a list. Technically, the following applies to work items which appear in your workflow resubmission folder: the work item status is set to waiting, your name is entered as the actual agent, and the requested start date is set to the resubmission date.

Integration
As is the case for the workflow inbox [Seite 1408] and the workflow outbox [Seite 1441], the workflow resubmissions is an integral part of the Business Workplace.

Features
Workflow Functions
The screen area in which the worklist is displayed has a toolbar containing the workflow functions and functions for configuring the list. For information about these functions, refer to the documentation on the ABAP List Viewer [Extern]. The following functions are available in the workflow resubmissions:

- **Update**
  You use this function to update the workflow resubmissions.

- **Display work item**
  This function goes directly to the work item display [Seite 1411].

- **Display workflow log**
  This function displays the workflow log [Seite 1420].

- **Display attachments**
  This function is only available if the work item has attachments. Refer to attachment management [Seite 1429].

- **End resubmission**
  The work item is put back into the workflow inbox. It has the status reserved.

- **Environment**
  - Display objects in workflow (dialog work items only)
  - Display workflow relationships [Seite 1436]
  - Start Workflow [Seite 1450]
  - Create link [Seite 1434]

The functions described above can be called using the relevant context menu.

Activities
You can access the workflow resubmissions in the overview tree in the Business Workplace [Seite 1368] via Workplace → Resubmissions.
You are asked for a date for resubmission of the work item. The work item then disappears automatically from the workflow inbox and is not displayed there again until the date specified.
Work Item Preview

Use
The work item preview is in the bottom right corner of the Business Workplace screen and provides a preview of the properties of a work item. All the information on a work item is displayed if you choose the function Display work item in the worklist of the workflow inbox [Seite 1408] or the workflow resubmissions [Seite 1443]. This function goes directly to the work item display [Seite 1411].
A concise view of the workflow log [Seite 1420] is offered in the workflow outbox [Seite 1441] when the started workflows are displayed.

Integration
The work item preview is part of the Business Workplace [Seite 1368].

Features
Workflow functions
The description of the work item is displayed in the work item preview. It is also possible to access objects and attachments of the work item directly from the preview.
The concise view of the workflow log enables you to access processed steps (individual work items), their agents and information objects addressed so far.
For more information about the concise view of the workflow log, please refer to Workflow Relationships [Seite 1436].

Tips & tricks
"Tips & tricks" are displayed when you position the mouse pointer on the title of a "tips & tricks" entry. Whenever the work item preview is updated, a new title is offered.
You can activate/deactivate "tips & tricks" in your personal workflow settings [Seite 1395] (Settings → Workflow settings → Personal settings).

User exit
The work item preview can be adapted to customer-specific requirements using a user exit [Extern].

Activities
The work item preview is activated/updated whenever you select a work item in your worklist.
WebFlow

Purpose
WebFlow provides functions that enable SAP Business Workflows to be executed across the Internet. WebFlow makes it possible to:

- Send XML documents to other systems from a Business Workflow
- Start a Business Workflow on another SAP System from your SAP System
- React to the results of a Business Workflow executed in another system
- Start a Business Workflow when an appropriate XML document is received

Prerequisites
A Web server and an Internet Transaction Server (ITS) must be configured for the SAP System for the transmission and reception of the XML documents to be successful. For further information, refer to Defining the Web Server [Seite 1513].

Process Flow

Overview of the process flow of a Web activity

Your SAP System A  Transmission with http or https  System to be called B

You can either send XML documents you have created yourself or have WebFlow create a Wf-XML document [Extern] from your data. For further information, refer to Creation of a Wf-XML Document [Seite 1093].

To send XML or Wf-XML documents, you use the step type Web activity [Extern] in a workflow. For further information, refer to Maintenance of a Web Activity [Seite 1037].

The transfer protocols http and https are used for the transmission. A transmission of this type is not completed until the other system has sent back an http reply.

A special service [Extern] must be active for a Business Workflow to be started by an inbound Wf-XML document in a SAP System. This service processes the inbound Wf-XML document and writes the data
contained into the import parameters of the Business Workflow to be started. The Business Workflow is then started and executed.

The following are possible scenarios:

**Sending an XML document to another system**

You can send an XML document that you created yourself from your SAP System A to system B using a WebFlow. The Web activity sends your XML document to the recipient you specified. If system B’s http reply contains an XML document, it can be stored in the workflow container. The Business Workflow in system A is then continued.

**Sending a Wf-XML document to another system**

You can have the WebFlow create the XML document to be sent. A Wf-XML document is then sent to system B, which contains your data and the http reply is evaluated by the WebFlow. The Business Workflow in system A is then continued.

**Sending a Wf-XML document to another system with feedback**

This scenario is the same as the scenario without feedback up to reception of the http reply. The Web activity waits for a Wf-XML reply document. The Web activity writes the data in it into the workflow container and the Business Workflow in system A is continued.

**Sending a Wf-XML document to another SAP system and starting a Business Workflow**

If you want to start a Business Workflow in SAP System B using a WebFlow, have the system create a Wf-XML document for you from your data. You can use a wizard to generate the URL to which the Wf-XML document is sent. To ensure that the Business Workflow to be started in system B receives all the data required, you must know the interface [Extern] of the Business Workflow to be called in full. The http reply is evaluated by the WebFlow and the Business Workflow in System A is then continued.

**Sending a Wf-XML document to another SAP system and starting a Business Workflow with feedback**

This scenario is the same as the scenario without feedback up to reception of the http reply. Once the Business Workflow in system B is terminated, the export parameter data of the interface is sent back to the calling Business Workflow in system A as a Wf-XML reply document. The Web activity writes this data into the workflow container and the Business Workflow in system A is continued.

**Starting a Business Workflow with an inbound Wf-XML document without reply**

When a Wf-XML document arrives, the WebFlow tries to start the Business Workflow named in the document. The data contained in the Wf-XML document is written into the relevant import parameters of the interface. The WebFlow sends an http reply back.

**Starting a Business Workflow with an inbound Wf-XML document with reply**

This scenario is the same as the previous scenario without reply. Once the Business Workflow has been terminated the WebFlow creates a Wf-XML reply document. The export parameter data of the interface is written into this, and the document is sent back to the calling system.
Workflow Developer

Definition
Person within the system of workflow roles [Seite 94] who implements the workflow based on the business process model developed by the process consultant [Seite 96] and using the SAP Business Workflow definition tools.

Use
This part of the documentation is to be used in conjunction with the user menu for the workflow developer. This user can access this user menu on the screen SAP Easy Access via .
Assign the role SAP_BC_BMT_WFM_DEVELOPER to the user.

Since workflow developers theoretically need all the functions of SAP Business Workflow, the role SAP_BC_BMT_WFM_DEVELOPER contains the authorizations for all SAP Business Workflow transactions. In this respect, the documentation assigned to this role is incomplete, because it contains only the documentation that you definitely need as a workflow developer.

The documentation for all transactions can be found under SAP Business Workflow - Navigation [Seite 972].

Structure
The documentation for this role consists of:

- **Tutorial: Workflow Modeling [Seite 1583]**
  Here you will find the tutorial on workflow modeling. The complete set of tutorials can be found under SAP Business Workflow - Tutorials.

- **Modeling Workflows [Seite 279]**
  Here you will find vital documentation for workflow developers. In particular, you will find a typical procedure model for this role.

Integration
This role is part of the role-based procedure model containing the following other roles:
Tutorial: Workflow Modeling

Purpose

This tutorial uses an example in a series of easy-to-follow units to explain the most important tools in SAP Business Workflow. The example used here is based on a scenario for approving a notification of absence. At the end of this tutorial, you will have defined and executed a workflow that automatically submits a notification of absence (leave request) to your superior for approval, and informs the requester of the result of the approval process. You will become familiar with the following areas of SAP Business Workflow throughout the course of this tutorial:

- Definition tools
- Business Workplace
- Reporting and analysis tools
- Using the Workflow Builder

The tutorial is not intended to provide a full description of all functions and concepts. This information is available in the documentation on SAP Business Workflow [Extern]. This tutorial does not deal with the definition of object types. If you want further information on this subject, please work through the tutorial on Workflow Programming [Seite 1667].

Process Flow

Work through the individual units in this tutorial in the specified order. Important units are followed by tests that you can use to test what you have learned to date. Please make sure to complete these tests.

Result

Example - the notification of absence

The scenario in this example begins with the completion of a leave request by an employee (requester or creator of the notification of absence). The completed form is then forwarded automatically to the head of department (employee’s superior).

- If the head of department approves the request, the employee receives a notification and the workflow is terminated.
- If the head of department rejects the request, the employee can decide to revise the request (possibly in accordance with the head of department’s wishes) or withdraw it. If the employee decides to revise it, the request form is resubmitted to the head of department after the revision is made.
Business Process and Workflow: Example

The diagram above shows that additional steps could follow the approval, such as updating the leave account, or notifying the secretary. These steps, however, do not arise in this example.

**All of the units at a glance**

The diagram below shows all of the units in this tutorial. Similar units are listed in the same column.
Unit 1: Organizational Structure

Use

In order for the workflow system to establish the relationship between the requester and their superior, you must create an organizational plan in the system. For this tutorial, of course, this organizational plan does not have to be complete and valid across the enterprise. To keep the test as simple as possible initially, define an organizational plan that only contains one administrator and one head of department. Assign both items to yourself. As a result, all work items will appear in your own Business Workplace. Later you will modify the organizational plan such that you will have to work through the scenario with two users.

Procedure

The organizational plan required for this tutorial consists, when complete, of one organizational unit (="department"), which contains two positions: a head of department and an administrator. Each position is described by one job and each position is assigned one user as holder. The head of department position is also designated as chief position of the organizational unit.

Of course, a “real” organizational plan is created by arranging several organizational units with their positions in a hierarchy. Usually several positions are described by one job.

For further information, refer to the documentation Organizational Plan.

The procedure in this unit is divided into four parts:
1. You create an organizational unit.
2. Enter necessary jobs as required.
3. You create a position for the head of department in the new organizational unit.
   You define a position in three steps:
   i. You create a position that is assigned your organizational unit.
   ii. You assign a holder to the position.
   iii. You assign a job to the position.
3. You create a position for the administrator in the organizational unit.

Creating an organizational unit

5. Choose Tools → SAP Business Workflow → Development → Definition tools → Organizational Management → Organizational plan → Create.

6. Confirm the validity period proposed in the dialog box Creating a Root Organizational Unit.
   This takes you to the Create Organization and Staffing (Workflow) screen. This user interface is divided into four screen areas:
7. On the Basic Data tab in the details area, enter an abbreviation and a name in the Organizational unit input fields.
   
   **Abbreviation:** <ini_sales>
   **Name:** <OrgUnit: Sales (ini)>

8. Choose create. You can now create the position for a head of department and one administrator.

**Create jobs**

When enhancing an organizational unit, the necessary jobs are usually already available. For this tutorial however, you create the necessary jobs for head of department and administrator yourself.

3. Choose Edit → Create jobs.

You go to the dialog box Create jobs. The lower area contains a list of existing jobs and the upper area contains an input table in which you can create new jobs by entering abbreviations and names.

4. In the input table, enter an abbreviation and a name for each of the new jobs.
   
   **Job - head of department:**
   **Object abbreviation:** <ini_dhead_C>
   **Name:** <job: head of department (ini)>
   **Job administrator:**
   **Object abbreviation:** <ini_admi_C>
   **Name:** <job: administrator (ini)>
5. Choose ☐.

Creating Position for Head of Department

1. Change to the overview area in the staff assignments [Extern] of the organizational unit, in order to assign positions, jobs and holders. Choose the arrow ☐ on the right next to the ☐ and then the staff assignments (list).

2. Choose ☐.

A new position is then created in the staff assignments and is displayed in a new line in the table. The position is vacant and no job is assigned to it.

3. Open the details view for the new position in the details area by double-clicking on the entry in the table.

4. On the Basic Data tab, enter a code and a description in the Position input fields. Overwrite the previous contents.

   Abbreviation: <ini_dhead_S>
   Description: <position: head of department (ini)>

Assigning a holder to the position

You now assign R/3 users to the positions. The staff assignments for your organizational unit are displayed and you see the vacant position in the table.

5. Select User in the search area and enter the search criteria in order to find your user names.

   All of the user names that match your search criterion are listed in the selection screen.

6. Select your user name in the selection area and drag it to the Person/User column of the position in the overview area.

   Confirm the message that the relationship period of the validity has been changed.

7. Set the Head of own organizational unit indicator in the details area.

Assigning a job to the position

Assign the job of the head of department you created earlier to the position.

8. Select Job in the search area and enter the search criteria in order to find the job of the head of department.

   All jobs that match your search criterion are listed in the selection screen.

9. Select job: head of department (ini) in the selection area and drag it to the Job column of the position in the overview area.

10. Choose ☐.

   The job is assigned the position. Check this by switching to the staff assignments of the organizational unit. Select the organizational unit in the overview area, choose the arrow ☐ on the right next to the ☐ and then the staff assignments (list). The newly created job is displayed in the Job column.

Create position for administrator and assign holder and job

You are now in the staff assignments of the organizational unit.

1. Choose ☐.
A new position is then created in the staff assignments and is displayed in a new line in the table. The position is vacant and no job is assigned to it.

2. On the Basic Data tab in the details area, enter an abbreviation and a name in the Position input fields. Overwrite the previous contents.

   ID: <ini_admi_S>
   Description: <position: administrator (ini)>

3. Choose.

4. Select User in the search area and enter the search criteria in order to find your user names.

5. Select your user name in the selection area and drag it to the Person/User column of the position in the overview area.

   Confirm the message that the relationship period of the validity has been changed.

6. Select Job in the search area and enter the search criteria in order to find the job of the administrator.

7. Select job: administrator (ini) in the selection area and drag it to the Job column of the position in the overview area.

8. Choose.

**Result**

Display your entire staffing schedule again and make sure that all the information listed is correct. You can display a detailed view of jobs, users, and positions. Choose the relevant cell in the table by double-clicking it.

In the details view of a position or job, all of the assigned tasks are displayed on the Tasks tab.

You have now completed the first unit (defining the organizational plan). You can now start on the next unit. To exit processing of the organizational structure, choose **Back**.
Unit 2: Creating a Workflow

Use
To define the flow of the approval process in the system, you first create a multistep task workflow template.
For further information, refer to Definition of Multistep Tasks [Seite 1194] in the SAP Business Workflow documentation.

Procedure: Creating a workflow template
   This takes you to the Task: Maintain screen.
2. In the Task type field, choose the Workflow template entry.
3. Choose .
   The Workflow Template: Add screen is then displayed.

Entering the basic data for the workflow template
1. Enter an abbreviation (of your choice) and a name (of your choice) for the workflow template to be created.
   Abbreviation: <ini_ws>
   Name: <Workflow: notification of absence (ini)>
2. Choose .
3. In the Create Object Directory Entry dialog box, choose Local object.
   The system has now allocated an 8-digit number for your workflow template. This number is made up of the 3-digit prefix number (Customizing setting) and a 5-digit number taken from a number range. This number together with the letters WS forms the workflow ID. Make a note of the ID so that you can later call the workflow more quickly.

Entering a workflow description
4. Go to the Description tab.
5. Choose the text type Task description and then .
   The SAPscript text editor is then launched.
6. Enter a text that describes the task. You can use the following proposal:
   <You can use this workflow to create a notification of absence. This will then be sent to your superior for review. You will be notified of the result of this review. If your request is rejected, you can revise it and resubmit it for review or withdraw it completely.>
7. Choose to return to the workflow template and then save the workflow template.
Determining agents
To enable the workflow template to be started in dialog, the workflow template must be assigned to its possible agents.

8. Select Additional data → Agent assignment → Maintain
   The workflow template: agent assignment screen is displayed.

9. Position the cursor on the name of your workflow template.
10. Choose Properties…
    The dialog box Task: <Workflow: Notification of absence (ini)> is displayed.
11. Set the General task flag.
12. Choose ✔ Copy and then 📚.

Designating your workflow template as a general task [Extern] means that every user in the system is authorized to start this workflow template in dialog.

Result
You have now created a workflow template as a "framework" for a workflow definition.
To create a workflow definition, you need tasks that are used as steps in the workflow definition.

Additional information: Tasks
A task can either be a single-step task [Extern] or a multistep task [Extern]. In our example of the notification of absence, making the application and checking the application are single-step tasks. The entire procedure involved in processing a leave request consists of these (and other) single steps, and is therefore a multistep task.

In this tutorial, a single step task is generally described as a task. The difference between the task types is only explicitly explained if there is a risk of confusion over multistep tasks. The term workflow is used as the umbrella term for the workflow template (and thus the multistep task) and the workflow definition. More precise differentiation only takes place if there is a risk of confusion.

On a technical level, the main difference between single-step tasks and multistep tasks is that:

- Every single-step task refers to an object method
- Every multistep task refers to a workflow definition
  A workflow definition is made up of a sequence of different steps (for example, all steps required to process a leave request).

Standard tasks and workflow templates are client-independent. Customer and workflow tasks are client-dependent. Customer and workflow tasks are no longer supported and are therefore no longer used in this tutorial.

An Activity is a possible step type in a workflow definition. An activity refers to a task. It references a task, which can be a single-step or multistep task (subworkflow).

For further information, refer to Tasks and Task Groups [Seite 1172] in the SAP Business Workflow documentation.
Unit 4: Defining and Inserting Task "Create Notification of Absence"

Use
The notification of absence is created in the first step of the workflow. You will also need a task that executes this function. You can create the task separately from the workflow definition or directly from the Workflow Builder. This unit describes how you create the task from the Workflow Builder. For further information on creating an individual task, refer to Definition of Single-Step Tasks [Seite 1175] in the SAP Business Workflow documentation.

Prerequisites
This task is a single-step task and it references one object method. The object type used (FORMABSENC (notification of absence)) and the required method (Create) are already defined and implemented in the Business Object Repository. You are in the Workflow Builder and the initial workflow definition is displayed.

Procedure
Creating an Activity as a Step in the Workflow Definition
Add a first step of the type Activity to the workflow definition.
1. Choose under step type. The cursor changes shape.
2. Position the cursor on the Undefined step and click.
   By doing so, you add an activity to your workflow definition. The step definition is shown in the right-hand part of the screen. You are on the tab page Control.

Defining a Task "Create Notification of Absence"
You now create a new task, with which a notification of absence can be created.
1. Choose the arrow on the right next to the and then Create task.
   The Basic data tab in the Standard task: Create screen is then displayed.

Entering the basic data
2. Enter an abbreviation (of your choice) and a name (of your choice) for the standard task to be created.
   Abbreviation: <ini_create>
   Name:<Create notification of absence (ini)>

Work item text
The work item text appears as an information text in the Business Workplace as soon as there is a work item for the step with this task.
3. Enter Create notification of absence in the Work item text field.

Object type and method
4. Enter an object type and a method:
   Object type: FORMABSENC
Unit 4: Defining and Inserting Task "Create Notification of Absence"

Method: CREATE
You can also use the F4 input help to select the method. Additional indicators associated with this method are set automatically.

You can view the definition of the object type entered by double-clicking on it. Information about its components is provided.

5. Save your task as a local object.
   The system has now allocated an 8-digit number for your task. This number is made up of the 3-digit prefix number (Customizing setting) and a 5-digit number taken from a number range. The identification of this task consists of TS and the eight digit number.

Determining agents
Selecting the possible agents [Extern] for this task determines who is to be authorized to make leave requests. For this task, this should be all employees in your enterprise. Define the task as a general task [Extern].

1. Select Additional data → Agent assignment → Maintain
   This takes you to the Standard task: Maintain Agent Assignment screen.

2. Position the cursor on the name of your task.

3. Choose Properties…
   The dialog box Task: <Notification of absence (ini)> is displayed.

4. Set the General task indicator.

5. Choose Copy and then .

6. Exit the task definition with .

Interim result
You have now defined your first task in full. This task is used in the activity that you created in the Workflow Builder.

Completing Activity
The Define Container Elements and Binding dialog box is then displayed in which the system proposes a change to the workflow container and a binding definition.

- Upper part of the dialog box: Proposal for new container elements in the workflow container.
  The system proposes that you create a local container element in the workflow container, which can store the reference to the notification of absence created in this step. The element has the technical name AbsenceForm.

- Lower part of dialog box: Proposed binding definition
  The system proposes that you define a binding from the task container to the workflow container. This binding transports the reference to the generated object (i.e. the created notification of absence) from the task (from the _WI_Object_ID element in the task container) to the workflow (to the AbsenceForm element in the workflow container).
Unit 4: Defining and Inserting Task “Create Notification of Absence”

Further information on automatic binding definition proposals is available under Binding Definitions from the Workflow Container [Seite 1223] and Binding Definitions from the Task Container [Seite 1225] in the documentation on SAP Business Workflow.

1. Confirm the proposal with ✔.

   The new AbsenceForm container element is displayed in the workflow container. Local container elements are marked 🗡. The remaining elements in the workflow container are workflow system elements. For further information, please refer to Workflow System Elements in the Workflow Container [Extern].

2. Enter a text to describe the step.

   You can, of course, leave the description of the task that is proposed by the system.

   (The current step in your workflow definition appears here.)

3. Go to the Outcomes tab.

   All of the outcomes for this step are displayed here.

4. Enter a text for the description.

   The description (proposal: <Application filled out>) in the Task executed synchronously line.

   Each step has at least one subsequent event (in this case: Task executed synchronously). The description is the text with which the outcome is labeled in the workflow definition.

   The Form does not exist and Form could not be generated outcomes are the possible exceptions that could occur in the method execution. You “run the risk” of not intercepting these exceptions and not modeling any exception handling in your workflow definition. If one of the exceptions actually occurs at runtime, the workflow will assume the error status.

Checking the binding definition (optional)

1. Go to the Control tab.

2. Choose Binding (present).

   The WF Builder: Binding for Step ‘<Create notification of absence>’ dialog box is then displayed.

   - In the upper part of the screen, the binding defined from the workflow container to the task container is displayed. The system executes this binding before providing the task for execution.

     This binding direction is not relevant for this step since no information is to be transported from the workflow to the task.

   - In the lower part of the screen, the binding defined from the task container to the workflow container is displayed. This system executes this binding after the user has processed the task.

     Here, the binding is _WI_ObjectId. &ABSENCEFORM& has been automatically defined here by the system. This binding ensures that the reference to the notification of absence created is not only known in the single-step task, but also in the workflow.
Unit 4: Defining and Inserting Task "Create Notification of Absence"

You can view the contents of both containers by choosing Workflow container or Task container.

(The element of the task container _WI_Object_ID is called Notif. of absence.)

3. Choose ✅.
You then return to the step definition.

**Entering the agents responsible**

On the tab page Control, choose the entry Workflow initiator (expression) in the area agent as the agent responsible [Extern].

The system enters the expression &_WF_INITIATOR& in the Expression field and in the input field beside the checkbox.

You start your executable workflow later manually. At this point in time, the system fills the workflow container element _WF_Initiator automatically with your user name.

The above assignment informs the workflow system that the first work item for creating the notification of absence should be addressed as the "starter" as the workflow.

You return to the screen Workflow definition: Create step: Activity

**Concluding step definition**

1. Choose ✅ to check the step definition.
2. Choose ✅ to exit the step definition.
3. Choose ✅ to view the entire workflow definition in the workflow area on the screen.
   You will notice that your workflow definition now contains a new step (the activity just created). The undefined step is also still there.
4. Choose ✅.
   The workflow definition is checked, saved, and, provided it does not contain any errors, activated. The system displays a message if problems are encountered during the test.
5. Exit the Workflow Builder.

**Result**

You have now completed this unit and can start the next unit.
Unit 3: Starting the Workflow Builder

Use
A workflow definition is created with the help of the Workflow Builder.

Prerequisites
You have now created a workflow template (Unit 2: Creating a Workflow).

Procedure

Calling the Workflow Builder


   The Workflow Builder is then launched with the workflow definition that was processed last. If the Workflow Builder has not yet been called, then an initial workflow definition is displayed.

   The Workflow Builder screen is divided into the following areas.

   The workflow definition is displayed in the Workflow area on the screen. If the workflow is shown in display mode, the step type area is not displayed. For further information, refer to Workflow Builder in the SAP Business Workflow documentation.

2. Choose and enter the abbreviation of your workflow template (eight digit number).
You can use the F4 input help. Enter the start of the abbreviation for your workflow (for example: <ini*>) and choose ✔️. All of the workflows that match are displayed. Select your workflow and choose ✔️.

**Result**
The workflow definition is displayed and if necessary can be processed. If the Workflow Builder is called for the first processing of a newly created initial workflow definition, the following graphic is displayed:

**Initial Presentation of a Workflow Definition**

If the graphic displayed differs from that above, check the graphic settings in the Workflow Builder. Choose Extras → Options, and click the Graphic tab. Choose the Without event-driven process chains view.

- The start of the workflow definition is indicated by ✅ Start Workflow. If the new workflow definition has been called from the definition of a multistep task for which triggering events [Extern] are defined, these are represented by the symbol ✅ and their description.
- The end of the workflow definition is indicated by ✅ Workflow terminated.
- The area in which the new workflow definition can be added is indicated by an undefined step with an outcome. Steps are displayed in the form of symbols. The description of an outcome is illustrated in the standard view with an arrow.

This unit is now complete. In the following unit you create the first step in your workflow and define a task for this purpose at the same time.

**Creating an Activity as a Step in the Workflow Definition**
Add a first step of the type Activity to the workflow definition.

1. Choose ✅ under step type. The cursor changes shape.
2. Position the cursor on the Undefined step and click.

   By doing so, you add an activity to your workflow definition. The step definition is shown in the right-hand part of the screen. You are on the tab page Control.
Unit 3: Starting the Workflow Builder

Entering basic data for control

3. Enter the abbreviation (TS<8-digit number>) of the standard task *Create Notification of Absence* in the *Task* field and choose.

   If you no longer know the full abbreviation, you can use different methods to search for it:
   - Use the *F4 input help* to search via the object type and method of the single-step task.
     This brings you to the *Search and Find: Tasks* dialog box.
     On the tab page *Obj. type and method*, enter the *object type FORMABSENCE* and the *method CREATE*.
     Then double-click your single-step task in the list. You should recognize your single-step task from your initials in the abbreviation. Choose.
   - Enter a character string in the abbreviation or description and choose.
     A list of tasks is then displayed in which you can double-click the desired task.

Every activity, as a step in a workflow definition, refers to a task.

At this point, the unique reference to this task is entered. The identification is made up of an abbreviation (T, TS, WF, or WS) followed by an 8-digit number.

If you select a workflow task (WF) or workflow template (WS) as your task, your activity is a subworkflow [Extern]. If you select a customer task (T) or standard task (TS), your activity is a single step.

The *Define Container Elements and Binding* dialog box is then displayed in which the system proposes a change to the workflow container and a binding definition.

   - Upper part of the dialog box: Proposal for new container elements in the workflow container.
     The system proposes that you create a local container element in the workflow container, which can store the reference to the *notification of absence* created in this step. The element has the technical name *AbsenceForm*.
   - Lower part of dialog box: Proposed binding definition
     The system proposes that you define a binding from the task container to the workflow container. This binding transports the reference to the generated object (i.e. the created notification of absence) from the task (from the _WI_Object_ID element in the task container) to the workflow (to the *AbsenceForm* element in the workflow container).

Further information on automatic binding definition proposals is available under *Binding Definitions from the Workflow Container* [Seite 1223] and *Binding Definitions from the Task Container* [Seite 1225] in the documentation on *SAP Business Workflow*.

5. Confirm the proposal with.

The new *AbsenceForm* container element is displayed in the object area of the workflow container folder. Local container elements are marked. The remaining elements in the
Starting the Workflow Builder

6. Enter a text to describe the step.
   
   You can, of course, leave the description of the single-step task that is proposed by the system.
   
   (The current step in your workflow definition appears here.)

7. Go to the Events tab.
   
   All of the outcomes for this step are displayed here.

8. Enter a text for the description.
   
   The description (proposal: <Application filled out>) in the Task executed synchronously line.
   
   Each step has at least one subsequent event (in this case: Task executed synchronously). The description is the text with which the outcome is labeled in the workflow definition.
   
   The Form does not exist and Form could not be generated outcomes are the possible exceptions that could occur in the method execution. You “run the risk” of not intercepting these exceptions and not modeling any exception handling in your workflow definition. If one of the exceptions actually occurs at runtime, the workflow will assume the error status.

Checking the binding definition (optional)

3. Go to the Control tab.

4. Choose Binding (present).
   
   The WF Builder: Binding for Step ‘<Create notification of absence>’ dialog box is then displayed.
   
   – In the upper part of the screen, the binding defined from the workflow container to the task container is displayed. The system executes this binding before providing the task for execution.
     
     This binding direction is not relevant for this step since no information is to be transported from the workflow to the task.
   
   – In the lower part of the screen, the binding defined from the task container to the workflow container is displayed. This system executes this binding after the user has processed the task.
     
     Here, the binding is _WI_ObjectId. &ABSENCEFORM& has been automatically defined here by the system. This binding ensures that the reference to the notification of absence created is not only known in the single-step task, but also in the workflow.

     You can view the contents of both containers by choosing Workflow container or Task container.
     
     (The element of the task container _WI_Object_ID is called Notif. of absence.)

4. Choose ✔.
   
   You then return to the step definition.
Entering the agents responsible

Choose *Workflow initiator (expression)* under *Agents*.

The system enters the expression &_WF_INITIATOR& in the *Expression* field and in the input field beside the checkbox.

By doing so, you choose the agent responsible [Extern].

You start your executable workflow later manually. At this point in time, the system fills the workflow container element _WF_Initiator automatically with your user name.

The above assignment informs the workflow system that the first work item for creating the notification of absence should be addressed as the "starter" as the workflow.

You return to the screen *Workflow definition: Create step: Activity*

Concluding step definition

6. Choose to check the step definition.

7. Choose to exit the step definition.

8. Choose to view the entire workflow definition in the workflow area on the screen.

   You will notice that your workflow definition now contains a new step (the activity just created). The undefined step is also still there.

9. Choose .

   The workflow definition is checked, saved, and, provided it does not contain any errors, activated. The system displays a message if problems are encountered during the test.

10. Exit the Workflow Builder.

Result

You have now completed this unit and can start the next unit.
Unit 5: First Test

Use
You are now going to start a workflow in dialog for the first time, which will run according to your workflow definition. This workflow definition only contains a step for creating a notification of absence so far. For further information on starting workflows, refer to Starting Workflows (Test Environment) [Seite 1452] in the SAP Business Workflow documentation.

Prerequisites
The creation of a notification of absence, which you carry out within your workflow, is reported on a system-wide basis by an event. Other workflows entered as event receivers for this event may therefore be started. The notification of absence created by you (and published by the event) is then also processed with these workflows.
This event is not used in connection with this tutorial. To ensure that only you work exclusively with the notification of absence, you can deactivate any existing linkages between the event and its receivers. Proceed as follows:

2. Enter FORMABSENC in the Object type field.
3. Enter created in the Event field.
4. Choose .
   The system simulates the event FORMABSENC.Created and establishes which workflows would be started.
5. If you find one or more entries in the list under the branch Tasks without syntax errors to be started, position the cursor on an entry and choose Event linkage.
   The dialog box Event linkage: Triggering events is displayed.
6. Position the cursor on the entry with the red background for the event and choose.
   The linkage between workflow and event is deactivated.
7. Choose .
8. Repeat steps 5 to 7 or repeat the whole simulation as applicable.

Procedure
Starting the workflow and filling out the form
1. If you are still in the Workflow Builder, choose .
The Start Workflow (Test Environment) screen is then displayed. The number of your workflow template is entered.

If you are no longer in the Workflow Builder, choose Tools → Business Workflow → Development → Runtime Tools → Start Workflow (Test Environment). Enter the identification of your workflow or use the F4 input help.

2. Choose .

To handle work processes more quickly, SAP Business Workflow supports Advancing with Immediate Dialog [Extern]. When a workflow is started in dialog, this means that the first work item of the workflow is made available for processing immediately provided that the user who starts the workflow is also one of the recipients [Extern] of this first work item. This is the case here because you entered the container element _WF_Initiator as the agent in the step definition. The Create notification of absence screen, therefore, is displayed immediately where you can see the notification of absence form. It is the method Create that is executed with the first work item.

3. Fill out the form with entries of your choice.

4. Choose .

The Start Workflow (Test Environment) screen is then displayed again.

5. Choose and exit the Workflow Builder if necessary.

Starting a workflow / Business Workplace

Before you extend your workflow definition, carry out the following test. It will familiarize you with the Business Workplace.


You go to the screen Start task.

2. Choose your workflow in the table on the left.

3. Choose Start on the right-hand side of the screen.

The notification of absence form is displayed.

4. Do not make any entries and do not save. Instead, choose .

You have now started the workflow but canceled the processing of its first step. But the processing of the work item is not yet completed. You have only broken the processing chain of advancing with immediate dialog.

5. Choose .


7. Open the Inbox node and choose the Workflow folder.

Your workflow inbox now contains (at least) one work item for processing. This is the work item for creating a notification of absence, the processing of which you canceled previously.

8. Select the work item, if necessary, and choose .

You return to the Create notification of absence screen where the notification of absence form is displayed.
9. Fill out the form and choose 📝.

   The work item disappears from your workflow inbox.

**Result**

You have now completed this unit and can start the next unit.
Reporting and analysis

In this first test, you will take a look at the work item analysis function.

   
   The Work Items Per Task screen is then displayed.

2. Select the monitoring period Today.

3. Select the work item type (Sub-)Workflow only.

4. Choose .

   The system then determines all of the workflows that were started today. These are then listed on the Work Items Per Task from <Date> to <Date> screen. The number of work items in each workflow is also shown here.

   ⚠️

   Make sure that you only ever analyze the work items for your workflow and your absence notification.

5. Display the list of associated work items by double-clicking the workflow ID.

   Both the workflows you just started for this workflow identification are displayed with status Completed.

6. Display the workflow log by double-clicking a work item ID.

   The Workflow Log screen is then displayed.

   For information on the workflow log, see Workflow Log [Seite 1420] in the SAP Business Workflow documentation.

7. Exit the work item analysis function.
Unit 6: Creating a Task "Check Notification of Absence"

Use

You define the task Check notification of absence. This task is incorporated into your workflow definition as the second step.

As this is not the first task you have defined, many steps will doubtless be familiar to you. (Refer to Unit 4: Defining and Inserting Task "Create Notification of Absence [Seite 1593]." )

Prerequisites

The object type used (here: FORMABSENC (notification of absence)) and the required method (here: Approve) are already defined and implemented in the Business Object Repository.

Procedure

   
The screen Task: Maintain is displayed.
2. In the Task type field, choose the Standard task entry.
3. Choose .
   
The Basic data tab in the Standard task: Create screen is then displayed.

Entering the basic data

1. Enter an abbreviation (of your choice) and a name (of your choice) for the task to be created.
   
   Abbreviation: <ini_check>
   Name: <Check notification of absence (ini)>

Object type and method

2. Enter an object type and a method:

   Object type: FORMABSENC
   Method: APPROVE

   You can also use the F4 input help to select the method.

Work item text

3. In the Work item text field, enter the text Check notification of absence from.
4. Choose . Save your standard task as a local object.
5. You can include variables in your work item text to be filled with values from the task container at runtime.

   The name of the creator of the notification of absence is to be included into the work item text. As soon as the work item appears in the superior's Business Workplace, the name of the creator appears in the work item information text.

   Position the cursor in the work item text after the word of.
Unit 6: Creating a Task “Check Notification of Absence”

6. Choose \texttt{\textbf{\textbullet{}}}.  

The Please choose an expression dialog box is then displayed.

7. Choose the expression Notif. of absence by double-clicking the Container node. \texttt{\textbullet{}} Issuer object ref. \texttt{\textbullet{}} Name.

\[ \text{If } _{WI}\text{Object_ID} \text{ is displayed instead of Notif. of absence, choose } \texttt{\textbullet{}}. \text{The system then displays the description of the expressions instead of their technical names.} \]

\[ \text{The variable } _{WI}\text{Object_ID}\text{.Creator.Name} \text{ has been added to the work item text automatically. The variable name displayed here is the technical name of the expression.} \]

You can integrate all of the elements in the task container in your work item text. This enables you to include information into the work item text, which is not available until runtime.

\textbf{Entering the description text}

1. Go to the \textit{Description} tab.
2. Choose \textbf{Task description} in the \textit{Test type} field.

\[ \text{The current task description is then displayed. This text is intended to inform future recipients of a work item in which this task is referenced and help them in their work.} \]

\[ \text{The text will often be similar to the work item text or may supplement it.} \]

3. Choose \texttt{\textbullet{}}.
4. Enter the following text:

\[ \text{Please check notification of absence no. } _{WI}\text{OBJECT_ID}.NUMBER\text{ from } _{WI}\text{OBJECT_ID}.CREATEDATE\text{ of employee } _{WI}\text{OBJECT_ID}.CREATOR.NAME\text{.} \]

\[ \text{Decide whether to approve the request.} \]

\[ \text{If you want to insert an expression from the task container as a text variable, choose Include } \rightarrow \text{ Expression. Then choose the relevant expression in the Please choose an expression dialog box by double-clicking the Container node.} \]

5. Choose \texttt{\textbullet{}}.
6. Choose \texttt{\textbullet{}}.

\textbf{Determining possible agents}

1. Select \textit{Additional data} \rightarrow \textit{Agent assignment} \rightarrow \textit{Maintain}

\[ \text{This takes you to the Standard task: Maintain Agent Assignment screen.} \]

2. Position the cursor on the name of your task.
3. Choose \texttt{\textbullet{}}.

\[ \text{You go to the dialog box Choose agent type.} \]

4. Double-click Job.

\[ \text{The dialog box Choose Job is displayed.} \]
5. Enter either part of or the full abbreviation of the job that you created for the head of department in Unit 1: Organizational Plan [Seite 1587].

You return to the screen Standard task: Maintain agent assignment.

6. Choose 

You have linked the task to the job of a head of department. In the organizational plan, you use this job to describe a position. The holder of the position is also displayed.

7. Choose 

8. Choose 

Result

The task for checking the notification of absence is defined in full. You can now exit the screen for defining a task.

In the next unit, you will incorporate this task into the workflow definition.

Display your organizational plan again.

  
  b. Choose your organizational unit in the search area. Display it by double-clicking the entry in the table.
  
  c. In the overview screen, choose the arrow on the right next to the and then Staff assignments (list).
  
  d. Display a detailed view of the position of the head of department by double-clicking the entry in the list.

Here, you can see that this position is assigned to the job and, on the Tasks tab, that it is also assigned to the task Check notification of absence.
Unit 7: Including "Check Notification of Absence" in the Workflow Definition

Use
The task with which an employee can check a notification of absence is ready to be included into your workflow definition as the next step.

Prerequisites
You have defined a workflow with a step (Unit 4: Task Defining and Inserting “Create Notification of Absence” [Seite 1597]) and created the task Check Notification of Absence (Unit 6: Create Task “Notification of Absence” [Seite 1606]).

Procedure

Calling the Workflow Builder
   The Workflow Builder is then launched in display mode with the workflow that was processed last.
2. Work through the following steps to display your workflow for this tutorial in change mode.
   i. If the workflow you created for this tutorial is already displayed, choose . Go to step 3.
   ii. If a different workflow is displayed, check whether the workflow you created is listed in the My available tasks area. Double-click it and choose . Go to step 3.
   iii. Choose . Enter the complete abbreviation in the Task field on the Open other workflow definition dialog box and press RETURN. You can now also enter the workflow abbreviation. Enter either the full abbreviation or just the first characters (for example: <i*>). Press F4. The search result is then displayed. Choose your workflow by double-clicking it. When you enter the abbreviation in the Task field on the Open other workflow definition dialog box, all of the available versions are displayed. Since you have not created any versions, select version 0000 and choose . The workflow is then loaded in display mode. To switch to change mode, choose .
   3. Choose to display the entire workflow in the workflow area on the screen.

Creating a container element in the workflow container
You need an additional container element in the workflow container to store the name of the user who will execute this step. This user name is to be used later in the notification text that is sent to the requester.
1. Choose the entry <Double-click to create> by double-clicking in the Workflow Container.
   The dialog box for entering a container element is displayed.
2. Make the following entries:
   Element: Approver
Unit 7: Including "Check Notification of Absence" in the Workflow Definition

Name: Approver
Description: Approver of the absence request
Reference table: WFSYST
Reference field: AGENT

3. Choose.

You have now added a local container element to the workflow container, which is ready to take an agent name, based on its data type reference. However, you have not yet determined a value for this container element.

Creating an Activity as a Step in the Workflow Definition

1. Choose under step type. The cursor changes shape.
2. Position the cursor on the Undefined step that follows the outcome Request completed and click.

By doing so, you add an activity to your workflow definition. The step definition is shown in the right-hand part of the screen. You are on the tab page Control.

Entering basic data for control

3. In the Task field, enter the abbreviation (TS<8-digit number>) of the task Check Notification of Absence that you defined in the previous unit. Choose.

If you no longer know the full abbreviation, you can use different methods to search for it:

– Use the F4 input help to search via the object type and method of the task.

This brings you to the Search and Find: Tasks dialog box.

On the tab page Obj. type and method, enter the object type FORMABSENCE and the method CREATE.

Then double-click your task in the list. You should recognize your task from your initials in the abbreviation. Choose.

– Enter a character string in the abbreviation or description and choose.

A list of tasks is then displayed in which you can double-click the desired task.

Every activity, as a step in a workflow definition, refers to a task.

At this point, the unique reference to this task is entered. The identification is made up of an abbreviation (T, TS, WF, or WS) followed by an 8-digit number.

If you select a workflow (WS or WS) as your task, your activity is a subworkflow [Extern]. If you select a task (TS or T), your activity is a single step.

You go to a dialog box in which the system proposes binding from the workflow container to the task container.

Here, the object reference to the notification of absence is transported from the workflow container (container element AbsenceForm) to the task container (to the container element _WI_Object_ID).
4. Choose ✍️ to confirm the proposal.

Extending the binding definition
Now define another binding via which the current agent [Extern] of the work item (the superior) is stored in a container element of the workflow container.

5. Choose Binding (present).

The Binding for Step 'Check notification of absence' dialog box is then displayed. If the first column is entitled Element, choose ✒️ to display the element description.

The system has created the following binding definition from the workflow container to the task container:

Notif. of absence ✒️ &AbsenceForm&

This binding ensures that the correct notification of absence is checked in the task.

6. Choose 🎨.

All of the container elements in the task container are now displayed on the left-hand side. You can define a binding from these container elements to the workflow container.

7. Position the cursor in the lower half of the screen, in the empty input field beside the container element Actual agent and call the F4 input help.


In addition to the bindings already existing, the following binding is now defined:

Actual Agent ✒️ &Approver&

9. Choose ✍️.

Determining outcomes
The method APPROVE of the object type FORMABSENC, to which you refer in the standard task definition, is defined with a result [Extern]. The three possible values of this result are offered on the tab page Outcomes as outcomes of this step:

- Approved
- rejected
- New

The ✍️ indicates that these outcomes have already been transferred to the workflow definition. The outcome New is not really appropriate at this point. At this stage of processing, the notification of absence cannot be “new” any more. It should therefore not be incorporated in the definition.

1. Click on the symbol ✍️ in front of the outcome New.

The symbol changes to 🎨.

The outcome Processing obsolete is processed if the relevant work item is set to obsolete via a process control step. This functionality is not used in this tutorial.

The outcome Form not available results from the exception [Extern] defined for the method APPROVE. You could include this outcome into the workflow definition and would then have to model appropriate subsequent steps. However, you do not model
Unit 7: Including "Check Notification of Absence" in the Workflow Definition

anything for the purposes of this tutorial and accept that your workflow will have an error at runtime if this exception is raised.

2. Choose ✅.

Two branches are now inserted in your workflow definition for the two marked outcomes Approved and Rejected.

Inputs for the agent

If you are working through this tutorial alone, do not specify anything for responsibility at this point. If it is processed when the workflow is executed, therefore, this step will be “offered” for processing to all employees declared as possible agents [Extern] of the single-step task Check notification of absence. This means all user whose positions are described with the job head of department.

In your organizational plan, there is only one users who is a possible agent -> you.

- You have maintained the organizational plan with jobs, positions, and user assignments in Unit 1: Organizational Plan [Seite 1587].
- You have defined the single-step task “Check notification of absence” and have assigned the possible agents in Unit 6: Definition of the Standard Task “Check Notification of Absence” [Seite 1606].

Concluding step definition

Choose ✅.

The workflow definition is checked, saved, and, provided it does not contain any errors, activated. The system displays a message if problems are encountered during the test.

Result

You can subject your workflow definition to a second test.
Unit 8: Second Test

Use
You start your workflow in dialog and two steps are executed. The notification of absence is created in the first step and then checked in the second.

Procedure

Starting the workflow and filling out the form

6. If you are still in the Workflow Builder, choose .
   The_start_workflow (Test Environment) screen is then displayed. The number of your workflow is entered.
   If you are no longer in the Workflow Builder, choose Tools → Business Workflow → Development → Runtime Tools → Start Workflow (Test Environment). Enter the number of your workflow or use the F4 input help.

7. Choose .
   You go to the screen Create Notification of Absence.

From the requester's perspective
When the workflow is started, the work item for creating the notification of absence is provided to you directly for processing. This is because of advancing with immediate dialog. You were introduced to this in the first test.

3. Fill out the form and choose .

From the head of department's perspective
You created the step for checking the notification of absence in your workflow definition without specifying the agents responsible. This means that all of the possible agents of the single-step task are recipients of the work item. Since you occupy the position of both requester and head of department, you are automatically and immediately presented with the work item for approving the application. This is again because of advancing with immediate dialog.
As the superior, you have several options:

- Approve the request
- Reject the request
- Cancel processing of the request.

4. Choose to cancel processing.
   The start_workflow (Test Environment) screen is then displayed again.
   Exit this dialog and, if necessary, the Workflow Builder.

Reporting and analysis: Workflow outbox
In the first test of your workflow, you were introduced at this point to work item analysis. Now, you will be introduced to the workflow outbox.
Amongst other things, the work items which you started in dialog along with date, time and current status are displayed in the workflow outbox.

2. In the tree displayed, choose *Outbox → Started workflows.*

3. Choose the arrow on the right next to the ☐ and then *Today only.*
   
   The work items for all the workflows you started today are then displayed. Note the statuses of the work items displayed in the Status column of the Business Workplace. Read the F1 help for the Status column.

4. Position the cursor on the work item for the workflow just started. This work item has the status *in process.*
   
   From the current workflow data, you can see that the step *Create notification of absence* has been completed successfully, when that was, and who processed it. The work item has the status *completed.*
   
   You can also see that the work item representing the task for checking the request still has status *ready* and can therefore still be seen in the superior’s Business Workplace.

**Business Workplace - check notification of absence**

You now once again occupy the position of the superior who canceled processing earlier when the application was to be checked.

5. In the tree, choose *Inbox → Workflow.*
   
   The work items for you to process are displayed with their work item texts and certain other attributes.
   
   A preview of the selected work item is displayed below the list and contains the description of the work item.

6. Select the work item for checking the notification of absence and choose ☑.
   
   You go to the work item display. There you find amongst other things the description text, also with replaced text variables.

7. Choose ☑ to return to the Business Workplace.

8. Choose ☑ or start execution by double-clicking the entry.

9. Either reject or approve the request. You should start the workflow twice and test each variant.

To check whether the workflow was completed correctly, go to your workflow outbox. The procedure is described above.

**Result**

Your workflow definition has now been tested and you can continue with the next unit.
Unit 9: Adding a User Decision

Use
In the previous units, you created a workflow whose definition contains two activities. These activities are based on two standard tasks that you created.
In this unit, you will add a user decision [Extern] to your workflow definition.
With the user decision, the requester can decide to revise and resubmit the notification of absence if the head of department rejects it.
For further information, refer to Maintaining a User Decision [Seite 1074] in the SAP Business Workflow documentation.

Procedure
You start the Workflow Builder and open your workflow definition in Change mode. Proceed as described in Unit 7: Including "Check Notification of Absence" in the Workflow Definition [Seite 1609].

Creating a user decision as a step in the workflow definition
1. Choose under step type. The cursor changes shape.
2. Position the cursor on the Undefined step that follows the outcome Rejected and click.
   By doing so, you add a user decision to your workflow definition. The step definition is shown in the right-hand part of the screen. The Decision tab is displayed.

Entering basic data for user decision
3. Enter Your request was rejected by &. Revise? in the Title field.
   This text is used as the title for the alternative decisions when the workflow is executed. The variable & is a placeholder for a parameter that is filled from the task container at runtime.
4. Choose the F4 input help in the Parameter 1 field.
   The Expression for first parameter of user decision dialog box is then displayed.
5. Choose the Approver container element by double-clicking it.

Entering alternative decisions
6. Define the alternative decisions. Enter the following texts:

<table>
<thead>
<tr>
<th>Decision texts</th>
<th>Name</th>
</tr>
</thead>
<tbody>
<tr>
<td>Decision: Revise request?</td>
<td>Revise</td>
</tr>
<tr>
<td>Decision: Withdraw request?</td>
<td>Withdraw</td>
</tr>
</tbody>
</table>

The decision texts you enter here appear as decision options on the screen that can be processed by the requester after their leave request has been rejected.
The descriptions are the terms used to describe the outcomes in the workflow definition.

Enter the agents responsible
7. Choose Workflow initiator (expression) under Agents.
   The system enters the expression &_WF_INITIATOR& in the Expression field and in the input field beside the checkbox.
Concluding step definition

1. Choose to check the step definition.
2. Choose to exit the step definition.
3. Choose to view the entire workflow definition in the workflow area on the screen.

You can see that your workflow definition contains a user decision in the rejected branch. This has two outcomes whose branches converge again in the rejected branch. Each new branch contains an undefined step.

4. Choose

The workflow definition is checked, saved, and, provided it does not contain any errors, activated. The system displays a message if problems are encountered during the test.

Result

You can now subject your workflow definition to another test.
Unit 10: 3rd Test

Use
You start your workflow in dialog and three steps are executed: Two activities, for creating and checking a notification of absence, and one user decision.

Procedure

Starting the workflow and filling out the form, checking the notification of absence, user decision

8. If you are still in the Workflow Builder, choose ▶️.
   The Start Workflow (Test Environment) screen is then displayed. The number of your workflow is entered.
   If you are no longer in the Workflow Builder, choose Tools → SAP Business Workflow → Development → Runtime tools → Start workflow (test environment). Enter the number of your workflow or use the F4 input help.

9. Choose ✅.
   You go to the screen Create Notification of Absence.

From the requester’s perspective

10. Fill out the form and choose ✅.
    When you save the form, you (as superior) can immediately approve or reject the application (reason: advancing with immediate dialog).

From the superior’s perspective

11. Do not approve the application.

12. Choose ✅.
    If the request is not approved, the next step in the workflow definition is the user decision. The requester was entered as the agent for the step. The decision is therefore offered to you again as the requester (reason: advancing with immediate dialog).

From the requester’s perspective

You are given three options:
- Revise request?
- Withdraw request?
- Cancel

It is irrelevant which option you choose since you have not defined any follow-up steps. You should start the workflow several times and try out the different options. The complete process flow can be checked each time in the work item analysis or workflow outbox.
Unit 11: Define and Include "Revise Notification of Absence" in the Workflow Definition

Use
You will now add the activity Revise notification of absence to your workflow definition. You will create the necessary task within the workflow definition as in unit 4.

Procedure
You start the Workflow Builder and open your workflow definition in Change mode. Proceed as described in Unit 7: Including "Check Notification of Absence" in the Workflow Definition [Seite 1609].

Creating an activity as a step in the workflow definition
1. Choose under step type. The cursor changes shape.
2. Position the cursor on the Undefined step that follows the outcome Revise after the user decision, and click.

   By doing so, you add an activity to your workflow definition. The step definition is shown in the right-hand part of the screen. You are on the tab page Control.

Creating a standard task for use in this step
1. Choose the arrow on the right next to the and then Create task.

   The Basic data tab in the Standard task: Create screen is then displayed.

2. Enter the following data.

   Abbreviation: <ini_revis>

   Name: <Revise notif. of absence (ini)>

   Work item text: Revise notification of absence

   Object type: FORMABSENC

   Method: UPDATE

   You can also use the F4 input help to select the method.

3. Save your standard task as a local object.
4. Select Additional data → Agent assignment → Maintain
5. Position the cursor on the name of your standard task.
6. Choose Properties…
7. Set the General task indicator.
8. Choose Copy and then .

   The screen for editing the standard task is then displayed.
9. Choose and then .

   You then return to the step definition in the Workflow Builder. For this task, a binding must be defined between the workflow and task containers. The system proposes a
Unit 11: Define and Include “Revise Notification of Absence” in the Workflow Definition

binding and displays it for you to check in the Define container elements and binding dialog box. This proposal defines that the object reference to the notification of absence is transported from the workflow container (container element AbsenceForm) to the task container (container element _WI_Object_ID).

10. Confirm the proposal with ✔.

The system has entered the abbreviation for the new standard task in the Task field. The Step description field contains a description of the standard task. You can change this entry if necessary.

11. Switch to the Outcomes tab and enter a description (for example request revised) for the Task executed synchronously outcome.

Entering the agents responsible


The system enters the expression &_WF_INITIATOR& in the Expression field and in the input field beside the checkbox.

By doing so, you choose the agent responsible [Extern].

Concluding step definition

11. Choose ✔ to check the step definition.

12. Choose ✔ to exit the step definition.

13. Choose ✈️ to view the entire workflow definition in the workflow area on the screen.

You will notice that your workflow definition now contains a new step (the activity just created). The undefined step is also still there.


The workflow definition is checked, saved, and, provided it does not contain any errors, activated. The system displays a message if problems are encountered during the test.

Result

You can now subject your workflow definition to another test. Carry out the test as described in Unit 10: Third Test [Seite 1617].

Test proposal

To test the new step, proceed in the following order:

1. Create notification of absence.
2. Reject application
3. Revise user decision.
4. Revise request (the new step)

Since you are both the requester and approver and because advancing with immediate dialog is activated, all of the steps are presented to you directly.
Unit 11: Define and Include “Revise Notification of Absence” in the Workflow Definition
Unit 12: Integrating the UNTIL Loop for Reapproval

Use
If the requester decides to revise and resubmit the notification of absence to their superior, the step Check notification of absence must be executed again. There are different ways of solving this problem. The solution outlined here is only one example.

- You create a container element in the workflow container that you use as a flag. This flag contains different values, depending on the status of the notification of absence:
  - Approved
  - Not approved and revised
  - Not approved and not revised

To assign values to the container element, container operations must be executed on the workflow container.

For information on the workflow log, see Definition of the Workflow Container [Seite 1213] in the SAP Business Workflow documentation.

- You use this container element to define the condition in the UNTIL loop [Extern].
  For further information, refer to Maintaining an UNTIL loop [Seite 1087] in the SAP Business Workflow documentation.

Procedure
You start the Workflow Builder and open your workflow definition in Change mode. Proceed as described in Unit 7: Including "Check Notification of Absence" in the Workflow Definition [Seite 1609].

Create container element in the workflow container
1. Choose the entry Double-click to create by double-clicking in the Workflow Container.
   The dialog box for entering a container element is displayed.

2. Make the following entries:
   
   - Element: Flag
   - Name: Flag
   - Description: Flag for approval status
   - Reference table: SYST
   - Reference field: INPUT

3. Choose ✓.
   You have now added a local container element to the workflow container, which is ready to store a flag, based on its data type reference. However, you have not yet determined a value for this container element.

Integrating the UNTIL loop as a step
10. Choose 🕵️‍♂️ under step type. The cursor changes shape.
11. Move the cursor to the *Undefined* step before the event workflow end and click.

   By doing so, you add an *UNTIL loop* to your workflow definition. The step definition is shown in the right-hand part of the screen.

**Entering the basic data**

12. Enter *Resubmission required?* in the *Step description* field.

13. Click on the condition area to open the condition editor.

   The bottom half of the screen contains the empty condition, whereas the top half shows the system fields as well as the content of the workflow container. The field in the *Expression 1* column is ready for input and marked 📒.

14. Choose the *Flag* container element as the first expression by double-clicking it.

   The first expression is now entered in the condition in the lower part of the screen. The *Expression 2* field is now ready for input and marked 📒.

15. Choose 📒 as the *operator*.

16. Enter the constant *X* in the field for *Expression 2* and confirm by pressing *Enter*.

17. Choose 📒.

   You have now defined the condition *Flag = X*.

   For further information, see [Condition Editor][Seite 1012].

   The comparison between the container element *Flag* and the constant *X* returns either the result *true* or the result *false*. Two outcomes are therefore possible.

18. Enter the following texts for the outcomes:

   - **True**: *Flag equal to X - no resubmission*
   - **False**: *Flag not equal to X - resubmission*

   If the result of the evaluated condition is *false*, the loop is processed again. If the notification of absence is to be resubmitted to the superior, the container element *flag* must not have the value *X*. For this purpose, you will insert [container operations [Extern]] in the workflow definition in the next unit.

19. Choose 📒.

   You have now inserted an isolated UNTIL loop into the workflow definition, which does not yet contain any steps. You must change the workflow definition so that all of the steps that are required to check and resubmit the notification of absence are within the UNTIL loop.

   In order to understand the steps required, you must familiarize yourself with block operations.

**Block operations**

1. Choose 📒 to display the entire workflow definition in the workflow area on the screen.

2. Display the block structure of your workflow definition. Choose *Graphic → Blocks → Show*.

   📒

   The workflow definition is block-oriented. Every block represents a basic structural element that is a self-contained and consistent arrangement of steps and outcomes.
If you, for example, create a new step that has one outcome, this pair (step, outcome) represents a block. Operations that are performed on one step (delete, cut,..) always affect the entire block associated with that step.

In the above example, in which the block consists of the pair (step, outcome), using the delete function will also delete the corresponding outcome as well.

You should test the operations listed under Edit → Block on blocks:

- First save your workflow definition.
- Test the various block operations.

You can undo changes at any time by choosing 🔄. Another possibility is to exit the Workflow Builder without saving your work and to call it again.

For further information, refer to Block Orientation in the Workflow Definition [Extern].

### Copying steps to the UNTIL loop

The step Check notification of absence, which follows the outcome Request completed, is the first step in a new block. To implement the resubmission, the workflow must send the notification of absence to this step after it has been revised by the requester. You have used an UNTIL loop for this purpose. The step Check notification of absence must be located within this loop. In order to do so, first cut the block that begins with this step. Then paste the cut block in the UNTIL loop.

1. Select the step Check notification of absence.
2. Choose 🕒 to cut the block.
3. Select the Undefined outcome in the UNTIL loop and choose 🕒.
4. Choose 🕒 to align the graphic.
5. Choose 🕒 to view the entire workflow definition in the workflow area on the screen.
   
   You will notice that your workflow definition now contains steps within the UNTIL loop.
6. Choose 🕒.

The workflow definition is checked, saved, and, provided it does not contain any errors, activated. The system displays a message if problems are encountered during the test.

### Result

The integration of the UNTIL loop is now complete. The workflow up to this point, however, delivers an incorrect result, since the Flag container element is not assigned a value during execution. The result of the evaluated condition in the UNTIL loop is always false and the workflow can never be completed. To ensure that the workflow is executed correctly, you will add further container operations to the workflow definition in the next unit.
Unit 13: Inserting Container Operations

Use
You have added an UNTIL loop to your workflow definition. To ensure that this loop condition is executed correctly, you have to add container operations [Extern] to your workflow definition that change the value of the flag container element in accordance with how the workflow is executed.

Prerequisites
You have added an UNTIL loop to your workflow definition.

Procedure
You start the Workflow Builder and open your workflow definition in Change mode. Proceed as described in Unit 7: Including "Check Notification of Absence" in the Workflow Definition [Seite 1609].

Integrating container operations as a step
You can manipulate the individual elements of the workflow container using a container operation type step.
You need to insert a container operation at three different points in your workflow definition.
- After the outcome Request revised
  At this point, you assign the workflow container element Flag the value Z. The comparison in the UNTIL loop then returns the result false. In this case, the workflow continues via the outcome Flag does not have value X and the request is resubmitted to the superior’s Business Workplace.
- After the outcome Withdraw
  At this point, you assign the workflow container element Flag the value X. The comparison in the UNTIL loop returns the result true. The loop is exited.
- After the outcome Approved:
  You assign the container element Flag the value X. The comparison in the UNTIL loop returns the result true. The loop is exited.

Container operation after the outcome revised event
1. Choose under step type. The cursor changes shape.
2. Position the cursor on the Undefined step that follows the outcome Request revised and click.
   By doing so, you add a container operation to your workflow definition. The step definition is shown in the right-hand part of the screen.
3. Enter the following basic data:
   Step name: Set flag to Z
   Outcome name: Flag = Z
4. Specify the following for the operation:
   Result element: Flag (select using F4 input help)
   Expression: Z
The input fields for the operator and second expression must be empty.

When this step is executed, the Flag container element in the workflow container receives the value \( Z \). This value remains as long as it is not changed explicitly.

5. Choose ✔.

**Container operation after the Withdraw outcome**

1. Choose ✔ under step type. The cursor changes shape.
2. Position the cursor on the Undefined step that follows the outcome Withdraw and click. By doing so, you add a container operation to your workflow definition. The step definition is shown in the right-hand part of the screen.
3. Enter the following basic data:
   - **Step name:** Set flag to X
   - **Outcome name:** Flag = X
4. Specify the following for the operation:
   - **Result element:** Flag (select using F4 input help)
   - **Expression:** X
      The input fields for the operator and second expression must be empty.
5. Choose ✔.

**Container operation after the Approved outcome**

Since this container operation is identical to the one created above, you can simply copy it.

1. Select the container operation Set flag to X after the Withdraw outcome and choose ✔.
2. Select the Undefined step that follows the Approved event and choose ✔.

**Concluding step definition**

Choose ✔. The workflow definition is checked, saved, and, provided it does not contain any errors, activated. The system displays a message if problems are encountered during the test.

**Result**

You have now extended the workflow definition in such a way that the results of the condition evaluated in the UNTIL loop are meaningful. You can now subject your workflow definition to another test. Carry out the test as described in Unit 10: Third Test [Seite 1617].
Unit 14: Changing Staff Assignments

Use
In the tests you have carried out so far you acted both as the requester and as the superior.
In this unit, you will change the organizational plan in such a way that the requester and superior are two
different R/3 users.

You can change the organizational plan without modifying the workflow definition.

Procedure
You enter a different user as the superior in the staff assignments of your organizational plan. Ask a
colleague whether they would like to take part in this test. Or have yourself a second user created.

1. Choose Tools → SAP Business Workflow → Development → Definition tools →
   Organizational Management → Organizational plan → Change.
   This takes you to the Change Organization and Staffing (Workflow) screen.

2. Search for your head of department position in the search area. The positions that were
   found are displayed in the selection area. Open yours by double-clicking on the entry in the
   change mode.

Deleting position holders
3. Select the row in the overview area in which the assignment to the user is entered.

4. Choose the arrow on the right next to the and then Assignment.
   The position of the department head is no longer occupied.

Reassigning position holders
5. Choose User in the search area and search for the user whom you want to assign to the
   position.
   All of the users that match the search criteria are displayed in the selection area.

6. Select the user whom you want to assign to the position and drag him or her to the line
   position: head of department (ini) in the overview area. Confirm the adaptations of the
   relationship period.

7. Choose .
   This completes the changes to the staff assignments. The two positions are held by
   different persons.

Result
You have changed the staff assignments by changing the assignment of users to a position. You can check
this by displaying the staff assignments. Proceed as described in Unit 1: Organizational Plan [Seite
1587].
You can also start with Unit 16: Fourth Test [Seite 1629] directly after this unit. The recipients of the
work item for the step Check notification of absence are then determined from the possible agents for the
single-step task. You assigned the job of the head of department as the possible agents of this task in Unit 6: Create task “Check Notification of Absence” [Seite 1606].
Unit 15: Using a Role

Use

In the organizational plan you have created up to now, there are two positions, each of which is described by a job. Each position must be linked to a different user.

The job of department manager is linked to the department manager position. The job of department manager, in turn, is linked to the task Check notification of absence. In a more extensive organizational plan containing several positions for department managers, each position can be linked to the job of department manager.

These links ensure that each user who is assigned to a position as department manager is one of the possible agents of the task Check notification of absence. If you want to assign all of the department managers as possible agents of a different task, all you have to do is link this task to the job of department manager. You do not need to make any detailed changes.

You can add a further organizational unit to your organizational plan by creating a new position as department manager and linking it to a third user. When you link this new position to the existing job of department manager, the third user is automatically added to the list of possible agents for the task Check notification absence.

If you then create a notification of absence, it is offered to all of the possible agents for processing. This is clearly undesirable, since the notification of absence should only be presented to your own superior. This superior is the head of department who has the chief position in the organizational unit to which you too belong.

To ensure that the notification of absence is not submitted to all of the possible agents, you must define agents responsible in your step definition. All of the agents responsible who are also possible agents for this task then become recipients of the work item for the task. In this example, the agents responsible must be defined dynamically, since the relevant agent depends on the requester (person initiating the workflow). To do so, you use a role.

The role Superior of is shipped as standard with the R/3 System. This role first establishes a user's position and the relevant organizational unit, and then finds the chief position in this organizational unit.

This role only works correctly if you are not your own superior. You should only work through this unit if you have completed Unit 14: Changing Staff Assignments and adapted your organizational plan accordingly.

By using roles to assign responsibilities, you do not need to have any specific knowledge of the organizational plan when you define a workflow definition. When the role is resolved at runtime, all of the necessary information is provided via the binding definition from the workflow container to the role container.

Procedure

You start the Workflow Builder and open your workflow definition in Change mode. Proceed as described in Unit 7: Including "Check Notification of Absence" in the Workflow Definition.

Selecting a role

1. Open the step definition for the activity Check notification of absence by double-clicking the activity icon in the workflow area on the screen.
2. Choose **Superior of workflow initiator** under Agents.

The system then enters the role ID (00000168) in the **Role** selection field and in the input field.

The binding between the workflow container and the role container is created automatically here. If you choose **Role** in the checkbox and enter the role in the input field using the F4 input help, you must define the binding manually.

**Concluding step definition**

5. Choose ![checkmark] to check the step definition.

6. Choose ![checkmark] to exit the step definition.


The workflow definition is checked, saved, and, provided it does not contain any errors, activated. The system displays a message if problems are encountered during the test.

**Result**

You can now subject your workflow definition to another test.
Unit 16: Fourth Test

Use
In the organizational plan that you changed in Unit 14: Adjusting the Staff Assignments [Seite 1625], your colleague has now assumed the tasks of the superior. As the requester, you must start the workflow. The work item for checking the request will then be sent to the Business Workplace of the superior.

Procedure

Procedure as requester
   All of the workflows that you are allowed to start are listed on the left-hand side of the screen. A description of the selected workflow is shown on the right.
2. Select your workflow and choose Start.
   The work item for creating the notification of absence is proposed to you immediately for processing.
3. Fill out the form and choose 📝.
   You have now submitted your request and your superior must decide whether to approve it. Since you are not now the recipient of the next work item, advancing with immediate dialog does not have any effect.

Procedure as superior
1. Choose Office → Workplace to start the Business Workplace.
2. Open the folder Inbox → Workflow.
   The request awaiting approval is displayed as a work item in this folder.
3. Select the work item.
   The task description of the single-step task is displayed in the work item preview. As you can see, the expressions used in the task description have been replaced with the current content.
4. Choose 📝 but do not approve the request.
5. Choose 📝.
   You are now in your Business Workplace. The work item you have just executed is no longer displayed in the list.

Procedure as requester
6. Choose Office → Workplace to start the Business Workplace.
1. Open the folder Inbox → Workflow.
   This folder contains a work item awaiting a user decision.
7. Select the work item and choose 📝.
8. Select Revise application in the user decision dialog box.
Since the flag *advancing with dialog* is set in the step definition for the activity *Revise notification of absence*, the request is immediately presented to you for revision. If this flag is not set, you must execute the work item from your Business Workplace.

9. Change the request and choose \[button\] and \[button\].

You are now in your Business Workplace. The work item you have just executed is no longer displayed in the list.

**Procedure as superior**

If you have not closed your Business Workplace, choose \[button\] to refresh the work item display. You have received the request from the requester again. They can proceed as above to approve the request or reject it again. This workflow does not have a modeled response to a situation in which the requester and superior do not reach agreement. The workflow definition can be easily extended for this situation (for example, automatic notification of a third person if the request is exchanged between the employee and head of department several times).

**Reporting and analysis with the workflow log**

You should start the workflow several times together with your colleague and try out the alternatives. These alternatives can then be analyzed via work item analysis or the workflow outbox.

Use the workflow log in this unit. If the workflow you want to analyze is displayed in the work item analysis or in the workflow outbox, select the entry and choose \[button\]. The workflow log provides information about all the stages of processing. This includes the following:

- Which steps were executed?
- Who was the agent?
- When was processing carried out?
- What was the result of the processing?

If the workflow log is displayed with ActiveX, you can view the various steps in the workflow in the form of a graphic. To do so, choose \[button\]. For further information on the workflow log, please refer to *Workflow Log: Standard View [Seite 1420]*, and on the ActiveX view under *Personal Settings [Seite 1395]*.
Unit 17: Including "Send Notification" into the Workflow Definition

Use

You have created a complete workflow with which a notification of absence can be created, revised and approved.
In this unit, you will insert a step in the workflow definition that sends a notification to the requester after the request has been approved.

Procedure

You start the Workflow Builder and open your workflow definition in Change mode. Proceed as described in Unit 7: Including "Check Notification of Absence" in the Workflow Definition [Seite 1609].
A separate step type is available for the sending of mails.

1. Select the outcome Approved of the step Check notification of absence and choose Create in the context menu.

   The dialog box Select step is displayed.

2. Choose Send mail.

   The step definition is then displayed. The input fields required for sending a mail are found on the tab page Mail.

3. Due to the fact that the mail is to be sent to the requester (the workflow initiator), you do not need to change the standard settings. Inspite of this, choose the entry Workflow initiator in the field Recipient type.

   When entering a recipient, you must first select a recipient type. The entries have the following meanings:

   - **Workflow initiator**: Choose this entry if only the workflow initiator is to receive a mail. The recipient type is automatically set to Organizational object and the expression &_WF_INITIATOR& is entered in the field Recipient. These entries are the standard settings.
   
   - **Organizational object**: You must enter an Organizational object [Extern] in the field Recipient or choose an expression that contains one or several organizational objects. Use the F4 input help when making the entries. When you enter an expression, the system automatically creates a container element in the task container and adds the binding definition.
   
   - **E-Mail address**: In the field Recipient, enter either an e-mail address directly or by using the F4 input help, select a container element from the workflow container that contains one or several Internet addresses at runtime. The corresponding expression is entered. The system automatically creates a container element in the task container and adds the binding definition.

4. Enter the basic data for the task:
Unit 17: Including "Send Notification" into the Workflow Definition

**Abbreviation:** <ini_nottxt>

**Name:** <Notification request accepted (ini)>

In the Create Object Directory Entry dialog box, choose Local object.

5. Enter the following text for the mail subject:

   <Your leave request was reserved>

6. Enter the following text that the system is to send to the requester if the request is approved:

   <Dear colleague,

   Your leave request number &AbsenceForm.Number& from &AbsenceForm.CreateDate& was approved on &AbsenceForm.ApprovDate& by &AbsenceForm.Approver.Name&.

   If you want to include an expression that is to be replaced at runtime, choose . Choose the expression from the container elements of the workflow container. The system automatically creates a container element in the task container and adds the binding definition.

7. Choose ✅.

**Activating workflow definition**

8. Choose ⏯.

The workflow definition is checked, saved, and, provided it does not contain any errors, activated. The system displays a message if problems are encountered during the test.

**Test**

Test your workflow definition in the usual way.

You receive a mail when your “superior” has approved your "notification of absence". You find the mail in your Business Workplace in the folder *Inbox → Unread documents*. 
Unit 18: Work Item Attachments

Use
The superior is to have the opportunity to add an attachment to their work item for checking the notification of absence. This attachment is a document (any SAPscript or PC document), which is then made available to all subsequent recipients in the workflow. The superior can use this to give a full explanation for a rejection.

An attachment can be added to a workflow before the workflow is actually executed in the Business Workplace.

To create an attachment after the workflow has been executed, you have to change the definition of the associated standard task.

To enable the superior to create an attachment after the notification of absence has been checked, you must set the Confirm end of processing flag for the task Check notification of absence. This property means that the agent has to confirm the end of processing of the work item explicitly. The agent can also add an attachment before this confirmation.

For further information about using documents in workflows, refer to Document Processing [Seite 1237] in the SAP Business Workflow documentation.

Procedure
You start the Workflow Builder and open your workflow definition in Change mode. Proceed as described in Unit 7: Including "Check Notification of Absence" in the Workflow Definition [Seite 1609].

Changing the task "check notification of absence"
1. Open the step definition for the step Check notification of absence by double-clicking it.
2. Double-click the task ID to open the task definition.
   You go to the screen Standard task: Display.
3. Choose to switch to change mode.
4. Set the Confirm end of processing flag under Execution.
5. Choose and then .

Concluding step definition
1. Choose to check the step definition.
2. Choose to exit the step definition.
3. Choose .

The workflow definition is checked, saved, and, provided it does not contain any errors, activated. The system displays a message if problems are encountered during the test.

Result
Execution of the work item for the step Check notification of absence must be confirmed expressly. This enables you to create an attachment after the request has been approved or rejected.

You can also create attachments while a work item is being executed. An example of this is provided in the demo workflows. Choose Tools → Business Workflow →
Development → Environment → Start demo workflows. Start the demo workflow Demo for WI execution with WF toolbox. Further information is provided on the right-hand side of the screen.
**Unit 19: Fifth Test**

**Use**
After you have executed the step *Check notification of absence*, you must confirm the end of processing.

**Procedure**

**Procedure as requester**
4. Choose *Office → Start Workflow*.
   
   All of the workflows that you are allowed to start are listed on the left-hand side of the screen. A description of the selected workflow is shown on the right.

5. Select your workflow and choose *Start*.
   
   The work item for creating the notification of absence is proposed to you immediately for processing.

6. Fill out the form and choose 📝.
   
   You have now submitted your request and your colleague must decide whether to approve it. Since you are not now the agent of the next activity, *advancing with immediate dialog* does not have any effect.

**Procedure as superior**
10. Choose *Office → Workplace* to start the Business Workplace.

11. Open the folder *Inbox → Workflow*.
   
   The request awaiting approval is displayed as a work item in this folder.

12. Select the work item.
   
   The task description of the task is displayed in the work item preview. As you can see, the expressions used in the task description have been replaced with the current content.

13. Choose 📝 but do not approve the request.

   
   The dialog box for confirming the end of processing appears.

15. Choose the arrow 📝 on the right next to the 📝 and then *Create attachment*.
   
   The *Create Document: Header* dialog box is then displayed.

16. Choose *RAW* as the *Type* and enter a *title*:
   
   *Title*: **Reason for rejecting the request**

17. Choose 📝.
   
   The SAPscript editor is then displayed.

18. Enter your reason and choose 📝. Choose 📝.
   
   You then return to the dialog box for *User confirmation* of end of processing. The attachment is displayed under *Objects and attachments*. 
19. Choose ✔ Complete work item.

You are now in your Business Workplace and the work item you have just processed has been removed from the worklist.

**Procedure as requester**

2. Choose Office → Workplace to start the Business Workplace.

3. Open the folder Inbox → Workflow.

   This folder contains a work item awaiting a user decision. The 📧 symbol in the Attachment column indicates that the work item has an attachment.

4. Select the work item.

   All of the attachments for this work item are displayed in the work item preview.

5. Choose the attachment in the work item preview.

   The attachment is then opened and displayed in a new screen.

6. Execute the work item.

**Further Procedure**

Continue processing the workflow and analyze it using the workflow log.
Unit 20: Monitoring Missed Deadlines

Use

When defining steps that require dialog with the user, you can instruct the workflow system to monitor certain deadlines. The workflow system can monitor the following types of deadline:

- Requested start
- Latest start
- Requested end
- Latest end

This unit explains how missed deadlines are usually handled: The standard system response to an exceeded deadline is to notify a recipient of the message for missed deadlines [Extern] by sending a work item for missed deadlines [Extern] to his or her Business Workplace.

For further information, please refer to Maintaining the Deadline Tabs [Seite 1081] in the SAP Business Workflow documentation.

In this unit, the workflow is to be extended so that the requester should be notified if the superior has not made the decision about the leave request after 10 minutes.

Procedure

You start the Workflow Builder and open your workflow definition in Change mode. Proceed as described in Unit 7: Including “Check Notification of Absence” in the Workflow Definition [Seite 1609].

Determining the recipient of the message for missed deadlines

1. Open the step definition for the step Check notification of absence by double-clicking it.
2. Go to the Latest end tab.
3. Choose the entry Work item creation as the reference time.

   The active deadline is indicated by the symbol in the tab index of the relevant tab.

4. Enter the value 10 in the field + and choose Minutes as the time unit.
5. On the Display text tab, choose the entry Workflow initiator (expression) as the recipient.

   As a result, the entry _WF_Initiator is automatically copied to the adjacent field and the entry in the checkbox changes to Expression.

   If a deadline is exceeded, the text visible in the display is automatically copied to the work item for missed deadlines. You will now change this text in the task definition.

6. Choose the highlighted text in the display to open the task definition.
7. Choose and go to the Description tab.
8. Choose the text type Text for latest end and then 📖.

   The SAPscript text editor is then launched.
9. Enter the following text:

   Your notification of absence no. & WI_OBJECT_ID.NUMBER& from & WI_OBJECT_ID.CREATEDATE& has not been approved yet.

   You should remind your superior.
If you want to insert an expression from the task container as a text variable, choose
Include → Expression. Then choose the relevant expression in the Please choose an
expression dialog box by double-clicking the Container node.

10. Choose  

11. Choose  and then  

**Concluding step definition**

9. Choose  to check the step definition.

10. Choose  to exit the step definition.

11. Choose  

The workflow definition is checked, saved, and, provided it does not contain any errors,
activated. The system displays a message if problems are encountered during the test.

**Test**

7. Choose Office → Start Workflow.

All of the workflows that you are allowed to start are listed on the left-hand side of the
screen. A description of the selected workflow is shown on the right.

8. Select your workflow and choose Start.

The work item for creating the notification of absence is proposed to you immediately for
processing.

9. Fill out the form and choose  

You have submitted your request and should now allow the deadline to expire.
Once the latest end passes, a work item for missed deadlines appears in your Business Workplace notifying
you of the missed deadline. This work item is a deadline monitoring work item, for which there is no
executable method. You can go from the display of this work item to the work item display of the late
work item.

**Notes on extensions**

The recipient of a message for missed deadlines is notified with a work item. The notification text is
stored in the task definition of the monitored task.
If you want to customize the notification, you can define you own Text for latest end on the Description tab
in the task definition for the single-step task Check notification of absence. This text can contain text
variables that are replaced at execution time.
Modeling Workflows

Purpose
This is a typical process flow for workflow modeling. It is only one way of working with the *SAP Business Workflow* definition tools. It also serves the purpose of introducing the various definition tools and their interrelationships.

Prerequisites
If you are adhering to the role-based procedure model in this documentation, the process consultant should have proposed a business process model.

An organizational plan must be created. This activity is also one of the tasks of the process consultant in our model.

Process Flow

The process consultant has created a business process model based on the business requirements and alternatives. This is now to be implemented by the workflow developer.

- **Business Workflow Explorer**
  
  Using the Business Workflow Explorer, you can get an overview of the existing tasks and workflows, and then edit them or use them as templates. By doing this, you can save yourself the effort of carrying out your own modeling.

  The Explorer covers not only the workflows and tasks you have created yourself, but also the workflows and tasks supplied by SAP, which are part of the workflow application...
scenarios. Both the process consultant and the workflow modeler should be familiar with the scenarios.

If you find an appropriate template, you can copy the template and adapt it to the specific characteristics of the business process model. If you only have to make changes to existing steps in a workflow, you should check whether you can create a workflow configuration [Seite 1171]. You also have the option of using the Workflow Wizard Explorer tool.

- **Workflow Wizard Explorer [Seite 977]**
  This tool can be used to generate particular workflow types (for example, workflows for approval procedures) using a wizard, and then edit them afterwards in task maintenance.

- **Business Object Repository (BOR) [Seite 1167]**
  If you cannot use any task as a template and there is no wizard suitable for your requirements, you proceed as follows:
  - You identify all business objects involved in your business process model.
  - You then check whether the relevant object types with their methods, attributes and events are defined in the Business Object Repository. The grouping of object types in the application component hierarchy and the option of searching generically for parts of a name help when looking for existing object types.
  - If you find an object type whose definition meets your requirements, you can use it without making any modifications.
  - If you find an object type whose definition does not quite meet your requirements, you add to its definition.
  - If you do not find a suitable object type, you define your own object type.

- **Task Maintenance [Seite 1173]**
  - You identify the tasks involved in the business process model. In doing this, you sort out which object method is to be executed with the task and which person or group of persons is responsible for it. You then define the tasks by specifying object type and method, and determine the possible agents.
  - To describe a business process that normally comprises several steps, you define a workflow.

- **Workflow Builder [Seite 1002]**
  The workflow definition is processed in the Workflow Builder. You can define the external relationships (call interface, triggering events) of the workflow, as well as the actual implementation of a business scenario as a sequence of connected steps in the Workflow Builder. You assign tasks or subworkflows to these steps.

**Result**
This results in the workflow created in the business process model being implemented in the R/3 System.
Demo and Test Workflows

Use
You use these workflows to test the workflow environment and familiarize yourself with certain functions. For the latter, you process the individual steps of the workflow in the Workflow Builder.

Features

Start demo workflows
When you choose this function, you go to the standard environment for starting workflows. The workflows listed demonstrate certain SAP Business Workflow functions, such as the user exit for the work item preview or the Workflow Toolbox.

The individual workflows are explained on the right-hand side of the screen.

For more examples, refer to Demo Examples for SAP Business Workflow.

Start test workflows
When you choose this function, you also go to the standard environment for starting workflows. You can use this workflow to test the workflow environment.

The individual workflows are explained on the right-hand side of the screen.

Demo: Embedded inbox
You can use this function to demonstrate the features of an embedded workflow inbox, workflow outbox or workflow resubmissions.

Demo: Fill out form
You can use this function to create a notification of absence. From a technical point of view, you execute the method Create of the object type Formabsenc. You can use this function, for example, if you have implemented a workflow that is started when a notification of absence is created.

Activities
You can access these functions by choosing Tools → Business Workflow → Development → Environment.
WebFlow

Purpose
WebFlow provides functions that enable SAP Business Workflows to be executed across the Internet. WebFlow makes it possible to:

- Send XML documents to other systems from a Business Workflow
- Start a Business Workflow on another SAP System from your SAP System
- React to the results of a Business Workflow executed in another system
- Start a Business Workflow when an appropriate XML document is received

Prerequisites
A Web server and an Internet Transaction Server (ITS) must be configured for the SAP System for the transmission and reception of the XML documents to be successful. For further information, refer to Defining the Web Server [Seite 1513].

Process Flow

Overview of the process flow of a Web activity

You can either send XML documents you have created yourself or have WebFlow create a Wf-XML document from your data. For further information, refer to Creation of a Wf-XML Document [Seite 1093].

To send XML or Wf-XML documents, you use the step type Web activity in a workflow. For further information, refer to Maintenance of a Web Activity [Seite 1037].

The transfer protocols http and https are used for the transmission. A transmission of this type is not completed until the other system has sent back an http reply.

A special service must be active for a Business Workflow to be started by an inbound Wf-XML document in a SAP System. This service processes the inbound Wf-XML document and writes the data...
contained into the import parameters of the Business Workflow to be started. The Business Workflow is then started and executed.

The following are possible scenarios:

**Sending an XML document to another system**

You can send an XML document that you created yourself from your SAP System A to system B using a WebFlow. The Web activity sends your XML document to the recipient you specified. If system B's http reply contains an XML document, it can be stored in the workflow container. The Business Workflow in system A is then continued.

**Sending a Wf-XML document to another system**

You can have the WebFlow create the XML document to be sent. A Wf-XML document is then sent to system B, which contains your data and the http reply is evaluated by the WebFlow. The Business Workflow in system A is then continued.

**Sending a Wf-XML document to another system with feedback**

This scenario is the same as the scenario without feedback up to reception of the http reply. The Web activity waits for a Wf-XML reply document. The Web activity writes the data in it into the workflow container and the Business Workflow in system A is continued.

**Sending a Wf-XML document to another SAP system and starting a Business Workflow**

If you want to start a Business Workflow in SAP System B using a WebFlow, have the system create a Wf-XML document for you from your data. You can use a wizard to generate the URL to which the Wf-XML document is sent. To ensure that the Business Workflow to be started in system B receives all the data required, you must know the interface [Intern] of the Business Workflow to be called in full. The http reply is evaluated by the WebFlow and the Business Workflow in System A is then continued.

**Sending a Wf-XML document to another SAP system and starting a Business Workflow with feedback**

This scenario is the same as the scenario without feedback up to reception of the http reply. Once the Business Workflow in system B is terminated, the export parameter data of the interface is sent back to the calling Business Workflow in system A as a Wf-XML reply document. The Web activity writes this data into the workflow container and the Business Workflow in system A is continued.

**Starting a Business Workflow with an inbound Wf-XML document without reply**

When a Wf-XML document arrives, the WebFlow tries to start the Business Workflow named in the document. The data contained in the Wf-XML document is written into the relevant import parameters of the interface. The WebFlow sends an http reply back.

**Starting a Business Workflow with an inbound Wf-XML document with reply**

This scenario is the same as the previous scenario without reply. Once the Business Workflow has been terminated the WebFlow creates a Wf-XML reply document. The export parameter data of the interface is written into this, and the document is sent back to the calling system.
Creation of a Wf-XML Document

Use
The data exchange with other systems is performed with an XML document in a WebFlow. The workflow system can generate an XML document automatically for the data to be transferred, which is in accordance with the Wf-XML transfer format of the Workflow Management Coalition in the beta version of 11 January 2000 with the following modification:
The operation CreateProcessInstance is used to start a Business Workflow.

Features
The following Wf-XML documents are created automatically:

Wf-XML documents created by a WebFlow

<table>
<thead>
<tr>
<th>SAP System A</th>
<th>Transmission with http or https</th>
<th>SAP System B</th>
</tr>
</thead>
<tbody>
<tr>
<td>...</td>
<td></td>
<td>...</td>
</tr>
<tr>
<td>Web activity</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Wf-XML document (1)
The import parameters from the task container are used in creation. These are derived from the interface [Extern] of the reference workflow of the Web activity [Extern].
The import parameter data of the task container is written into the Wf-XML document. The following content is transferred depending on the data type reference [Extern]:

<table>
<thead>
<tr>
<th>Data type reference</th>
<th>Content transferred</th>
</tr>
</thead>
<tbody>
<tr>
<td>ABAP Dictionary</td>
<td>All data</td>
</tr>
<tr>
<td>Object reference to XML_DOC</td>
<td>Referenced XML document itself is transferred in full</td>
</tr>
<tr>
<td>Other object reference</td>
<td>Not supported</td>
</tr>
</tbody>
</table>

Multiline import parameters are not supported.
Within the Wf-XML document the data from a container element is stored in separate areas denoted using `<container_element_ID> data </container_element_ID>`. The assignment of the data to the individual areas is made on the basis of name parity. When an SAP System receives a WF-XML document that is to start or continue a workflow, the data from the individual sections of the Wf-XML document is assigned to container elements in the workflow container on the basis of name parity as well. For an example of a generated Wf-XML document (1), refer to number 1 in Examples of Created Wf-XML Documents [Extern].

If the indicator Wait for feedback is set within the Web activity, the URL to which the Wf-XML reply document (3) from system B is to be sent is written into the Wf-XML document (1) under `<ObserverKey>`. This ensures that the Wf-XML reply document (3) can be evaluated by the Web activity. The Web server [Seite 1513] must be defined for the URL to be created.

http reply (2)

If SAP System B receives a Wf-XML document, it sends an http reply (2) in the form of an XML document. This confirms the start of the Business Workflow. For an example of an http reply (2), refer to number 2 Examples of Created Wf-XML Documents [Extern].

Wf-XML reply document (3)

If system A is waiting for feedback, SAP System B sends a Wf-XML reply document (3) after the Business Workflow is completed. The export parameters of the workflow are written in the relevant areas of the Wf-XML reply document (3) as with the import parameters when the call is made. For an example of a generated Wf-XML reply document (3), refer to number 3 Examples of Created Wf-XML Documents [Extern].

Activities

The system creates the Wf-XML document automatically if you chose transfer format 01 in the Web activity. You will find the XML documents created in the work item display of the Web work item [Extern].

If a Wf-XML document was to start a Business Workflow in your system but this has failed, you can find and display the Wf-XML document using the report Displaying XML Documents [Seite 1512].
Ad Hoc Workflow

Purpose
There are two types of ad hoc workflow:

Ad hoc agent assignment
A workflow of this type enables agents to be assigned to workflow steps when the workflow is started. During workflow execution the agent assignment can be changed by authorized users at any time for steps not yet completed.

Ad hoc definition
Once you have selected a work item in the Business Workplace, you can start workflows that use the application object of the work item. If the workflow required does not yet exist, you can create it ad hoc.

Prerequisites
To execute an ad hoc definition effectively, tasks groups must be assigned to the application object types. For further information, refer to Defining Task Groups for Ad Hoc Workflows [Seite 1099].

Process Flow

Ad hoc agent assignment
You can convert an existing workflow into an ad hoc workflow in the Workflow Builder. No responsible agents should be entered in the step definitions for all steps to which ad hoc users are to be assigned during execution. Container elements are created automatically by the system in the workflow container and the relevant task containers during conversion into an ad hoc workflow. An expression [Extern] is assigned to the input fields for responsible agents. For further information, refer to Defining Ad Hoc Agent Assignment [Seite 1096].
If this ad hoc workflow is started by a user, this workflow initiator defines the recipients [Extern] who are to execute the individual steps. Throughout the execution of the workflow the workflow initiator can change this assignment of recipients for steps not yet completed at any time. For further information, refer to Executing Workflows with Ad Hoc Agent Assignment [Seite 1097].

Ad hoc definition
If you select a work item in the Business Workplace, you can choose Environment → Start Workflows. This displays a dialog box for starting workflows, in which only the workflows that work with the application object of the selected work item are offered. Here you can define a workflow ad hoc if the workflow you require does not exist yet. The Workflow Builder (with restricted functionality) can be used to define the ad hoc workflow. Tasks are available in the tray My workflows and tasks, which you can use in the workflow definition. For further information, refer to Defining Workflows Ad Hoc [Seite 1098].
Defining Ad Hoc Agent Assignment

Use
You can define ad hoc agent assignment for a workflow. This enables the user who starts the workflow to assign a recipient to all relevant steps at the start of execution. This user can also change the assignment of recipients for steps that have not yet been completed at any time during execution of the workflow.

Prerequisites
You have defined the workflow for which you want to enable ad hoc agent assignment in the Workflow Builder.

Procedure
You are in the Workflow Builder. The workflow is opened in change mode.
1. Identify all the steps for which you want to enable ad hoc agent assignment.
2. Delete the responsible agents [Extern] in the respective step definitions.
3. Choose Extras → Ad hoc functions → Enable ad hoc agent assignment.
   The system adds container elements required for the ad hoc agent assignment to the workflow container. An expression [Extern] is assigned automatically to the input fields for the responsible agents of a step as well.
   
   If you want to deactivate ad hoc agent assignment for particular steps, replace the expression for the responsible agents with an entry. If you want to integrate steps added subsequently into the ad hoc agent assignment, carry out step 3 again.

Result
Your workflow has been changed making ad hoc agent assignment possible. Test this by choosing Extras → Ad hoc functions → Test ad hoc agent assignment.
Executing Workflows with Ad Hoc Agent Assignment

Use
The ad hoc agent assignment is performed at the start of the workflow. The user who started the workflow can perform an ad hoc agent assignment for steps that have not yet been completed at any time while a workflow is running.

Procedure

Ad hoc agent assignment at the start of a workflow
   This displays the screen Start Workflow.
2. Select the workflow to be started.
   If this is an ad hoc workflow, input fields are displayed for agents in the lower part of the preview area.
3. Assign the relevant users to the individual steps.

Changing the ad hoc agent assignment while a workflow is running

Only the user who started the workflow can change the assignment during execution.
1. Open the workflow outbox in your Business Workplace.
2. Select the workflow for which you want to change the ad hoc agent assignment.
3. Select to change the agent assignment.
   This displays the dialog box Ad hoc change of workflow agents. If a step has already been completed, the actual agent [Extern] is displayed. For steps that have not yet been completed, the assigned recipients are displayed.
4. Change the agent assignment.

Result
The workflow is executed with the new agent assignment. If you change the agent assignment for a step that is already being executed, the work item is forwarded from the old to the new actual agent.
Defining Task Groups for Ad Hoc Workflows

Use
If you want to define a workflow for an application object of a work item ad hoc, an associated task group must be defined. Only certain tasks that already exist are to be used as activities in the workflow definition. These tasks must all be based on the object type of the application object. You collect these tasks in a task group and assign this to the object type.

A workflow is to be defined ad hoc for a work item that has a notification of absence (object type FORMABSENC) as application object. The following are tasks that deal with a notification of absence in your system:

- Approve notification of absence
- Process notification of absence
- Display notification of absence
- Delete notification of absence

Users who are allowed to create workflows ad hoc are only to use the tasks Process notification of absence and Display notification of absence. You therefore create a task group containing only the tasks Process notification of absence and Display notification of absence, and assign this task group to the object type FORMABSENC.

Procedure
To create a task group containing appropriate tasks for a particular object type, proceed as follows:

1. Determine the tasks and workflows that refer to the object type in question. You can use the Business Workflow Explorer [Seite 975] for this.
2. Create a new task group. For further information, refer to Definition of a Task Group [Seite 1208].
3. Transfer all the tasks and workflows that the user who is to create workflows ad hoc is to be able to use in their workflows into the task group.
4. Assign this task group to the object type. To do this, go to the Workflow Builder and choose Extras → Ad hoc functions → Link object and task group.

Result
When a workflow is created ad hoc, the task group appropriate to the object type appears in the tray My workflows and tasks in the Workflow Builder.
Defining Workflows Ad Hoc

Use
You can define a workflow ad hoc to be executed with the application object of a selected work item.

Prerequisites
A task group must be defined to make tasks and workflows with which you can define a workflow ad hoc available. This task group must contain all tasks and workflows that you can use for the definition of the workflow. For further information, refer to Defining Task Groups for Ad Hoc Workflows [Seite 1099].

Procedure
1. To define a workflow ad hoc you must select a work item in the Business Workplace, which has an application object, and choose Environment → Start Workflow.

   This displays the dialog box Start Workflow. All workflows that can be started immediately are listed on the left.

2. Select Create workflow to define a workflow that uses the application object.

   This displays the Workflow Builder for creating an ad hoc workflow. The task group defined for the application object type is displayed in the tray My workflows and tasks.

3. Define the workflow.

   Use the tasks available in the tray My workflows and tasks. In the workflow definition select the position at which the task is to be inserted. Then select the task in the tray and Choose in the context menu. In the step type area there is a number of step types from the Workflow Builder available, which you can use in your workflow. For further information about the step types available and their uses, refer to Step Types and Their Symbols [Seite 1005].

4. Save your ad hoc workflow. Give your workflow an abbreviation and a name when doing so.

   You should always save the workflow as a local object. The workflow system administrator can assign the workflow to a development class at a later date as well.

5. Exit the Workflow Builder.

   This displays the dialog box Start Workflow again.

Result
Your newly created workflow is now in the table of startable workflows and is already selected. You can now start it directly.
Business Workflow Explorer

Use
You use the Business Workflow Explorer [Extern] to display an overview of all tasks for a selected search range. All the tasks displayed can also be processed from the Business Workflow Explorer.

Prerequisites
You have to define a search range when you first call the Business Workflow Explorer. When the Business Workflow Explorer is called thereafter, the last active search range is used. The Business Workflow Explorer is only available on a 32-bit platform.

Features
In the left-hand screen area, the Business Workflow Explorer displays all the tasks assigned to the selected search range.
In the right-hand screen area, the Business Workflow Explorer displays the workflow (with associated tasks) or the task, which you selected in the left-hand screen area.

Activities
You can call the Business Workflow Explorer by choosing Tools → Business Workflow → Development → Definition tools → Business Workflow Explorer.

Task-related functions
The functions available on the tasks displayed are shown in the context menu.

Quick view
With tasks an overview of the most important properties is displayed and with workflows the workflow definition is displayed.

Display or change
A new session is opened to display or change.

Create
You can create new tasks and workflows. If you have selected a workflow you create a new workflow template, and if you have selected a task you create a new standard task.

Workflows of a definition
You go to the work item selection in which the task ID is already entered. Further information on carrying out the selection, refer to Work Item Selection [Seite 1490].

Where-used list
All workflow definitions in which the task is used are displayed.

Choose another search range
This enables you to change the Business Workflow Explorer search range. The search range can be limited according to specific criteria. These criteria are:

- Task groups [Extern]
- Application components
- Your own or all tasks and workflows saved locally
- Tasks and workflows with the SAPphone property
- The last ten workflows processed
- Workflows defined as demo workflows or as verification and test workflows.
- Tasks that refer to a particular object type and possibly a particular method

If you define a new search range, it replaces the last active range.

**Search / extended search**
You can use this to search for tasks and workflows. The standard search function only searches for the term in your current search range. The extended search covers all existing tasks and workflows.

**Copy**
You select a task or a workflow in the right-hand tree and choose. The selected task/workflow is copied to form a new task/workflow. You only see the new task/workflow in the current session of the Business Workflow Explorer as well if it is in the search range displayed.

If you copy a task of the type T (customer task) or a workflow of the type WF (workflow task), the copy will be defined as a standard task or workflow template since customer tasks and workflow tasks will not be supported in the future.

**Detailed view**
You can activate a detailed view in the right-hand tree by choosing View → Details. This displays attributes of the task or workflow. The business object type used is also displayed with tasks.
Workflow Wizard Explorer

Use
You use the Workflow Wizard Explorer [Extern] to:

- Find out about the Workflow Wizards available
- Define a workflow with a Workflow Wizard

Workflows created with a Workflow Wizard are never complete and are only a framework. To create a workflow that meets your requirements you must use the Workflow Builder [Seite 1002] to edit the workflow generated.

Integration
New Workflow Wizards can only be created in the Workflow Wizard Repository [Extern]. To display the technical content of the Workflow Wizard, select it and choose ☑.

Workflow Wizard Explorer as part of the Workflow Builder
The Workflow Wizard Explorer can be used to execute the individual Workflow Wizards from the Workflow Builder. A workflow definition created with a Workflow Wizard is integrated directly into the workflow definition.

Workflow Wizard Explorer as a separate tool
The Workflow Wizard Explorer can be called as a separate tool from which you can start Workflow Wizards.

Features
A Workflow Wizard supports you in the definition of a workflow with a dialog sequence. Before the Workflow Wizard starts its work, you must make certain settings. The dialog box for these settings is displayed automatically when you execute a Workflow Wizard. This dialog box varies depending on whether you call the Workflow Wizard Explorer via the Workflow Builder or as a separate transaction.

Workflow Wizard Explorer when called in the Workflow Builder
The following fields and settings need to be maintained:

- Insert workflow definition only, do not create workflow template
  The steps created by the Workflow Wizard are inserted into the workflow definition at the selected position in accordance with the positioning rules [Extern].

- Insert workflow definition and create workflow template
  The steps created by the Workflow Wizard are inserted into the workflow definition at the selected position in accordance with the positioning rules [Extern]. In addition, a new workflow is created, which contains the workflow definition generated.

  The workflow definition created with the Workflow Wizard can be used again via its task.

Workflow Wizard Explorer when called as a separate tool
The following fields and settings need to be maintained:

- Extended settings: Activate workflow definition
  If this indicator is set, the workflow definition is activated after generation.
• Extended settings: Classify task as **general task** [Extern]
  
    If this indicator is not set, you have to assign possible agents to the task.

**Documentation on Workflow Wizards**

You can go to this documentation from every Workflow Wizard. To display the documentation, select in the line of the Workflow Wizard in question.

**Activities**

**Workflow Wizard Explorer in Workflow Builder**

To call the *Workflow Wizard Explorer* in the *Workflow Builder*, select a step or an outcome and choose *Wizards* → *Workflow Wizard Explorer*.

The part of the workflow definition created by the Workflow Wizard is inserted into the existing workflow definition at the selected position in accordance with the **positioning rules** [Extern].

**Workflow Wizard Explorer as a separate tool**

To call the Workflow Wizard Explorer as a separate tool, choose *Tools* → *Business Workflow* → *Development* → *Definition Tools* → *Wizards* → *Workflow Wizard Explorer*. You have the opportunity to continue any Workflow Wizards whose execution was aborted.

**Starting Workflow Wizards**

To start a Workflow Wizard, select in the line of the Workflow Wizard in question.
Workflow Wizards for Approval Procedures

Use
This folder contains Workflow Wizards that define workflows for various approval procedures. The approval procedures vary in their complexity.

Deadline monitoring for an approval procedure created must then be defined manually in the Workflow Builder.

All approval procedures have the same evaluation part. For further information, refer to Evaluation Part in All Approval Workflows [Extern].

Integration
The approval task [Extern] you use in the Workflow Wizard must reference the method Approval. The method is inherited from the interface Approval [Seite 1112]. Each approval object type should support the interface Approval and implement the method Approval.

For an extended approval, you use an approval task that references the method XApproval. Implement this method from the interface Extended Approval [Seite 1113].

Features
The following Workflow Wizards define approval procedures:

<table>
<thead>
<tr>
<th>Workflow Wizard</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Hierarchical Approval Workflow [Seite 986]</td>
<td>This Workflow Wizard maps the basic form of an approval procedure. Several recipients specified in the definition decide one after the other whether the approval object should be approved or rejected.</td>
</tr>
<tr>
<td>Dynamic Hierarchical Approval Workflow [Seite 980]</td>
<td>This Workflow Wizard implements a hierarchical approval procedure for a particular object type and with a particular approval task. The levels and the recipients are not determined until runtime.</td>
</tr>
<tr>
<td>Extended Hierarchical Approval Workflow [Seite 982]</td>
<td>This Workflow Wizard allows not only rejection or approval on every level, but also proposed rejection or approval.</td>
</tr>
</tbody>
</table>
| Release Subworkflow for FI (FIPP, BSEG) [Seite 984]| The Workflow Wizard is specially adapted for use as a subworkflow in the workflows FIPP_FRAME* and BSEG_FRAME*.
These workflows from the Financial Accounting area are distinguished by the fact that the actual approval procedures are implemented as subworkflows that are determined dynamically at runtime. The Workflow Wizard helps to provide these subworkflows. |
| Parallel Approval Workflow [Seite 988]| This Workflow Wizard creates an approval procedure in which the recipients decide to approve or reject the approval object at the same time. An approval often does not require agreement from all recipients. Agreement from a particular number of recipients specified in the definition often suffices (agreement of n from m recipients). |
Dynamic Hierarchical Approval Workflow

Use
This Workflow Wizard creates a dynamic, hierarchical approval workflow. During execution of the Workflow Wizard you determine the type of the object to be approved and the approval task [Extern] to be used for the approval.
You do not determine the extent of the approval procedure and the recipients approving at each level until the workflow is executed. When you do this you can assign a user, a position [Extern], a job [Extern], an organizational unit [Extern] or a work center at each level. Once a recipient does not approve the object, the recipients higher in the hierarchy do not receive the object for approval.

Prerequisites
The object type of the objects to be approved must have an approval method and should also have a method for displaying the objects, which is defined as the default method. The Workflow Wizard can insert this default method into the approval step as a secondary method [Extern]. All users who are to be able to carry out approvals in this workflow must be possible agents [Extern] of the approval task.

Features
The workflow created by the Workflow Wizard has three parts:
1. The users who are to approve the object are established.
2. Each approver gets the opportunity to approve in turn in a loop. If someone rejects, the loop is ended and no further approvals are sought.
3. The approval status is evaluated and the applicant has an opportunity to revise the object if it was rejected.
The workflow initiator establishes the approvers from the possible agents of the approval task. When the Workflow Wizard is executed you can specify whether the workflow initiator can be one of the approvers as well. When you choose the approvers you must ensure that the users who approve the object first are lowest in the hierarchy.
All approvers receive the object for approval in their respective Business Workplace in a loop. Once an approver does not approve the object, execution of the loop is aborted. If the object is approved, the approval status is changed.
The approval status is checked in the evaluation part of the workflow. Only once all approvals have been made is the approval result set to approved. The workflow initiator receives an e-mail stating that the object has been approved. Otherwise, the workflow initiator receives a work item in their Business Workplace. They must decide whether the object is to be revised or deleted. The approval result is set accordingly.

Activities
After execution of the Workflow Wizard you still have to add steps to the workflow, in which the object that is to be approved is created or established. This object must be available in the container element ApprovalObject.
When the Workflow Wizard is executed, the following container elements are created in the workflow container:

<table>
<thead>
<tr>
<th>Description (technical name)</th>
<th>Explanation</th>
</tr>
</thead>
<tbody>
<tr>
<td>Approval object (ApprovalObject)</td>
<td>Reference to the object to be approved. The object to be approved and the approval task are determined when the Workflow Wizard is executed.</td>
</tr>
<tr>
<td>List of all possible agents (AllAgents)</td>
<td>This element is no longer required.</td>
</tr>
<tr>
<td>List of selected agents</td>
<td>The responsible agents [Extern] of the approval task, who are selected</td>
</tr>
</tbody>
</table>

April 2001
### Dynamic Hierarchical Approval Workflow

<table>
<thead>
<tr>
<th>(ApprovalAgents)</th>
<th>by the workflow initiator from the possible agents of the approval task, are stored in this multiline container element.</th>
</tr>
</thead>
<tbody>
<tr>
<td>Approval status (ApprovalState)</td>
<td>The status is initialized at the number of approvers in a background step, and decreased by one with each approval. If the status is zero on evaluation, the object was approved. If it is greater than zero, the object was rejected.</td>
</tr>
<tr>
<td>Table index (Index)</td>
<td>This container element is used for index access to the table of selected agents.</td>
</tr>
<tr>
<td>Current agent (CurrentAgent)</td>
<td>The user who processes the work item is saved in this container element.</td>
</tr>
<tr>
<td>List of excluded agents (ExcludedAgents)</td>
<td>The list of agents contains all users, positions [Extern], jobs [Extern], organizational units [Extern] and work centers that have been involved in the approval process. This list is used to exclude users who have already granted an approval from approving the same object again. You can also include the workflow initiator in this list. This prevents applicants from being able to approve their own requests.</td>
</tr>
<tr>
<td>Rejection indicator (RejectedFlag)</td>
<td>This indicator is set in the event of a rejection.</td>
</tr>
<tr>
<td>Approval result (Result)</td>
<td>This field contains the result of the approval procedure after the evaluation part has been executed: A for approved, D for delete, E for edit</td>
</tr>
</tbody>
</table>
Extended Hierarchical Approval Workflow

Use
Several users are involved at each approval level in an extended, hierarchical, approval procedure. Only some of them are authorized to actually approve or reject the object. The other users are only authorized to propose that the object be approved or rejected.
Each approval level can only be left when the object has actually been approved or rejected. To be proposed for approval or rejection is not sufficient to leave the approval level. A user who has granted an approval, or proposed an approval or rejection, is excluded from further processing in the same approval procedure.
If the object is approved on an approval level, the next approval level is called. You define the number of approval levels when executing the Workflow Wizard.
Once one user does not approve the object, no further approval levels are initiated.
You can specify whether the same approval task is to be used for all approval levels when the Workflow Wizard is executed.

Prerequisites
The object type of the objects to be approved must have an extended approval method and should also have a method for displaying the objects, which is defined as the default method. The Workflow Wizard can insert this default method into the approval step as a secondary method. All users who are to be able to carry out approvals in this workflow must be possible agents of the approval task.

Features
With the Workflow Wizard for extended hierarchical approvals, you can create approval procedures of any nesting depth.
In the first step of any approval procedure, the approval status (container element ApprovalState) is first initialized at one.
This initialization is followed by the approval steps nested in each other, all of which have at least four possible results (approval, rejection, proposed approval, proposed rejection). In each approval step, the approval method of the object to be approved is executed and, optionally, its default method as a secondary method.
In the event of approval, the approval status is increased by one and the next approval level initiated.
In the event of rejection, the workflow goes immediately to its evaluation part. If the approval procedure is terminated in this way, the approval status is always less than or equal to the number of approval levels.
If the approval or rejection is only proposed by an agent due to their restricted authorization, the approval procedure remains at this approval level. The agent is included in the list of excluded agents. A work item is generated again, and the object is submitted for approval again.
For further information, refer to Evaluation Part of All Approval Workflows.

Enhancements to the workflow definition
The workflow definition created by the Workflow Wizard can only be used practically as part of a superordinate workflow definition. Steps still need to be added to it in which the object that is to be approved is created or established.
A reference to this object must be stored in the container element ApprovalObject in the workflow container.

Activities
An extended hierarchical approval procedure created by a wizard also requires additional container elements in the workflow container:
## Extended Hierarchical Approval Workflow

<table>
<thead>
<tr>
<th>Data type</th>
<th>Description (technical name)</th>
<th>Explanation</th>
</tr>
</thead>
<tbody>
<tr>
<td>Object reference</td>
<td>Approval object (ApprovalObject)</td>
<td>Reference to the object to be approved. The object to be approved and the approval task must be determined at definition time.</td>
</tr>
<tr>
<td>Address object (AddressObjects)</td>
<td></td>
<td>Reference to the address objects of those who are informed of the approval.</td>
</tr>
<tr>
<td>ABAP Dictionary reference</td>
<td>Approval status (ApprovalState)</td>
<td>Status of the approval procedure. The status is initialized at one at the start, and increased by one with each approval. If the status is greater than the number of approval levels on evaluation, the object was approved.</td>
</tr>
<tr>
<td>Current agent (CurrentAgent)</td>
<td></td>
<td>The user who processes the work item is saved in this container element.</td>
</tr>
<tr>
<td>List of excluded agents (ExcludedAgents)</td>
<td></td>
<td>The list of agents contains all users, positions [Extern], jobs [Extern], organizational units [Extern] and work centers that have been involved in the approval process. This list is used to exclude agents who have already granted an approval from approving the same object again. You can also include the workflow initiator in this list. This prevents applicants from being able to approve their own requests.</td>
</tr>
<tr>
<td>Decision indicator (DecisionFlag)</td>
<td></td>
<td>This indicator is set when a decision is made (approval or rejection).</td>
</tr>
<tr>
<td>Rejection indicator (RejectedFlag)</td>
<td></td>
<td>This indicator is set in the event of a rejection.</td>
</tr>
<tr>
<td>Approval result (Result)</td>
<td></td>
<td>This container element holds the result of the approval procedure after the workflow. In the case of detailed evaluation, there are the following values: A for approved D for delete E for edit In the case of the minimum evaluation, there are the following values: 0 for approved 4 for not approved</td>
</tr>
</tbody>
</table>
Release Subworkflow for FI (FIPP, BSEG)

Use
The workflow definition created is specially adapted for use as a subworkflow in the workflows FIPP_FRAME* and BSEG_FRAME*. These workflows from the Financial Accounting area are distinguished by the fact that the actual approval procedures are implemented as subworkflows that are determined dynamically at runtime. The Workflow Wizard helps to provide these subworkflows.

In a hierarchical approval workflow, the type of object to be approved, the approval task [Extern], the levels and the specific agents for each level must be specified at definition time. The approval procedure is otherwise such that every agent can reject the object to be approved with the consequence that the agents following know nothing of the approval procedure started, because it has already broken down at a lower level in the hierarchy.

Features
In the first step of every release procedure, the release status (element Level) is first initialized at one. This initialization is followed by the approval steps nested in each other, all of which have at least two possible results (approved or rejected). The approval method of the object to be approved is executed and, optionally, its default method as a secondary method. This execution takes place asynchronously in a new session without influencing the execution of the workflow.

In the event of approval, the approval status is increased by one and the next approval step initiated. In the event of rejection, nothing happens and the workflow goes immediately to its evaluation part. If the approval procedure is terminated in this way, the approval status is always less than the number of levels.

Enhancements to the workflow definition
The workflow definition created by the Workflow Wizard can only be used practically as part of a superordinate workflow definition. Steps still need to be added to it in which the object that is to be approved is created or established.

A reference to this object must be stored in the container element ApprovalObject in the workflow container.

Activities
A hierarchical approval procedure created by a wizard requires a few additional elements in the workflow container:

<table>
<thead>
<tr>
<th>Data type</th>
<th>Short description (name)</th>
<th>Explanation</th>
</tr>
</thead>
<tbody>
<tr>
<td>Object reference</td>
<td>Approval object (ApprovalObject)</td>
<td>Reference to the object to be approved.</td>
</tr>
<tr>
<td>ABAP Dictionary</td>
<td>Release status (Level)</td>
<td>Status of the approval procedure. The status is initialized at one at the start, and increased by one with each approval. If the status is equal to zero on evaluation, the object was approved. If it is greater than zero, the object was rejected.</td>
</tr>
<tr>
<td></td>
<td>Current agent (CurrentAgent)</td>
<td>The user who processes the work item is saved in this container element.</td>
</tr>
<tr>
<td></td>
<td>List of excluded agents</td>
<td>The list of agents contains all users, positions [Extern], jobs [Extern], organizational units [Extern] and work centers which are involved in the approval process.</td>
</tr>
</tbody>
</table>

April 2001 300
<table>
<thead>
<tr>
<th></th>
<th>This list is used to exclude agents who have already granted an approval from approving the same object again. You can also include the workflow initiator in this list. Thus, you prevent applicants from being able to approve their own requests.</th>
</tr>
</thead>
<tbody>
<tr>
<td>Release result (Released)</td>
<td>This field contains the result of the approval procedure after the workflow: X for approved ' ' for not approved</td>
</tr>
</tbody>
</table>
Hierarchical Approval Workflow

Use
In a hierarchical approval procedure, one agent must agree before the next agent can decide. Each agent has the option of rejecting the request therefore terminating the entire procedure. Since the workflow can be terminated by any agent, it is very important that the hierarchy be respected (lower positions first, higher positions last), because it would not make sense for a managing director's decision to be nullified by a head of department, for example. This logical sequence cannot be checked by the wizard, so must be defined correctly by the user.

An administrator needs a new PC and enters a purchase order starting a workflow for approval.

First their head of department must agree, and then a designated agent from the IT department must grant their approval. Lastly an employee from Controlling must grant their approval to finally release the purchase order.

Another possibility is that the approvers are on the same hierarchical level but can belong to different departments. The basic principle is otherwise the same.

First a Financial Accounting clerk (from several possible) must agree and then a Controlling administrator (from several possible). It may also make sense, within the double verification principle, for two or more agents from the same department to carry out an approval.

In a hierarchical approval workflow, the type of object to be approved, the approval task [Extern], the levels and the specific agents for each level must be specified at definition time.

Prerequisites
The object types that are to use an automatically created approval procedure must have an approval method. It is best if these object types support the interface Approval [Seite 1112], which incorporates a method for approval and whose interface is ideal for the Workflow Wizard. The object type should also always have a method for displaying objects, which should be entered as default method, because the Workflow Wizard integrates the default method of the object type to be approved into the approval step as a secondary method during instantiation.

Features
The Workflow Wizard for hierarchical approvals can be used to create approval procedures with any nesting level.
In the first step of every approval procedure, the approval status (container element ApprovalState) is first initialized with the number of hierarchical levels, that is the number of approvals to be granted. This initialization is followed by the approval steps nested in each other, all of which have at least two possible results (approved or rejected). The approval method of the object to be approved is executed and, optionally, its default method as a secondary method. This execution takes place asynchronously in a new session without influencing the execution of the workflow.
In the event of approval, the approval status is decreased by one and the next approval step initiated. Once all agents have agreed, this is therefore zero.
In the event of rejection, nothing happens and the workflow goes immediately to its evaluation part. If the approval procedure is terminated in this way, the approval status is always greater than zero.
Hierarchical Approval Workflow

For further information, refer to Evaluation Part of All Approval Workflows [Extern].

Extensions to the workflow definition

The workflow definition created by the Workflow Wizard can only be used practically as part of a superordinate workflow definition. Steps still need to be added to it in which the object that is to be approved is created or established.

A reference to this object must be stored in the container element ApprovalObject in the workflow container.

Activities

A hierarchical approval procedure created by a wizard requires certain additional elements in the workflow container, which are created by the workflow system:

<table>
<thead>
<tr>
<th>Data type</th>
<th>Description (technical name)</th>
<th>Explanation</th>
</tr>
</thead>
<tbody>
<tr>
<td>Object reference</td>
<td>Approval object (ApprovalObject)</td>
<td>Reference to the object to be approved. The object to be approved and the approval task must be determined at definition time.</td>
</tr>
<tr>
<td>List of addresses</td>
<td>AddressObjects</td>
<td>Reference to the address objects of those who are informed of the approval.</td>
</tr>
<tr>
<td>ABAP Dictionary reference</td>
<td>Approval status (ApprovalState)</td>
<td>Status of the approval procedure. The status is initialized at the start at the number of approval levels, and decreased by one with each approval. If the status is zero on evaluation, the object was approved. If it is greater than zero, the object was rejected.</td>
</tr>
<tr>
<td>Current agent</td>
<td>CurrentAgent</td>
<td>The user who processes the work item is saved in this container element.</td>
</tr>
<tr>
<td>List of excluded agents</td>
<td>ExcludedAgents</td>
<td>The list of agents contains all users, positions [Extern], jobs [Extern], organizational units [Extern] and work centers that have been involved in the approval process. This list is used to exclude agents who have already granted an approval from approving the same object again. You can also include the workflow initiator in this list. This prevents applicants from being able to approve their own requests.</td>
</tr>
<tr>
<td>Approval result</td>
<td>Result</td>
<td>This field contains the result of the approval procedure after the workflow: A for approved D for delete E for edit</td>
</tr>
</tbody>
</table>
Parallel Approval Workflow

Use

This workflow implements majority decisions of the form \( n \) from \( m \). A total of \( m \) users receive a work item with the activity for approval at the same time. They can also agree or reject at the same time.

If the necessary number of \( n \) agreements required for an approval is reached, the object is regarded as having been approved.

The object is regarded as rejected when the number of users rejecting it renders it impossible for the necessary number of agreements to be reached.

A 2 from 4 approval procedure is implemented for a purchase order. A total of 4 recipients receive a work item for approval at the same time.

If 2 recipients agree, the purchase order is approved. Once 3 have rejected, the purchase order is rejected.

With this variant \( n \) (necessary agreements) from \( m \) (total number of recipients) must agree. To implement this, the Workflow Wizard creates a workflow with parallel processing with \( m \) branches in which the approvals are carried out.

The approval procedure is terminated once approval has been granted in \( n \) branches, or once the number of branches in which the object has been rejected renders it impossible for the number of necessary agreements to be reached.

For a parallel approval workflow, the type of the object to be approved, the approval task [Extern] and the recipients for each branch must also be specified at definition time.

Prerequisites

The object types that are to use an automatically created approval procedure must have an approval method. It is best if these object types support the interface Approval [Seite 1112], which incorporates a method for approval and whose interface is ideal for the Workflow Wizard.

The object type should always, however, have a method for displaying objects, which should be entered as default method, because the Workflow Wizard integrates the default method of the object type to be approved into the approval step as a secondary method during instantiation.

Features

In the first steps, the two elements in the workflow container are initialized with the number of necessary approvals (container element Approvals) or the number of necessary rejections (container element Rejections).

The workflow definition then contains a fork that has a branch for each agent who is to receive the approval task. Each approval task can have Approved or Rejected as a result.

In the event of approval, the approval counter (Approvals) is decreased by one and checked to see whether it has reached zero.

In the event of rejection, the rejection counter (Rejections) is decreased by one and checked to see whether it has reached zero.

If one of the check routines ascertains that the approval or rejection counter is less than or equal to zero, the fork is terminated and the evaluation step is implemented.

For further information, refer to Evaluation Part of All Approval Workflows [Extern].
**Enhancements to the workflow definition**

The workflow definition created by the Workflow Wizard can only be used practically as part of a superordinate workflow definition. Steps still need to be added to it in which the object that is to be approved is created or established.

A reference to this object must be stored in the container element `ApprovalObject` in the workflow container.

**Activities**

The system adds the following container elements to the workflow container:

<table>
<thead>
<tr>
<th>Data type</th>
<th>Description (technical name)</th>
<th>Explanation</th>
</tr>
</thead>
<tbody>
<tr>
<td>Object reference</td>
<td>Approval object (ApprovalObject)</td>
<td>Reference to the object to be approved. The object to be approved and the approval task must be determined at definition time.</td>
</tr>
<tr>
<td></td>
<td>Address object (AddressObjects)</td>
<td>Reference to the address objects of those who are informed of the approval.</td>
</tr>
<tr>
<td>ABAP Dictionary reference</td>
<td>Number of approvals (Approvals)</td>
<td>This container element counts the agreements. If the necessary number of agreements is reached, the workflow is set to <em>approved</em>.</td>
</tr>
<tr>
<td></td>
<td>Number of rejections (Rejections)</td>
<td>This container element counts the rejections. If the necessary number of rejections is reached, the workflow is set to <em>rejected</em>.</td>
</tr>
<tr>
<td></td>
<td>Exit indicator (Exit)</td>
<td>This indicator exits the fork if it is set to X.</td>
</tr>
<tr>
<td></td>
<td>List of (excluded) agents (Agents)</td>
<td>The list of agents contains all users, positions [Extern], jobs [Extern], organizational units [Extern] and work centers that have been involved in the approval process. This list is used to exclude agents who have already granted an approval from approving the same object again. You can also include the workflow initiator in this list. This prevents applicants from being able to approve their own requests.</td>
</tr>
<tr>
<td></td>
<td>Approval result (Result)</td>
<td>This field contains the result of the approval procedure after the workflow: A for <em>approved</em> D for <em>delete</em> E for <em>edit</em></td>
</tr>
</tbody>
</table>
Workflow Wizards for Circulars

Use
This folder contains Workflow Wizards which define workflows for the forwarding of any documents (circulars). This enables you to ensure that the circular reaches all recipients and does not remain for too long with the individual recipients.

Features
All workflow definitions for circulars have the following properties in common:

- The object type of the circular is determined, for example:
  - *Office document* (object type SOFM)
  - *Parked document* (object type FIPP)
  - *Notification of absence* (object type FORMABSEN)
  - *Material* (Object type BUS2001)

- The processing task (and therefore the method) is determined, with which the circular is displayed or processed.

The workflow definitions for circulars can be differentiated as follows:

- Static or dynamic
  - With the static variant, all recipients of the circular are determined in the workflow definition. This specification then applies to all workflows which refer to this definition.
  - With the dynamic variant, the recipients of the circular are not specified individually for the respective current workflow until runtime.

- Parallel or sequential
  - With the parallel variant, all recipients can access the circular at the same time.
  - With the sequential variant, the circular is presented to the recipients for display purposes in sequence. Only when one recipient has finished, does the next recipient get the opportunity to access the circular. With the help of the Workflow Wizard, you can define deadline monitoring so that the circular is forwarded after a specific period of time automatically or the workflow initiator is notified.

The following combinations result:

<table>
<thead>
<tr>
<th></th>
<th>Static</th>
<th>Dynamic</th>
</tr>
</thead>
<tbody>
<tr>
<td>Parallel</td>
<td>Parallel Circular [Seite 997]</td>
<td>Dynamic Parallel Circular [Seite 995]</td>
</tr>
<tr>
<td>Sequential</td>
<td>Circular [Seite 991]</td>
<td>Dynamic Circular [Seite 993]</td>
</tr>
</tbody>
</table>
Circular

Use

A user can send a specific circular (for example a Microsoft Word document or any business document) sequentially to several defined users in the R/3 System, who are to view the circular and make comments if applicable.

In workflows defined with this wizard, you specify all the recipients of the circular in the workflow definition.

If you do not want to define the recipients of the circular until runtime, you must use one of the Workflow Wizards for dynamic circulars.

The workflow definition created with the Workflow Wizard contains the recipient of the circular as a defined "distribution list". This should only contain positions [Extern], but it can also contain users or organizational units [Extern].

The user who starts the workflow (workflow initiator) passes the object for circulation at runtime in an initial value assignment or in an initial workflow step.

Prerequisites

The workflow definition created by the Workflow Wizard can only be executed without postprocessing if the method used by the processing task for displaying or changing the circular does not use parameters but rather only the object reference. Otherwise you must revise the binding definition.

Features

The current recipient receives a work item in their Business Workplace. If they execute the work item, the circular is displayed or can be changed.

You can define a latest end for this work item, after which the workflow initiator is either notified by means of an e-mail or the workflow with the next step is continued.

The recipients of the circular can add attachments and notes using the standard functions in the Business Workplace which are available both to the recipients of the subsequent steps and to the workflow initiator.

The number of recipients cannot be changed at runtime. Any changes that need to be made to the agents can be made using the organizational plan (changes to the staffing assignments).

The workflow initiator receives the circular at the end as the final recipient.

Enhancements to the workflow definition

You must amend the workflow definition created by the Workflow Wizard by adding steps, in which the circular is created or determined.

Activities

The system adds the following container elements to the workflow container:

<table>
<thead>
<tr>
<th>Data type</th>
<th>Description (ID)</th>
<th>Explanation</th>
</tr>
</thead>
<tbody>
<tr>
<td>Object reference</td>
<td>Object name from object type definition</td>
<td>Object reference to the circular.</td>
</tr>
<tr>
<td>ABAP Dictionary reference</td>
<td>List of agents (CircularAgents)</td>
<td>Multiline container element in which all recipients are stored.</td>
</tr>
<tr>
<td>reference</td>
<td>(CircularAgents) recipients are stored.</td>
<td></td>
</tr>
</tbody>
</table>
Dynamic Circular

Use
A user can send a specific circular (for example a Microsoft Word document or any business document) sequentially to a number of users not yet defined at definition time in the R/3 System, who are to view the circular and make comments if applicable.
A dialog box is displayed in which the user who starts the workflow (workflow initiator) can determine the recipient of the circular.
The workflow initiator passes the object for circulation at runtime in an initial value assignment or in an initial workflow step.

Prerequisites
The workflow definition created by the Workflow Wizard can only be executed without postprocessing if the method used by the processing task for displaying or changing the circular does not use parameters but rather only the object reference. Otherwise you must revise the binding definition.

Features
The list of all users authorized to change/display the circular from an organizational point of view is displayed to the workflow initiator. This organizational authorization is derived from the assignment of the possible agents [Extern] for the processing task. The workflow initiator selects the recipient to whom the circular is to be sent. The workflow initiator cannot specify an order.

The object is distributed to the agents using the task Selection of agent for subsequent task (TS30200146) supplied by SAP. This task is classified in automatic Customizing as a general task [Extern].
The current recipient receives a work item in their Business Workplace. If they execute the work item, the circular is displayed or can be changed. For this, a loop (UNTIL) [Extern] is defined in the workflow definition, in which the next recipient is read from the table of all recipients (CircularAgents).
You can define a latest end for this work item, after which the workflow initiator is either notified by means of an e-mail or the workflow with the next step is continued.
The recipients of the circular can add attachments and notes using the standard functions in the Business Workplace [Seite 1368] which are available both to the recipients of the subsequent steps and to the workflow initiator.

The number of recipients cannot be changed at runtime. Any changes that need to be made to the agents can be made using the organizational plan (changes to the staffing assignments).
The condition section of the UNTIL loop checks whether additional recipients are contained in the table. The workflow initiator receives the circular at the end as the final recipient.

Enhancements to the workflow definition
You must amend the workflow definition created by the Workflow Wizard by adding steps, in which the circular is created or determined.

Activities
The system adds the following container elements to the workflow container:
<table>
<thead>
<tr>
<th>Data type</th>
<th>Short description (ID)</th>
<th>Explanation</th>
</tr>
</thead>
<tbody>
<tr>
<td>Object reference</td>
<td>Object name from object type definition</td>
<td>Object reference to the circular.</td>
</tr>
<tr>
<td>ABAP Dictionary</td>
<td>List of all possible agents (AllAgents)</td>
<td>This container element is no longer required.</td>
</tr>
<tr>
<td>reference</td>
<td></td>
<td></td>
</tr>
<tr>
<td>List of agents</td>
<td>List of agents (CircularAgents)</td>
<td>Multiline container element in which all recipients are stored.</td>
</tr>
<tr>
<td>(CircularAgents)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Recipient</td>
<td>Recipient (Current_Agent)</td>
<td>The user who processes the work item is saved in this container element.</td>
</tr>
<tr>
<td>(Current_Agent)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Index (Index)</td>
<td>Index for access to the list of recipients (CircularAgents)</td>
<td></td>
</tr>
</tbody>
</table>
Dynamic Parallel Circular

Use
A user can send a specific circular (for example a Microsoft Word document or any business document) at the same time to a number of users not yet defined at definition time in the R/3 System, who are to view the circular and make comments if applicable.

A dialog box is displayed in which the user who starts the workflow (workflow initiator) can determine the recipient of the circular.

The workflow initiator passes the object for circulation at runtime in an initial value assignment or in an initial workflow step.

Prerequisites
The workflow definition created by the Workflow Wizard can only be executed without postprocessing if the method used by the processing task for displaying or changing the circular does not use parameters but rather only the object reference. Otherwise you must revise the binding definition.

Features
The list of all users authorized to change/display the circular from an organizational point of view is displayed to the workflow initiator. This organizational authorization is derived from the assignment of the possible agents [Extern] for the processing task. The workflow initiator selects the recipient to whom the circular is to be sent. The workflow initiator cannot specify an order.

The object is distributed to the agents using the task Selection of agent for subsequent task (TS30200146) supplied by SAP. This task is classified in automatic Customizing as a general task [Extern].

All recipients receive a work item in their Business Workplace at the same time. If you execute the work item, the circular is displayed or can be changed. The Table-Driven Dynamic Parallel Processing [Extern] is used which contains all the recipients in the table CircularAgents.

You can define a latest end for this work item, after which the workflow initiator is either notified by means of an e-mail or the workflow with the next step is continued.

The recipients of the circular can add attachments and notes using the standard functions in the Business Workplace [Seite 1368] which are available both to the recipients of the subsequent steps and to the workflow initiator.

The number of recipients cannot be changed at runtime. Any changes that need to be made to the agents can be made using the organizational plan (changes to the staffing assignments).

The condition section of the UNTIL loop checks whether additional recipients are contained in the table. The workflow initiator receives the circular at the end as the final recipient.

Enhancements to the workflow definition
You must amend the workflow definition created by the Workflow Wizard by adding steps, in which the circular is created or determined.

Activities
The system adds the following container elements to the workflow container:
### Dynamic Parallel Circular

<table>
<thead>
<tr>
<th>Data type</th>
<th>Short description (ID)</th>
<th>Explanation</th>
</tr>
</thead>
<tbody>
<tr>
<td>Object reference</td>
<td>Object name from object type definition</td>
<td>Object reference to the circular.</td>
</tr>
<tr>
<td>ABAP Dictionary reference</td>
<td>List of all possible agents (AllAgents)</td>
<td>This container element is no longer required.</td>
</tr>
<tr>
<td>List of agents (CircularAgents)</td>
<td>Multiline container element in which all recipients are stored.</td>
<td></td>
</tr>
<tr>
<td>Recipient (Current_Agent)</td>
<td>The user who processes the work item is saved in this container element.</td>
<td></td>
</tr>
<tr>
<td>Index (Index)</td>
<td>Index for access to the list of recipients (CircularAgents)</td>
<td></td>
</tr>
</tbody>
</table>
Parallel Circular

Use
A user can send a specific circular (for example a Microsoft Word document or any business document) at the same time to several defined users in the R/3 System, who are to view the circular and make comments if applicable.

In workflows defined with this wizard, you specify all the recipients of the circular in the workflow definition.

If you do not want to define the addressees of the circular until runtime, you must use one of the Workflow Wizards for dynamic circulars.

A workflow definition created with this Workflow Wizard contains the readers of the circular as a defined "distribution list". It is better if this only contains positions, but it can also contain users or organizational units.

The workflow initiator transfers the object for circulation at runtime in an initial value assignment or in an initial workflow step, and leaves the rest to the workflow.

Prerequisites
The workflow definition created by the Workflow Wizard can only be executed without postprocessing if the method used by the processing task for displaying or changing the circular does not use parameters but rather only the object reference. Otherwise you must revise the binding definition.

Features
All recipients receive a work item in their Business Workplace at the same time. If you execute the work item, the circular is displayed or can be changed. For this, a fork [Extern] is used.

You can define a latest end for this work item, after which the workflow initiator is either notified by means of an e-mail or the workflow with the next step is continued.

The recipients of the circular can add attachments and notes using the standard functions in the Business Workplace [Seite 1368] which are available both to the recipients of the subsequent steps and to the workflow initiator.

The number of recipients cannot be changed at runtime. Any changes that need to be made to the agents can be made using the organizational plan (changes to the staffing assignments).

The workflow initiator receives the circular at the end as the final recipient.

Enhancements to the workflow definition
You must amend the workflow definition created by the Workflow Wizard by adding steps, in which the circular is created or determined.

Activities
The system adds the following container elements to the workflow container:

<table>
<thead>
<tr>
<th>Data type</th>
<th>Short description</th>
<th>Explanation</th>
</tr>
</thead>
<tbody>
<tr>
<td>Object reference</td>
<td>Object name from object type definition</td>
<td>Object reference to the circular.</td>
</tr>
<tr>
<td>ABAP Dictionary reference</td>
<td>List of agents (CircularAgents)</td>
<td>Multiline container element in which all recipients are stored.</td>
</tr>
</tbody>
</table>
Other Workflow Wizards

Use
This folder contains Workflow Wizards which help to simplify workflow modeling.

Features
The following Workflow Wizards are available for simplified workflow modeling:

- Creation of Customizing workflows [Seite 1000]
- Simplified workflow definition [Seite 1001]
Creation of Customizing Workflows

Use
Customizing activities usually require a number of views [Extern] or view clusters [Extern] to be maintained in a particular order. So that view maintenance does not have to be called for each view or view cluster, all views/view clusters that are to be maintained together in one context should be processed within one workflow definition.

Features
To create the workflow definition, the Workflow Wizard asks for the views to be maintained. In the workflow definition, an object reference is first created from the view name to an object of type VIEWDATA for each view to be processed. This object reference is stored in the workflow container in a container element with the same name as the view. A step for maintaining this view is then created. The individual maintenance dialogs are processed sequentially. The Customizing navigator provides information between the individual maintenance dialogs on the views that have already been maintained or are still to be maintained.
Simplified Workflow Definition

Use
In many cases, workflow definitions must be created in which only a single object is processed in some form in several sequential or parallel steps. The possible operations on this object are available in the form of methods and the single-step tasks derived from them.

Prerequisites
To use an object type in a simplified workflow definition without encountering problems, you should make sure during definition of the object types that the methods only require the object reference and not additional method parameters, since these cannot always be created using automatic binding.

Features
The Workflow Wizard first requests the object type on whose object the operations of the workflow definition are to be performed. The system automatically creates a container element with an object reference to the requested type in the workflow container. Then you choose the tasks that are to be executed in workflow. Agent selection can be performed. The system caters for all appropriate object references of the tasks in the binding. All other input and output parameters required and container elements in the workflow container are created by the system.

You must check the generated workflow definition in the Workflow Builder.
The tasks are arranged in sequential steps as standard. It is possible to group individual steps in a fork. The forks cannot be nested in each other. Other modeling elements, such as loops, are not supported.

Activities
All tasks of the selected object type are displayed in the left-hand part of the screen. Using the pushbuttons Add and Add all, the selected tasks can be transferred to the field on the right-hand side. Cut and Paste can be used to define the sequence in which the single-step tasks are to be executed. Several tasks can be grouped together in a fork with Select block. Identical numbers then indicate this association.
The pushbuttons Agent assignment and Task details take you to task processing.
Workflow Configuration

Use
To adapt an SAP Workflow, you can create a configuration. In this configuration you can redefine values for every step of the workflow definition. These values are evaluated at runtime instead of the values originally defined.

Features
You can set the following data individually in each step definition:

- Responsible agents [Extern]
- Excluded agents [Extern]
- Message recipient for completion [Extern]
- Priority [Extern]
- Requested start [Extern]
- Indicator denoting whether the step is included into the workflow log
- Activation of a latest end, a latest start or a requested end with the reaction *Send mail*
- Reference date/time for latest end, latest start and requested end
- Message recipient for missed deadline [Extern]
- Information about the work item display

Activities
You start workflow configuration by choosing *Tools → Business Workflow → Development → Definition Tools → Workflow Configuration*. This displays the screen *Process Workflow Configuration for 'workflow name'*. Using a workflow configuration is like using the *Workflow Builder [Seite 1002]*. Double-clicking on a step in the workflow definition displays the step definition, where you can enter values. These values are then used at runtime instead of the values defined in the workflow definition.

Once you have saved a workflow configuration for a workflow, it is active. To delete the configuration, choose *Configuration → Delete*.

If a workflow configuration exists for a workflow definition, this is apparent in the Workflow Builder when you check the workflow. The workflow configuration belongs to the workflow version for which it was created. If a new version is created, the new version also has this configuration.

A workflow configuration is client-specific. For further information on workflow configuration transport, refer to *Transport and Client Copy of Tasks [Seite 1528]*.
Tasks and Task Groups

Use
Within SAP Business Workflow you use tasks for the following purposes:

- To use methods of an object type in a workflow (single-step task [Extern])
- As a framework for defining a workflow (multistep task [Extern])

You use a task group [Extern] to collect tasks used in a common context.

Prerequisites
To create a single-step task, you must know the object type [Extern] and its method [Extern] to be executed with the task.

To create a multistep task, you must be familiar with the business process you wish to map as a workflow definition, its flow and its individual steps.

Features
You can process tasks of the following types.

- Standard task (single-step task, task type TS)
- Customer task (single-step task, task type T )
- Workflow template (multistep task, task type WS)
- Workflow task (multistep task, task type WS)
- Task group (task type TG)

Only cross-client single-step tasks (= standard tasks) and cross-client multistep tasks (= workflow templates) can be created now.

When a workflow task/customer task is copied, the task type is changed automatically to workflow template/standard task.

For further information, refer to Creating and Maintaining Tasks [Seite 1173]. Also refer to the relevant task type:

- Definition of a Single-Step Task [Seite 1175]
- Definition of a Multistep Task [Seite 1194]
- Definition of Task Groups [Seite 1208]
Creating and Maintaining Tasks

Creating tasks
2. Select the relevant task type.
3. Select  

Changing tasks
2. Identify the task that you want to process by:
   - Selecting the type of task and entering the 8-digit task number
   - Entering the task abbreviation in the field Task (using the F4 input help if applicable)
3. Select  

In a customer system you should not change workflows and tasks supplied by SAP directly. If the only changes you want to make to the task are to triggering events, terminating events or texts, you can extend the definition. To make this extension, open the definition in display mode and go to the relevant tab pages. If you want to make more extensive changes, copy the task or workflow.

Displaying tasks
2. Identify the task that you want to process by:
   - Selecting the type of task and entering the 8-digit task number
   - Entering the task abbreviation in the field Task (using the F4 input help if applicable)
3. Select  

If you are not in the original system of the task, you can create additional triggering events and terminating events, as well as change the work item text and the texts on the tab page Description.

Copying tasks
2. Identify the task that you want to process by:
Creating and Maintaining Tasks

- Selecting the type of task and entering the 8-digit task number
- Entering the task abbreviation in the field Task (using the F4 input help if applicable)

3. Select 'Select'.
4. Enter a new abbreviation and a new name for the copy.

The abbreviation and name of the copied task only exist in the logon language of the user who copied the task. The abbreviation and name of the original task, which may already exist in other languages, are not copied. However, translations of the work item text and other texts are retained in the copying process.

Deleting tasks

2. Identify the task that you want to process by:
   - Selecting the type of task and entering the 8-digit task number
   - Entering the task abbreviation in the field Task (using the F4 input help if applicable)
3. Select 'Select'.

The system checks whether there are still work items [Extern] for the task to be deleted. If this is the case, you cannot delete the task. To delete the task anyway, you must remove the work items from the system. For further information, refer to Archiving Work Items [Extern].

Result

If you have deleted or copied a task, you remain on the same screen. Otherwise you go to the screen for displaying or processing the selected task. For further information, refer to the relevant task type:

- Definition of a Single-Step Task [Seite 1175]
- Definition of a Multistep Task [Seite 1194]
- Definition of Task Groups [Seite 1208]
Definition of a Single-Step Task

Use
You use a single-step task [Extern] to define an activity that can be executed within a workflow definition or independently. This activity can be a transaction in the R/3 System.

⚠️
You have to create new single-step tasks as standard tasks [Extern]. Customer tasks [Extern] can only be edited now. There will be no functional enhancements for customer tasks.

Single-step task as a single step
It is not always necessary for a complex workflow definition involving several steps to be executed to process a business task. Often a single step in the form of a standard task can perform the function required.

Single-step task as part of a workflow definition
A single-step task can be part of a workflow definition as a step of the type activity. In the workflow definition, additional information can also be specified regarding responsibilities and deadlines. If dialog with a user is required to execute the single-step task, all possible agents [Extern] receive a dialog work item [Extern] in their Business Workplaces. If one of the recipients [Extern] executes the work item, the relevant activity is executed.

Integration

Single-step task activity
You define the activity of the single-step task using an object type method [Extern] of one of the object types [Extern] defined in the Business Object Builder.

Possible agents of single-step tasks
The possible agents [Extern] of a single-step task are specified in the relevant definition. These possible agents can be defined with reference to the organizational plan of the enterprise, with the consequence that the same task is offered for execution to several users, who have the same authorizations from an organizational perspective, in their Business Workplaces. If the task is reserved by a recipient, it is removed from the Business Workplaces of the other recipients. If the task is used within a workflow, you can define responsible agents [Extern] in the workflow definition. The recipients of the work item are all users who are both responsible agents and possible agents.

Prerequisites
The object type method and the object type are defined, tested and generated. You do not have to have defined the object type and its methods yourself. You can also use methods supplied as standard.

Features
The definition of single-step tasks is spread across several tab pages. You can make all the mandatory specifications on the tab page Basic data. The specifications on the other tab pages are optional.
Customer tasks are processed on a screen without tab pages. But the activities to be carried out are the same.

**Activities**

You should adhere to the following sequence when defining single-step tasks (and analyzing existing single-step tasks):

<table>
<thead>
<tr>
<th>What?</th>
<th>Where?</th>
</tr>
</thead>
<tbody>
<tr>
<td>Maintain names, work item text, object type and method [Seite 1177]</td>
<td>Tab page <em>Basic data</em></td>
</tr>
<tr>
<td>Maintain agent assignment [Seite 1198]</td>
<td>Additional data → Agent assignment → Maintain</td>
</tr>
<tr>
<td>Maintain elements in task container [Seite 1215]</td>
<td><img src="container.png" alt="Container" /></td>
</tr>
<tr>
<td>Maintain triggering events [Seite 1206]</td>
<td>Tab page <em>Triggering events</em></td>
</tr>
<tr>
<td>Maintain terminating events [Seite 1189]</td>
<td>Tab page <em>Terminating events</em></td>
</tr>
<tr>
<td>Maintain default roles [Seite 1191]</td>
<td>Tab page <em>Default roles</em></td>
</tr>
<tr>
<td>Maintain description and notification texts [Seite 1201]</td>
<td>Tab page <em>Description</em></td>
</tr>
<tr>
<td>Maintain additional data [Seite 1192]</td>
<td>Additional data → Selection criteria...</td>
</tr>
<tr>
<td></td>
<td>Additional data → <em>Classification</em> → Change</td>
</tr>
<tr>
<td>Maintain SAPphone properties [Seite 1207]</td>
<td>Tab page <em>SAPphone</em></td>
</tr>
</tbody>
</table>
Maintenance of Basic Data of Single-Step Task

Use
The basic data is used to identify a single-step task. You also specify the activity that is to be executed with the task in the basic data.

Prerequisites
You must know the object type and its method.

Features
Basic data maintenance is carried out on the tab page Basic data and includes the following functions:

- Processing names and work item text [Seite 1197]
- Maintaining the reference to an object method [Seite 1178]
- Maintaining the execution properties [Seite 1179]

Identification of a single-step task
Each single-step task is identified by the object identifier T (for customer task) or TS (for standard task) and an 8-digit task number allocated automatically by the system during creation. (The settings required for the number assignment are made in SAP Business Workflow Customizing [Extern]).

Assignment to application component
If you do not save the single-step task as a local object, you must specify a development class when saving. The system completes the fields Development class and Application component from this.

Transport
As cross-client transport objects (transport object PDTS), standard tasks are always connected to the transport system. Customer tasks are only connected to the transport system (transport object PDT) in a particular client if Automatic recording of changes is set for this client in table T000. In other clients, you can include customer tasks in a transport request manually. To do this, choose Task → Transport.


Maintenance of Reference to Object Method

Use
You define the activity to be executed with a single-step task by specifying an object type method.

Activities
You determine the reference to the object method on the tab page Basic data by specifying object type [Extern] and method [Extern]. The object type must be activated and defined with the method in the Business Object Builder.

The system takes the following from the definition of the object method, as applicable:

- Synchronous object method
- Object method with dialog

You cannot change these.

If method parameters are defined for the object method, the system gives you the option of creating relevant container elements automatically in the task container. The names of these container elements are then identical in the task container and the method container. You should make use of this option.

For further information, refer to Binding Definition from Method Container [Seite 1229] and Binding Definitions from Task Container [Seite 1225].

If the task is executed as a form task for SAPforms, you can specify a method as an alternative. You enter this method on the tab page Alternative methods in the line Form task. The alternative method must have the same interface as the method entered on the tab page Basic data.

The alternative method ProcessExternally should be entered for form tasks that use the method Process. The alternative method is used if the execution is not started from the Business Workplace in SAP GUI for Windows.

The alternative method entry is deleted automatically if you enter another method on the tab page Basic data.

The specifications in the lines WEB METHOD and MASS PROCESSING (IN R/3) are not currently evaluated.

Navigation
By double-clicking on the object type ID or method ID entered, you can go to the relevant object type in the Business Object Builder [Seite 1100] and process it.

Constraints
For technical reasons, the workflow system can only process method parameters created with a data type reference [Extern] to a structure or a table under the following circumstances:

- The total length of all fields of the table or the structure does not exceed 255 characters.
- Only alphanumeric data types are used in the fields of the table or structure.

If a method is used in a single-step task with parameters, the system checks whether the method parameters meet the requirements formulated above in the single-step task definition. If this is not the case, you cannot define the single-step task.
Maintenance of Execution Properties

Use
The execution properties determine additional processing characteristics of the single-step task or the work item that represents the single-step task at runtime.

Features
The following execution properties are recognized by the system:

Background processing
Set this indicator if the object method does not necessitate dialog with a user, and the processing is to be started automatically by the system.
If this indicator is set, the single-step task is executed at runtime as a background work item as soon as it is its turn in the sequence. No work item appears in the Business Workplace. You can, however, find a background work item via the work item selection.
If this indicator is not set, a dialog work item appears in the Business Workplaces of the recipients. If this work item is executed by one of its recipients, the object method is executed in the background without any further dialog.

You cannot set this indicator if the indicator Object method with dialog is set for this object method.

Confirm end of processing
If this indicator is set, the end of processing of the single-step task must be confirmed explicitly at runtime. After execution of a single-step task, the system asks via a dialog box whether the processing of the task has actually been completed. It is also possible, for example, to enter an attachment at this stage. Until this confirmation is made, the work item in question can be processed and forwarded. This may be useful if a single-step task is to be processed several times and by different people.

- The property Confirm end of processing cannot be assigned to single-step tasks that are to be executed in the background (see above).
- The property Confirm end of processing can be assigned both to single-steps task that do have terminating events and to those that do not. If there are terminating events, the relevant work item must be completed using the function Set to ‘Done’ in the Business Workplace.

Executable as form
If this indicator is set, the task meets the technical requirements for execution as an electronic form.

The system cannot check whether this property has been assigned correctly. Form tasks always have this property. If you define a form task, you must set this indicator.
Editing Names

Use
A task is identified by its abbreviation. It can also be used in input fields as a search term. The name is output in logs to describe the task. The work item that represents the task is displayed with the work item text in the Business Workplace, in the work item selection hit list and in the workflow log. If you do not specify a work item text, the name of the task is used instead. The release status is of an informal nature and indicates the development status of the task.

Procedure
1. Edit the abbreviation.
2. Edit the name.
3. Enter the work item text.
4. If you want to use expressions [Extren] in the work item text, position the cursor at the relevant point in the text and select .

The expression is replaced at execution time with the current values from the relevant container [Extren]. Note that container elements used in the work item text must be filled via a binding.

If you are looking at a task in display mode, whose original system is not the current system, you can redefine the work item text. Select and enter a text that is to replace the SAP standard text. This creates an extension of the task definition.

5. Select an appropriate release status for the task.
Maintenance of Agent Assignment

Use
This function assigns possible agents to a task.

No possible agents are assigned initially to an SAP task or an SAP workflow. You can perform this assignment in Customizing. You can change and extend these assignments at a later date as well.

Integration

Establishing recipients at runtime
The system establishes the recipients of a work item at runtime. The recipients are derived from the intersection of possible agents and responsible agents barring excluded agents. You specify the responsible agents for a step in the step definition. You can specify not only a system user, but also an object of Organizational Management such as a position, a job, an organizational unit or a work center. Alternatively, you can define a container element that contains the object of Organizational Management at runtime or a role for dynamic agent determination.

You can define agents excluded from processing within the step definition.

The user who actually executes the work item at runtime is the actual agent. The possible agents for a task must be maintained so that the recipients can be established at runtime.

Using a default role
You can specify a default role when defining a single-step task so that the agents are established dynamically when this single-step task is executed.

The default role is evaluated if the single-step task is executed alone. The default role takes on the function of the responsible agents in this case.

The default role is also evaluated if the single-step task is used in a workflow definition and no responsible agents are specified in the step definition in question.

Features

General

Task
The possible agents of a task are all of the users who are organizationally responsible and authorized to execute the task at runtime. Only they can receive a work item for this task in their Business Workplaces. Starting the task in dialog outside a workflow is included in execution.

You must assign possible agents to a task for the task to be started and executed at runtime.

You can create several assignments for a task and also use various types of agent doing so.
The task “Check invoice” is linked with the following possible agents.

- Organizational unit Purchasing
- Organizational unit Internal auditing
- Job Accountant

You do not have to assign possible agents to tasks designated as background tasks. You must assign possible agents to those background tasks that are also to be started as single steps in dialog.

**Multistep task**

The possible agents of a multistep task are those users who are authorized to start the workflow connected with the task manually.

You only have to assign possible agents to a multistep task if the workflow is to be started directly by users. If the workflow is only to be started by events, do not assign any possible agents to the multistep task.

**Task groups**

If you assign possible agents to a task group, this assignment is passed on to all tasks contained in the task group.

Check whether the assignment defined here really is to be apply to all task groups, single-step tasks and multistep tasks contained.

**Basic properties of tasks**

**Task**

Choose the basic property of the task from the following alternatives:

- **General task**
  
  If you define a task as a general task [Extern], all users can execute the task. This is useful if the task is used in a workflow and you only want to define the recipients in the step definition. A recipient can forward associated work items to all users.

  Work items whose tasks are defined as general tasks and for which no responsible agents or default roles are defined are offered to all users of the R/3 System for execution in their Business Workplaces.

- **General forwarding allowed**
  
  A work item that represents a task with this property can be forwarded by one of its recipients to all users, even if they are not possible agents of the task.

- **General forwarding not allowed**
  
  A work item that represents a task with this property can be forwarded by one of its recipients only to the possible agents of the task.

**Multistep task**

With a multistep task you can only decide whether it is a general task or not.

**Activities**

To process the basic properties of a task, choose Additional data → Classification → Change in task maintenance.
To assign a task you have defined to its possible agents or process its basic properties, choose
Additional data → Agent assignment → Maintain and select the task.
Select Attributes... to define the basic properties of the task.
Choose Agent assignment → Create... to assign the possible agents.

You can process the possible agents of the task and workflow (multistep task) from
the Workflow Builder as well.
Maintaining Task Texts

Prerequisites
You enter the following information on the tab page Description:

- Task description
  You can describe the task or make notes and recommendations for processing. The task description is displayed in the work item display.

- Notification texts
  These texts are sent as notification to the message recipients for completion or missed deadlines.

Procedure
1. Switch to the tab page Description.
2. Select the text type that you want to create or change.
3. Select .
   If you are looking at a task in display mode, whose original system is not the current system, you can change the existing texts. This creates an extension of the task definition.
4. Compose your text.
   To add text variables to your text, choose Include → Expression. You can add as many text variables to the text as you want. For further information, refer to Text Variables and Replacement and Text Processing with the SAPscript Editor.

   You want to display the number of the accounting document currently being processed in the task description for a standard task. The document number is defined under the name DocumentNo as a key field for the object type BU2068.

   The reference to the processed object (of type BU2068 in this example) is located in the task container under the element _WI_Object_ID. To reference the document number in the text, write &BU2068.DocumentNo at the appropriate point in the text.
Definition of the Task Container

Use
This function is used to process the container elements of the task container [Extern]. Each task container already contains the workflow system elements [Extern] and any additional elements for the method parameters of the referenced object method.

You have to add additional elements [Seite 1212] to the task container if:

- Additional values are required for variable replacement in the work item text or in the long texts
- Additional information is required to resolve a default role

Values are assigned to the task container at runtime in the following ways:

- By initial value assignment [Seite 1205]
- Via binding from the workflow, event or method container

Initial values are overwritten by bindings. Initial value assignment and processing of the task container are only possible during definition of the single-step task.
Creating Container Elements

Prerequisites
You are in one of the definition tools and want to add a new element to the container.

Procedure
1. Select Container.
2. Select 
3. Enter the technical name of the container element into the field Element.
   You give each container element a technical name that can be used to identify it uniquely. The technical name must begin with a letter. This can be followed by letters, underscores or digits. Not case-sensitive. The technical name you choose should be in English.
4. Under Texts maintain the name and the description.
   This text can include upper and lower cases.
5. Under Properties, select whether the new element is to be an import and/or an export element. Mark the element as mandatory if applicable.
   The values for each of the container elements may be in the form of a multiline list of field values or object references. This list is then saved under an element name. You can select multiline in the container definition so that this container element can hold a multiline list of values.
6. According to the data type reference [Extern] of the container element, make the following entries. If the new container element:
   - Is to reference a structure, enter the table or structure in the field Reference table
   - Is to reference an ABAP Dictionary field, specify a reference field as well
   - Is to reference an object type, enter the ID of the object type in the field Object type
   You can use the F4 input help for all of these.
   Specification of an ABAP Dictionary field reference is not mandatory. If no field reference is specified, the data type is C(255) (upper and lower case). Specification of an object type is not mandatory either. If no object type is specified, the container element can be assigned a reference to an object of any type.
7. Select .
Assigning Initial Values

Use
The initial value assignment of constants to container elements should be regarded as a preassignment of values to container elements.

Procedure
1. Select Initial values.
   All container elements designated as import elements are displayed. The import workflow system elements are not displayed until you select except for _WI_Object_Id.

   You can only assign values to those container elements that are designated as import elements.

2. Enter the initial value in the field Expression.
   To assign several values to a target element defined as multiline, create the additional input lines using.

Result
At runtime the system executes the initial value assignment in accordance with your definition.

If there is a binding definition from another container to the container with the initial values, the system executes this binding at runtime after the initial value assignment. Any initial values are overwritten.
Creating Triggering Events

Procedure
You maintain the triggering events on the tab page Triggering events.

1. Identify the event by specifying its instance ID and object type. The event must be defined for this object type.

If you are looking at a task in display mode, whose original system is not the current system, you can create additional triggering events. This creates an extension of the task definition.

2. To activate the event receiver linkage, select in the column. Active event linkages are denoted by .

3. To define the binding from the event container, select the relevant event line and choose . For further information, refer to Binding Definitions from the Event Container [Seite 1227].

4. Select the triggering event and choose . The properties of the event linkage are displayed and can be changed.

Result
The workflow system enters the linkage between the event and the event-receiving task in the relevant linkage table automatically. For further information on linkage transport, refer to Transport and Client Copy [Seite 1528].

If an error occurs when a task or workflow is being started with an event, the standard procedure is for the system to deactivate the event linkage to avoid any follow-on errors. At the same time, the system sends an e-mail to the workflow system administrator. But you can change this in event queue administration [Seite 1515].
Definition of Terminating Events of Task

Use
This function is used to define the terminating events [Extern] of the task.

- Tasks that refer to an asynchronous object method [Extern] must always be defined with at least one terminating event. At runtime, the relevant work item is only terminated if one of the defined terminating events occurs.
- Tasks that refer to a synchronous object method [Extern] can also be defined with terminating events. At runtime, the relevant work item is generally terminated when the synchronous method has been successfully executed. But the work item is also terminated if one of the defined terminating events occurs.

Whether a method is defined as synchronous or asynchronous depends on its implementation in the Business Object Builder.

The terminating events of a task express the possible end statuses of the task. A terminating event always refers to one object for which the event must be triggered. You define the object for which the event must be triggered in a container element in the task container.

The event DELETED of object type BUS2032 is defined as a terminating event for a task. At runtime the sales order 123456 is passed via the task container. The task is now only terminated if the sales order 123456 is deleted and the event BUS2032.DELETED is triggered. If another sales order is deleted, the task is not terminated.

Integration
If the task is used as a step in a workflow definition, you can go to a separate modeling branch when the event occurs. For further information, refer to Maintenance of Tab Page Outcomes [Seite 1028]

The task Release budget has the terminating event Budget cancelled.

If a task is terminated by an event, the execution of the workflow is continued. The method does not necessarily have to have been executed. If a task is aborted or non-defined processing statuses arise, the work item is not terminated since no event is created. The workflow is not continued.

Activities
You maintain the terminating events on the tab page Terminating events.

- You identify the event by specifying its instance ID and object type. The event must be defined for this object type.

If you are looking at a task in display mode, whose original system is not the current system, you can create additional terminating events. This creates an extension of the task definition.
Definition of Terminating Events of Task

- You specify a container element of the task container, which at runtime contains a reference to the object for which the event has occurred. This is generally the task container element _WI_Object_ID.

The workflow system enters the instance linkage [Extern] required into the linkage table [extern] automatically after the task is started.

If you want to check the properties of the terminating event, select . The properties of the instance linkage are displayed and can be changed.

If you change the properties of terminating events, all terminating events that exist for this task in the workflow system are affected by the change. Therefore do not change the indicator for activation of the event.
Roles in Single-Step Tasks

Use
You use roles [Extern] to define the responsible agents [Extern] of the single-step task and the recipients of notifications.

Features
To determine the responsible agents and recipients of notifications, you can specify roles. Depending on the use of the task, these roles are evaluated as follows:

Task is started in dialog or via an event
The system resolves the role Agents at runtime and determines the responsible agents of the task. All responsible agents who are also possible agents [Extern] become recipients [Extern] of the relevant work item.

Task is started from a workflow
The roles entered are only resolved if nothing is specified in the step definition.

Activities
You make the presettings on the tab page Default roles.

Binding
You must define the binding from the task container to the role container for each role. For further information, refer to Role Resolution [Seite 1276].
Maintenance of Classification and Lock Indicators

Use

Classification
You can arrange single-step tasks into various task classes within the SAP R/3 System. The task classes professional, personal or disciplinary are standard.
The system checks the task class under the field name TASK_CLASS in the authorization object S_WF_WI. This system uses this authorization object to check the authorization for carrying out specific actions on particular work items.
The assignment of single-step tasks to classes is also important when defining substitutes. You can specify that only the single-step tasks of a particular class are displayed to your substitute. Take this into account when you create new classes and assign single-step tasks to the classes.

Lock indicators
You can set the following lock indicators for single-step tasks:
- Locked against instantiation
  The system does not create a work item for this single-step task.
- Locked against execution
  The system does create a work item for this single-step task, but it cannot be executed.

Integration

Adding new task classes in the system
In Customizing for Organizational Management, you can add more task classes to the predefined classes.

Activities
The maintenance of task classes and lock indicators is carried out under Additional data → Classification.
Maintenance of SAPphone Properties

Use
The SAPphone properties determine the possible uses of a task in the SAPphone environment.

Integration
SAPphone integrates telephony functions into the R/3 System. R/3 applications use this to integrate the telephone into the work process. SAPphone provides the Basis technology required. The processing of outgoing and incoming telephone calls is supported with data from the R/3 System environment (for example, telephone number or name). An incoming call starts a search for caller data. The result of the search is displayed automatically in an information window. For further information, refer to SAPphone [Extern].

Features
If you label the task accordingly, it is displayed and can be executed in the event of incoming or outgoing calls. The task container must contain appropriate container elements to enable the information derived from the telephone data to be used in the task.
Definition of a Multistep Task

Use
A multistep task is a formal framework for a workflow definition. When you create a workflow directly in the Workflow Builder, the multistep task is created by the system. Multistep tasks can be divided up into workflow templates and workflow tasks.

New multistep tasks must be created as workflow templates. Workflow tasks can no longer be created, but can be edited. There will be no functional enhancements for workflow tasks.

Integration

Reference to workflow definition
A multistep task contains a reference to a workflow definition. The business activity described with the multistep task therefore usually covers several steps in what may be a complex arrangement. This is a major difference when compared with tasks.

Multistep task as interface description
A multistep task defines the formal framework for a workflow definition with the description of the interface. In addition to the reference to the workflow definition, the multistep task contains other information:

- Definition of the workflow container (interface description)
- Triggering events

Multistep Task
Definition of a Multistep Task

Features
The definition of workflow templates is spread across several tab pages.

- **Basic data**
- **Description**
- **Triggering events**
- **SAPphone**

Workflow tasks are defined on a screen without tab pages. But the activities to be carried out are the same.

Activities
You should adhere to the following sequence when defining multistep tasks (and analyzing existing multistep tasks):

<table>
<thead>
<tr>
<th>What?</th>
<th>Where?</th>
</tr>
</thead>
<tbody>
<tr>
<td>Maintain abbreviation, name and work item text [Seite 1197]</td>
<td>Tab page <em>Basic data</em></td>
</tr>
<tr>
<td>Define elements in workflow container [Seite 1213]</td>
<td>Container</td>
</tr>
<tr>
<td>Maintain triggering events if required. [Seite 1206]</td>
<td>Tab page <em>Triggering events</em></td>
</tr>
<tr>
<td>Maintain description and notification texts [Seite 1201]</td>
<td>Tab page <em>Description</em></td>
</tr>
<tr>
<td>Maintain agent assignment [Seite 1198]</td>
<td>Additional data → Agent assignment → Maintain</td>
</tr>
<tr>
<td>Maintain SAPphone properties [Seite 1207]</td>
<td>Tab page <em>SAPphone</em></td>
</tr>
</tbody>
</table>
Maintenance of Basic Data of Multistep Task

Use
The basic data is used to identify a multistep task.

Features
The tab page Basic data is used for processing the names of the multistep task [Seite 1197]. You can also call the Workflow Builder from there.

Identification of a multistep task
Each multistep task is identified with an 8-digit task number and the organizational object type WS (for workflow template) or WF (for workflow task). The number is assigned automatically by the system when a newly created task is saved.
You define the first three digits of the number in Customizing [Extern].

Transport
As cross-client transport objects (transport object PDWS), workflow templates are always connected to the transport system.
Workflow tasks are only connected to the transport system in a particular client if Automatic recording of changes is set for this client in table T000. In other clients, you can include workflow tasks in a transport request manually. To do this, choose Workflow task → Transport.

Activities
If you do not save the multistep task as a local object, you must specify a development class when saving.
The system completes the fields Development class and Application component from this.
Editing Names

Use
A task is identified by its abbreviation. It can also be used in input fields as a search term. The name is output in logs to describe the task.
The work item that represents the task is displayed with the work item text in the Business Workplace, in the work item selection hit list and in the workflow log. If you do not specify a work item text, the name of the task is used instead.
The release status is of an informal nature and indicates the development status of the task.

Procedure
6. Edit the abbreviation.
7. Edit the name.
8. Enter the work item text.
9. If you want to use expressions [Extern] in the work item text, position the cursor at the relevant point in the text and select .
The expression is replaced at execution time with the current values from the relevant container [Extern]. Note that container elements used in the work item text must be filled via a binding.

If you are looking at a task in display mode, whose original system is not the current system, you can redefine the work item text. Select and enter a text that is to replace the SAP standard text. This creates an extension of the task definition.
10. Select an appropriate release status for the task.
Maintenance of Agent Assignment

Use
This function assigns possible agents to a task.

No possible agents are assigned initially to an SAP task or an SAP workflow. You can perform this assignment in Customizing. You can change and extend these assignments at a later date as well.

Integration

Establishing recipients at runtime
The system establishes the recipients of a work item at runtime. The recipients are derived from the intersection of possible agents and responsible agents barring excluded agents. You specify the responsible agents for a step in the step definition. You can specify not only a system user, but also an object of Organizational Management such as a position, a job, an organizational unit or a work center. Alternatively, you can define a container element that contains the object of Organizational Management at runtime or a role for dynamic agent determination.

You can define agents excluded from processing within the step definition.
The user who actually executes the work item at runtime is the actual agent. The possible agents for a task must be maintained so that the recipients can be established at runtime.

Using a default role
You can specify a default role when defining a single-step task so that the agents are established dynamically when this single-step task is executed.
The default role is evaluated if the single-step task is executed alone. The default role takes on the function of the responsible agents in this case.
The default role is also evaluated if the single-step task is used in a workflow definition and no responsible agents are specified in the step definition in question.

Features

General

Task
The possible agents of a task are all of the users who are organizationally responsible and authorized to execute the task at runtime. Only they can receive a work item for this task in their Business Workplaces. Starting the task in dialog outside a workflow is included in execution.

You must assign possible agents to a task for the task to be started and executed at runtime.
You can create several assignments for a task and also use various types of agent doing so.
The task “Check invoice” is linked with the following possible agents.

- Organizational unit Purchasing
- Organizational unit Internal auditing
- Job Accountant

You do not have to assign possible agents to tasks designated as background tasks. You must assign possible agents to those background tasks that are also to be started as single steps in dialog.

**Multistep task**

The possible agents of a multistep task are those users who are authorized to start the workflow connected with the task manually.

You only have to assign possible agents to a multistep task if the workflow is to be started directly by users. If the workflow is only to be started by events, do not assign any possible agents to the multistep task.

**Task groups**

If you assign possible agents to a task group, this assignment in passed on to all tasks contained in the task group.

Check whether the assignment defined here really is to be apply to all task groups, single-step tasks and multistep tasks contained.

**Basic properties of tasks**

**Task**

Choose the basic property of the task from the following alternatives:

- **General task**
  
  If you define a task as a general task [Extern], all users can execute the task. This is useful if the task is used in a workflow and you only want to define the recipients in the step definition. A recipient can forward associated work items to all users.

  Work items whose tasks are defined as general tasks and for which no responsible agents or default roles are defined are offered to all users of the R/3 System for execution in their Business Workplaces.

- **General forwarding allowed**
  
  A work item that represents a task with this property can be forwarded by one of its recipients to all users, even if they are not possible agents of the task.

- **General forwarding not allowed**
  
  A work item that represents a task with this property can be forwarded by one of its recipients only to the possible agents of the task.

**Multistep task**

With a multistep task you can only decide whether it is a general task or not.

**Activities**

To process the basic properties of a task, choose Additional data → Classification → Change in task maintenance.
Maintenance of Agent Assignment

To assign a task you have defined to its possible agents or process its basic properties, choose Additional data → Agent assignment → Maintain and select the task.

Select Attributes... to define the basic properties of the task.

Choose Agent assignment → Create... to assign the possible agents.

You can process the possible agents of the task and workflow (multistep task) from the Workflow Builder as well.
Maintaining Task Texts

Prerequisites
You enter the following information on the tab page Description:

- Task description
  You can describe the task or make notes and recommendations for processing. The task description is displayed in the work item display.

- Notification texts
  These texts are sent as notification to the message recipients for completion [Extern] or missed deadlines [Extern].

Procedure
4. Switch to the tab page Description.
5. Select the text type that you want to create or change.
6. Select .

If you are looking at a task in display mode, whose original system is not the current system, you can change the existing texts. This creates an extension of the task definition.

5. Compose your text.
To add text variables to your text, choose Include → Expression. You can add as many text variables to the text as you want. For further information, refer to Text Variables and Replacement [Extern] and Text Processing with the SAPscript Editor [Extern].

You want to display the number of the accounting document currently being processed in the task description for a standard task. The document number is defined under the name DocumentNo as a key field for the object type BUS2068 (accounting document).

The reference to the processed object (of type BUS2068 in this example) is located in the task container under the element _WI_Object_ID. To reference the document number in the text, write &BUS2068.DocumentNo& at the appropriate point in the text.
Definition of the Workflow Container

Use
In the definition of the workflow container, you process the container elements required for the workflow execution. This may be data that you require for controlling the workflow execution directly, or data that you want to forward from one step execution to another. Every workflow container already contains workflow system elements.

⚠️

The system no longer evaluates activity-related container elements as of Release 4.6A. The IDs of the activity-related container elements begin with ACTIVITY_<StepNumber>.

You can assign initial values to container elements at the start of the workflow. You must carry out this initial value assignment in the definition of the multistep task. For further information, refer to Assigning Initial Values.

Integration
You can process the workflow container both during definition of the multistep task and from the Workflow Builder.

Features
Container elements for which the import or export indicator is set form the interface of the workflow.

Definition of the data interface
Both the import and the export indicators can be set for container elements.

Container elements of the interface are not subject to the versioning of the workflow definition.

Container elements with import indicator set
These container elements hold the values that can be passed when execution of the workflow is started. If the values have to be passed, set the indicator mandatory as well.

These container elements can be supplied with values in the following ways at the start of execution:

- Via a binding from the event container if the workflow has one or more triggering events
- Via initial value assignment
- Via an electronic form if the workflow is started with it
- Via direct entry if the workflow is started with the function Start workflow (test environment)

If the object to be processed is not created in the workflow, a container element with import indicator set must be defined in the workflow container, which holds the object reference. You create this container element with a data type reference to the relevant object type.
Definition of the Workflow Container

**Container elements with export indicator set**
These container elements hold the values which the workflow is to return to a calling workflow. You need them if the workflow is a subworkflow of a superordinate workflow and is to return data to its superordinate workflow.

**Definition of local container elements**
All container elements for which neither the import nor the export indicator is set are local container elements of the workflow definition. They are only valid in this workflow definition. Typical applications are:

- Passing data from one task container to another
- Determining the responsible agents for an activity dynamically
- Setting deadlines dynamically
- Implementing counting loops

All local container elements are subject to the versioning [Seite 1019] of the workflow definition.

If you create a local container element in the Workflow Builder, it is only included into the workflow definition currently being processed. If you create it in the definition of the multistep task, it is included into all versions existing at the time.

For further information, refer to Creating Container Elements [Seite 1212].
Creating Container Elements

Prerequisites
You are in one of the definition tools and want to add a new element to the container.

Procedure
7. Select `Container`.
8. Select `Container`.
9. Enter the technical name of the container element into the field `Element`.
   You give each container element a technical name that can be used to identify it uniquely. The technical name must begin with a letter. This can be followed by letters, underscores or digits. Not case-sensitive. The technical name you choose should be in English.
10. Under `Texts` maintain the `name` and the `description`.
    This text can include upper and lower cases.
11. Under `Properties`, select whether the new element is to be an `import` and/or an `export` element. Mark the element as `mandatory` if applicable.
    The values for each of the container elements may be in the form of a multiline list of field values or object references. This list is then saved under an element name. You can select `multiline` in the container definition so that this container element can hold a multiline list of values.
12. According to the `data type reference [Extern]` of the container element, make the following entries. If the new container element:
    - Is to reference a `structure`, enter the table or structure in the field `Reference table`
    - Is to reference an `ABAP Dictionary field`, specify a reference field as well
    - Is to reference an `object type`, enter the ID of the object type in the field `Object type`
    You can use the F4 input help for all of these.

    Specification of a ABAP Dictionary field reference is not mandatory. If no field reference is specified, the data type is `C(255)` (upper and lower case). Specification of an object type is not mandatory either. If no object type is specified, the container element can be assigned a reference to an object of any type.

8. Select `Container`.
Assigning Initial Values

Use
The initial value assignment of constants to container elements should be regarded as a preassignment of values to container elements.

Procedure
3. Select initial values.

   All container elements designated as import elements are displayed. The import workflow system elements are not displayed until you select except for _WI_Object_Id.

   You can only assign values to those container elements that are designated as import elements.

4. Enter the initial value in the field Expression.

   To assign several values to a target element defined as multiline, create the additional input lines using.

Result
At runtime the system executes the initial value assignment in accordance with your definition.

If there is a binding definition from another container to the container with the initial values, the system executes this binding at runtime after the initial value assignment. Any initial values are overwritten.
Creating Triggering Events

Procedure

You maintain the triggering events on the tab page Triggering events.

5. Identify the event by specifying its instance ID and object type. The event must be defined for this object type.

If you are looking at a task in display mode, whose original system is not the current system, you can create additional triggering events. This creates an extension of the task definition.

6. To activate the event receiver linkage, select in the column. Active event linkages are denoted by .

7. To define the binding from the event container, select the relevant event line and choose . For further information, refer to Binding Definitions from the Event Container [Seite 1227].

8. Select the triggering event and choose . The properties of the event linkage are displayed and can be changed.

Result

The workflow system enters the linkage between the event and the event-receiving task in the relevant linkage table automatically. For further information on linkage transport, refer to Transport and Client Copy [Seite 1528].

If an error occurs when a task or workflow is being started with an event, the standard procedure is for the system to deactivate the event linkage to avoid any follow-on errors. At the same time, the system sends an e-mail to the workflow system administrator. But you can change this in event queue administration [Seite 1515].
Maintenance of SAPphone Properties

Use
The SAPphone properties determine the possible uses of a task in the SAPphone environment.

Integration
SAPphone integrates telephony functions into the R/3 System. R/3 applications use this to integrate the telephone into the work process. SAPphone provides the Basis technology required. The processing of outgoing and incoming telephone calls is supported with data from the R/3 System environment (for example, telephone number or name). An incoming call starts a search for caller data. The result of the search is displayed automatically in an information window. For further information, refer to SAPphone [Extern].

Features
If you label the task accordingly, it is displayed and can be executed in the event of incoming or outgoing calls. The task container must contain appropriate container elements to enable the information derived from the telephone data to be used in the task.
Definition of a Task Group

Use
You define task groups if you want to collect and group standard tasks and workflow templates that are logically related. You can also collect other task groups within a task group, hence making hierarchies of task groups.

Task groups are not restricted within applications. You can collect tasks from within one application component as well as tasks from different application components.

A standard task or workflow template can be used in several task groups.

You can use task groups to delimit the search range when calling the Business Workflow Explorer.

Integration
The application component is also available as a collective evaluation criterion for standard tasks and workflow templates. Each task is assigned to an application component automatically via its development class.

Features
You define the task group with its basic data and enter the tasks and task groups that it is to contain. Enter a description for the task group.

Agent assignment
It is not necessary to assign possible agents to a task group. If you do, the possible agents are passed on to the tasks and task groups contained.

Activities
You should adhere to the following sequence when defining single-step tasks (and analyzing existing single-step tasks):

<table>
<thead>
<tr>
<th>What?</th>
<th>Where?</th>
</tr>
</thead>
<tbody>
<tr>
<td>Maintain abbreviation and name.</td>
<td>Tab page Basic data</td>
</tr>
<tr>
<td>The system assigns the identifying number of the task group when saving.</td>
<td></td>
</tr>
<tr>
<td>Maintain task description</td>
<td>Tab page Description</td>
</tr>
<tr>
<td>Enter tasks and task groups collected in this task group.</td>
<td>Tab pages Standard tasks, Workflow templates and Task groups</td>
</tr>
<tr>
<td>Maintain agent assignment</td>
<td>Extras → Agent assignment → Maintain</td>
</tr>
<tr>
<td>Graphical representation of the hierarchy of the task group, including the subordinate task groups.</td>
<td>Hierarchy</td>
</tr>
</tbody>
</table>
Definition of a Container

Use
The elements of a container are described with ID, data type reference and other properties. Container definition is carried out with the container editor.

Container

Table-like data structure for holding
- Field values (including multiline)
- Object references (including multiline)

<table>
<thead>
<tr>
<th>Element ID</th>
<th>Data type reference</th>
<th>Description</th>
<th>Multiline</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>CostCenter</td>
<td>VBAK-KOSTL</td>
<td>Cost center</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Customer</td>
<td>KNAV</td>
<td>Customer</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Items</td>
<td>VBAP</td>
<td>Document items</td>
<td>X</td>
<td></td>
</tr>
</tbody>
</table>

Integration
A container definition is not an independent action. Container definitions are always carried out within a definition tool using the editor provided.

Features

Container definition
To describe which data a container can hold at runtime, of which type and under which name, the container must first be defined. This container definition defines which container elements a container holds. You can define the following containers as required:

- Method container
- Event container
- Role container
- Task container
- Workflow container
Usually containers are already defined. These containers hold standard container elements defined by the workflow system. These container elements will often be adequate for your requirements meaning that a more extensive container definition will not be necessary.

**Container instance**

A container instance is an instance of a container at runtime. It contains a value of the defined data type for each container element.

**Binding**

Values can be assigned to container elements in different ways.

- Value assignment via a binding [Page 1218] from another container according to a binding definition carried out previously.
- Value assignment directly to a container instance in the program of an application.
- Value assignment with an initial value determined at definition time when the container instance is created.

When processing container instances within a program for object type implementation (write and read accesses), use the macro instructions [Page 1133] provided.

**Activities**

You can perform these activities from the container editor. The editor is called at the following places:

- For the method container or the event container in the *Business Object Builder*

  ![Next Action](image)

  When you create a method parameter or an event parameter, it is automatically stored in the relevant container.

- For the task container or the workflow container in the task definition or the workflow definition

- For the role container in the definition of the standard roles

**Creating container elements**

To create a new container element [Page 1212], select ![](image).

**Editing container elements**

Double-click on the relevant table entry to edit a container element.

**Copying, cutting or pasting container elements**

You select an element in a container by positioning the cursor on the relevant line and choosing ![](image). You can copy or cut selected container elements. If they are pasted in again, you must rename them.

**Display conventions**

The following conventions apply in the container editor:

- Workflow system elements have a green background and are locked for processing
- Container elements with import or export indicator set have a gray background
- Local container elements have a white background

Container elements have the following symbols in the Workflow Builder:
Definition of a Container

- Local container element
- Container element with import indicator set
- Container element with import and export indicators set
- Container element with export indicator set
Creating Container Elements

Prerequisites
You are in one of the definition tools and want to add a new element to the container.

Procedure
13. Select Container.
14. Select ![Select](icon).
15. Enter the technical name of the container element into the field *Element*.
   You give each container element a technical name that can be used to identify it uniquely. The technical name must begin with a letter. This can be followed by letters, underscores or digits. Not case-sensitive. The technical name you choose should be in English.
16. Under Texts maintain the *name* and the *description*.
   This text can include upper and lower cases.
17. Under Properties, select whether the new element is to be an *import* and/or an *export* element. Mark the element as *mandatory* if applicable.
   The values for each of the container elements may be in the form of a multiline list of field values or object references. This list is then saved under an element name. You can select *multiline* in the container definition so that this container element can hold a multiline list of values.
18. According to the data type reference [Extern] of the container element, make the following entries. If the new container element:
   - Is to reference a *structure*, enter the table or structure in the field *Reference table*
   - Is to reference an *ABAP Dictionary field*, specify a reference field as well
   - Is to reference an *object type*, enter the ID of the object type in the field *Object type*
   You can use the F4 input help for all of these.
   ![Arrow](icon)
   Specification of a ABAP Dictionary field reference is not mandatory. If no field reference is specified, the data type is *C(255)* (upper and lower case). Specification of an object type is not mandatory either. If no object type is specified, the container element can be assigned a reference to an object of any type.
9. Select ![Select](icon).
Definition of the Workflow Container

Use
In the definition of the workflow container [Extern], you process the container elements required for the workflow execution. This may be data that you require for controlling the workflow execution directly, or data that you want to forward from one step execution to another. Every workflow container already contains workflow system elements [Extern].

⚠️ The system no longer evaluates activity-related container elements as of Release 4.6A. The IDs of the activity-related container elements begin with ACTIVITY_<StepNumber>.

You can assign initial values to container elements at the start of the workflow. You must carry out this initial value assignment in the definition of the multistep task. For further information, refer to Assigning Initial Values [Seite 1205].

Integration
You can process the workflow container both during definition of the multistep task and from the Workflow Builder.

Features
Container elements for which the import or export indicator is set form the interface of the workflow.

Definition of the data interface
Both the import and the export indicators can be set for container elements.

Container elements of the interface are not subject to the versioning [Seite 1019] of the workflow definition.

Container elements with import indicator set
These container elements hold the values that can be passed when execution of the workflow is started. If the values have to be passed, set the indicator mandatory as well. These container elements can be supplied with values in the following ways at the start of execution:

- Via a binding from the event container [Seite 1227] if the workflow has one or more triggering events
- Via initial value assignment [Seite 1205]
- Via an electronic form [Seite 1722] if the workflow is started with it
- Via direct entry if the workflow is started with the function Start workflow (test environment) [Seite 1452]

If the object to be processed is not created in the workflow, a container element with import indicator set must be defined in the workflow container, which holds the object reference. You create this container element with a data type reference to the relevant object type.
Definition of the Workflow Container

Container elements with export indicator set
These container elements hold the values which the workflow is to return to a calling workflow. You need them if the workflow is a subworkflow of a superordinate workflow and is to return data to its superordinate workflow.

Definition of local container elements
All container elements for which neither the import nor the export indicator is set are local container elements of the workflow definition. They are only valid in this workflow definition. Typical applications are:

- Passing data from one task container to another
- Determining the responsible agents for an activity dynamically
- Setting deadlines dynamically
- Implementing counting loops

All local container elements are subject to the versioning [Seite 1019] of the workflow definition.

If you create a local container element in the Workflow Builder, it is only included into the workflow definition currently being processed. If you create it in the definition of the multistep task, it is included into all versions existing at the time.

For further information, refer to Creating Container Elements [Seite 1212].
Definition of the Task Container

Use
This function is used to process the container elements of the task container [Extern]. Each task container already contains the workflow system elements [Extern] and any additional elements for the method parameters of the referenced object method.

You have to add additional elements [Seite 1212] to the task container if:

- Additional values are required for variable replacement in the work item text or in the long texts
- Additional information is required to resolve a default role

Values are assigned to the task container at runtime in the following ways:

- By initial value assignment [Seite 1205]
- Via binding from the workflow, event or method container

Initial values are overwritten by bindings. Initial value assignment and processing of the task container are only possible during definition of the single-step task.
Definition of the Method Container

Use
The method container holds the method parameters [Extern], which are used for information exchange between the caller of a method and the method. At runtime, method parameters are:

- Passed to the method (import parameters)
- Returned from the method (export parameters)

The definition of the parameters determines the interface of the method call.

The method container is called when the binding from the method to the single-step task is called.

Features
All method parameters are container elements of the method container, which is created automatically by the system when the method is defined.

The object reference to the object on which the method operates is not a container element of the method container.

You can display the content of the method container from the binding definition in the container editor.
Definition of Event Container

Use
The event container contains the event parameters [Extern] as container elements. If you define triggering [Extern] events or terminating [Extern] events for a task, you can specify the binding from the event container to the task container. If you define triggering events for a workflow, you can specify the binding from the event container to the workflow container.

Features
The event container contains workflow system elements [Extern] as standard. You can add more container elements to the event container within object type definition in the Business Object Builder. This is not usually necessary, however.
Definition of Binding

Use
At definition time you create binding definitions for the workflow. This involves specifying the assignment rules for how data is exchanged between two containers. At runtime these binding definitions are executed making workflow execution possible. You can also allocate values to container elements using initial value assignment at definition time. These initial values are always overwritten by a binding.

Application
Within the transaction for creating an accounting document (object type AccountingDocument), an event is generated (event created), which indicates that a document has been successfully created. This event is to be the triggering event of a workflow.

Event
The event container is passed with this event. This container always contains an element _Evt_Object with a reference to the object created.

Workflow container
In the container of the workflow that is to be started by the event created of the object type AccountingDocument, you have created a mandatory import element (under the ID AccountingDocument for example) with a data type reference to the object type AccountingDocument. The element can therefore hold the reference to the object created (type AccountingDocument) at runtime.

Binding to the workflow
The event parameter (= element in event container)_Evt_Object can be passed in a binding to the workflow container of the workflow that is to be started in response to the event. You define the following binding for this:

_Evt_Object (event container) ➔ AccountingDocument (workflow container)

Features
The graphic below illustrates which bindings can be defined.
### Possible Binding Definitions

#### Binding definition from task container
You define the bindings from the task container in the following two places.

**In the definition of a task**
There are binding definitions to the:
- Method container (for import parameters of method).
- Role container (for definition of default roles).

**In the step definition in the Workflow Builder**
The binding definition to the workflow container.

#### Binding definition from workflow container
You define the bindings from the workflow container within the Workflow Builder. There are binding definitions to the:
- Task container (activity, user decision, document from template, send mail).
- Event container (event creator).
- Role container (for definition of responsible agents and recipients).

#### Binding definition from method container
You define the binding from the method container to the task container (for export parameters of the method with synchronous methods) during definition of a task.

#### Binding definition from event container
You define the bindings from the event container in the following three places.
In the definition of a task for the binding definition to the task container (for triggering or terminating events of the task).

- In the Workflow Builder in the basic data of a workflow for the binding definition to the workflow container (for triggering events of the workflow).
- In the Workflow Builder in the definition of a wait step for the binding definition to the workflow container.

The system makes proposals automatically for the binding definition between the workflow container and the task container. You can change these proposals. For further information, refer to Rule-Based Proposal for Binding Definitions [Seite 1221].

**Activities**

You define the bindings in the binding editor [Seite 1232], which is always called via a definition tool. This cannot be done separately.
Rule-Based Proposal for Binding Definitions

Use
The workflow system automatically makes proposals for the following binding definitions:
- Binding definition from workflow container to task container
- Binding definition from task container to workflow container
- Binding definition from task container to method container
- Binding definition from method container to task container

If additional elements are required in a container for this proposal, the system offers to create these elements.

Features
The system evaluates the data type references [Extern] of the relevant container elements and follows specific rules to generate a proposal for the binding definition.

If the system cannot establish a valid binding definition with the elements available, it proposes an appropriate new element for the workflow container and a binding definition based on this new element.

If the system can establish more than one valid binding definition with equal probability using the elements available, it does not propose any binding definition.

The system displays the proposals for new elements in the container in the upper part of the screen and the proposals for the binding definitions in the lower part of the screen.

Rules for binding generation
For each binding definition proposal, you can display the rules used by the system to establish the proposal. To do this, double-click or of the binding definition.

Further Information
If the system does not make any proposals for the binding definition to the target container for a particular element, you can display the reasons behind this.

To do this, choose Further information. A dialog box with a list of the relevant elements is displayed. Select an element and choose for further information.

Activities
You can adopt the proposed container elements and binding definitions without changing them.

If you do not want to adopt a proposal for a new element or a binding definition, you proceed as follows:
1. You deselect the proposal that you do not want to adopt.

2. You select . If applicable, the binding definition that uses a deselected container element is then also deselected.

3. To adopt a proposal after changes, you select .

If you want the system to create a new element in the container for this binding, even though a proposal has been made for the binding using an existing element, you proceed as follows:
1. You position the cursor on the line with a proposed binding.

2. You select . The system then proposes a new element for the container, which has the same name as the element in the source container.
The system enters this proposal in the upper part of the screen. The proposal for the binding refers to the new element.

3. To adopt the proposal after changes, you choose ☑. You can also make more changes as described above.
Binding Definitions from Workflow Container

Use
You can define bindings from the workflow container to the following containers:

- Task container
- Role container
- Event container

You can allocate an expression to the elements of the target container, which refers to an element of the workflow container.

Features

Binding definition to task container
This binding can be defined for step types that refer to a task. The workflow system proposes this binding definition. If new elements are required in the workflow container, proposals are also made for these. You can edit and confirm these proposals or reject them. For further information, refer to Rule-Based Proposal for Binding Definitions.

Special information on object reference
A binding must generally be defined from the element in the workflow container with the reference to the object to be processed, to the element _WI_Object_ID in the task container. Only in this way can the right object be processed with the single-step task.

- If the workflow container contains an element which is defined with a data type reference to the same object type that is created in the referenced single-step task, this binding definition is proposed automatically when an activity is defined.
- If no element with an appropriate data type reference is available in the workflow container, a suitable element is added to the workflow container and the binding entered.

Binding definition to event container
This binding definition is only possible for the step type event creator.

- The binding definition to the event container is not necessary if the latter only contains the standard event parameters. These standard event parameters are assigned values automatically from the information you specify when defining the event creator.
- The binding definition to the event container is necessary if the latter contains elements other than the standard event parameters.

Binding definition to role container
This binding definition is only necessary if you use a role for establishing the agents or recipients.
Activities
For further information, refer to Operation of the Binding Editor [Seite 1232].
Binding Definitions from Task Container

Use
You can define bindings from the task container to the following containers:

- Workflow container
- Role container
- Method container

You can allocate an expression [Seite 1235] to the elements of the target container, which refers to an element of the task container.

Features

Binding definition to workflow container
The binding is applied after execution of the work item. The binding can be defined in step type definitions that refer to a task. It is necessary if you require information for the workflow, which is available after execution of the underlying object method.

Special information on object reference
If an object is created (for example, with the method Create) or established (for example, with the method Find) in task processing, you must define the binding from the element _WI_Object_ID in the task container to the element in the workflow container, which can contain the object reference. Only then is this information available in the workflow and therefore for the subsequent steps.

If the workflow container contains an element which is defined with a data type reference to the same object type that is created in the referenced single-step task, this binding definition is proposed automatically when an activity is defined.

If no element with an appropriate data type reference is available in the workflow container, a suitable element is added to the workflow container and the binding entered.

Current agent: Element _WI_Actual_Agent in task container
If you want to transfer the current agent, available under the element ID _WI_Actual_Agent in the task container, to the workflow container for further use, you define an appropriate element in the workflow container to contain this information.

You create this element with a reference to the ABAP Dictionary field WFSYST-AGENT.

Result of the object method: Element _WI_Result in the task container
The result of the object method is available under the element ID result(_WI_Result) in the task container. You must define an appropriate element in the workflow container, if you want to transfer the result into the workflow container. When defining the container element, use the same data type reference that the element _WI_Result has in the task container.

You require the result of this activity in a subsequent step of the type loop (UNTIL) in order to formulate the loop condition.
Binding Definition from Task Container

**Binding definition to role container**
You define this binding in the definition of a single-step task if you enter roles on the tab page *Default roles*. 
You must add the elements required to resolve the role to the task container in order to define a binding between these new elements and the appropriate elements of the role container.

Note that you must create a binding from the workflow container to the task container in the step definition, via which these "new" container elements are given their values.

**Binding definition to method container**
You define this binding in the definition of a single-step task. It is only necessary if import parameters are defined for the object type method.
The workflow system proposes this binding definition if you have specified an object method *with parameters* during definition of the single-step task. If new elements are required in the task container, proposals are also made for these. You can edit and confirm these proposals or reject them. For further information, refer to [Rule-Based Proposal for Binding Definitions (Seite 1221)](Seite_1221).
You can modify this binding definition. But this will usually only be necessary in exceptional situations.

**Catering for the container elements _WI_Object_ID**
The container element _WI_Object_ID in the task container is not incorporated into the binding definition to the method container. At runtime, the workflow system ensures that the object reference from the element _WI_Object_ID is available in the key fields in the program of the object type.

**Activities**
For further information, refer to [Operation of the Binding Editor (Seite 1232)](Seite_1232).
Binding Definitions from Event Container

Use
You can define bindings from the event container to the following containers:

- Task container
- Workflow container

You can allocate an expression [Seite 1235] to the elements of the target container, which refers to an element of the event container. The event may contain information that you want to use in the workflow. If so, you must read this information from the event container for the event. In order for the information that is transported as event parameters in the container for this event to reach the workflow, you can define the binding from the event container to the workflow container in the wait step.

Features

Binding definition to workflow container
This binding can be defined at the following places:

- In the definition of the multistep task on the tab page Triggering events
- In the Workflow Builder in the basic data of the workflow on the tab page Start
- In the Workflow Builder in the step definition of a wait step on the tab page Control

This binding enables you to transfer event parameters of a triggering event [Extern] or an event for which the workflow waits during execution into the workflow.

You can only use import elements of the workflow container in a binding definition from event parameters of a triggering event to elements of the workflow container.

Binding definition to task container
This binding can be defined in the definition of the task on the tab page Triggering events.

Activities

Assigning the object reference for triggering events
If you want to use the reference to the object whose status change the event reports in the task or workflow, you must assign this reference to an element of the task container or the workflow container. The object reference is located in the event container under the element ID _Evt_Object.

- In the task container, the reference to the object to be processed must always be located in the element _WI_Object_ID.

- No element is provided for the object reference in the workflow container as standard. You define this element in the workflow container with a data type reference [Extern] to the relevant object type and assign it the value of the element _Evt_Object from the event container in the binding definition.
Assigning the event creator for triggering events

If you want to use the information about the user under whose responsibility the event was created in the task or workflow, you must assign this information to an element in the task container or workflow container.

The user name of the employee under whose responsibility the event was created is located in the event container under the element ID _Evt_Creator.

- No element is provided for this information in the task container as standard. If applicable, you define this element in the task container with a data type reference to the ABAP Dictionary field WFSYST-AGENT and assign it the value of the element _Evt_Creator from the event container in the binding definition.

- The element _WF_Initiator is provided for this information in the workflow container. In the binding definition, you assign this element the value of the element _Evt_Creator from the event container.

For further information, refer to Operation of the Binding Editor [Seite 1232].
Binding Definition from Method Container

Use
You can define a binding to the task container.
You can allocate an expression [Seite 1235] to the elements of the task container, which refers to an element of the method container.

Features
You define this binding in the definition of a single-step task. It is only necessary if export parameters are defined for the object type method (only for synchronous methods [Extern]). The binding from the method container to the task container is executed when a synchronous object method has been processed successfully.
The workflow system proposes this binding definition if you have specified an object method with parameters during definition of the single-step task. If new elements are required in the task container, proposals are also made for these. You can edit and confirm these proposals or reject them. For further information, refer to Rule-Based Proposal for Binding Definitions [Seite 1221].
You can modify this binding definition. But this will usually only be necessary in exceptional situations.

Catering for the container elements _WI_Object_ID and _WI_Result
The container elements _WI_Object_ID and _WI_Result of the task container are not incorporated into the binding definition between the method container and the task container.
At runtime, the system ensures the following:

- If an object was created for a method, the reference to the object created is entered in the container element _WI_Object_ID.
- The result (if applicable) of the method is entered in the container element _WI_Result.

Activities
For further information, refer to Operation of the Binding Editor [Seite 1232].
Binding Definition for Document from Template

Use
In the binding definition you enter elements of the workflow container, which are to either pass values to the task container or receive values from the task container after processing.
Binding definitions are only necessary for the step type Document from template under the following circumstances:

- If you use your own single-step task
- If you have used expressions from the task container in the title or in the task texts of the single-step task
- If an attachment is to be created based on a document template
- If the creator of the document is still required in the workflow

The binding for the document templates required, which you specified on the tab page Document templates, is defined automatically. The binding from the workflow container to the task container for the element Flowitem and the binding from the task container to the workflow container for the element _WI_Object_Id are also defined automatically.

Features
The following container elements in the task container are particularly significant for this binding:

_\_WI_Actual_Agent (actual agent)
If you want to transfer the agent for the document generation to the workflow container for further use, you define an element with the ABAP Dictionary reference WFSYST-AGENT there to hold this information.

Note_Reference (template for notes)
If you assign a workflow container element containing a template (business object reference SOFM) to this container element, the agent can enter a document as an attachment at runtime, which is based on this template.

Activities
For further information, refer to Operation of the Binding Editor [Seite 1232].
Binding Definition for Decision Tasks

Use

In the binding definition you enter elements of the workflow container, which are to either pass values to the task container or receive values from the task container after processing.

For a user decision, a binding definition is only necessary from the workflow container to the task container of the user decision under the following circumstances:

- If you use your own decision task for the user decision
- If you have used expressions from the task container in the title or in the task texts of the decision task
- If an attachment is to be created based on a document template

A binding definition from the task container to the workflow container is only required for a user decision if you want to store the result or the current agent of the user decision in an element in the workflow container in order to access it again in a subsequent step.

Features

The following container elements in the task container are particularly significant for this binding:

- **_WI_Result (result)**
  
  The result of the user decision is available in this element as a text field after processing. Depending on the number and sequence of possible decision options, these are the values 0001, 0002 up to nnnn. A binding definition is only required for this element if you need this value in the remainder of the workflow definition. This may be the case, for example, if you need the value again as an indicator in a decision. To implement this, you define an appropriate element (ABAP Dictionary reference SWD_LINES–RETURNCODE) in the workflow container to hold this user decision result.

- **_WI_Actual_Agent (actual agent)**
  
  If you want to transfer the agent for the user decision to the workflow container for further use, you define an element with the ABAP Dictionary reference WFSYST–AGENT there to hold this information.

- **Note_Reference (template for notes)**
  
  If you assign a workflow container element containing a template (business object reference SOFM) to this container element, the agent can enter a document as an attachment at runtime, which is based on this template.

Activities

For further information, refer to Operation of the Binding Editor [Seite 1232].
Operation of the Binding Editor

Use
The binding editor is used to define bindings between two containers. In the case of binding definition between task container and workflow container as well as between task container and method container, you can process both directions of binding at the same time. You can only process one direction of binding for the remaining binding definitions.

Integration
Each binding definition is linked to the execution of an action which processes the data provided and produces results that can be returned. The following actions can be carried out:

- Execution of a step
- Execution of a single-step task or multistep task
- Execution of an object type method
- Resolution of a role

Features
The container elements of the target container of the binding definition are listed on the left-hand side. You can use a button to switch between displaying the technical name and displaying the element name. Only the following container elements in the target container are displayed as standard:

- Container elements with import or export indicator set, which are not workflow system elements
- _WI_Object_Id

In order to display the remaining workflow system elements in the target container as well, you select a button. The binding definition from the source container to the target container (binding direction) is processed in the upper part of the screen. The associated binding is executed at runtime before the action. This binding definition is always available.

The binding definition from the target container to the source container (binding direction) is processed in the lower part. The associated binding is executed at runtime after the action. This binding definition is only available if the action can produce results that can be made available to the source container.

Binding Definition from Source Container to Target Container
Here you process the allocation of values to elements in the target container. Only the elements in the target container for which the indicator Import is set are displayed. You can allocate an expression to each element of the target container.

For further information, refer to Binding Definitions from the Source to the Target Container.

Binding Definition from Target Container to Source Container
Here you process the allocation from elements in the target container. Only the elements in the target container for which the indicator Export is set are displayed. You can allocate a container element in the source container to each element of the target container. You can allocate the container element in the source container to a compatible container element in the target container.
To a field of a structure, if the container element of the target container is defined with structure reference. You can allocate one element in the source container to several elements in the target container. To make the source element available several times as a starting point for the binding definition, you select the source element and choose ▶. For further information, refer to Binding Definitions from the Target Container to the Source Container [Extern].

Definitions of containers involved
You can call up the full definitions of the two containers involved by using the pushbuttons with the same names. Important information is displayed here about the data type references of the container elements.

Permitted entries for expressions
Permitted expression types for binding directions

<table>
<thead>
<tr>
<th>Expression type</th>
<th>Permitted binding direction</th>
<th>Explanation</th>
</tr>
</thead>
<tbody>
<tr>
<td>&amp;&lt;Element&gt;&amp;</td>
<td>← and →</td>
<td>Source and target element with compatible field references</td>
</tr>
<tr>
<td>&amp;&lt;Object.Attribute&gt;&amp;</td>
<td>←</td>
<td>Source and target element with compatible field references</td>
</tr>
<tr>
<td>&amp;&lt;Object.Object Attribute&gt;&amp;</td>
<td>←</td>
<td>Source and target element with compatible field references</td>
</tr>
<tr>
<td>&amp;&lt;Object&gt;&amp;</td>
<td>← and →</td>
<td>Source and target element with compatible object references</td>
</tr>
<tr>
<td>&amp;&lt;Object.Object&gt;&amp;</td>
<td>←</td>
<td>Source and target element with compatible object references</td>
</tr>
<tr>
<td>&amp;&lt;Structure&gt;&amp;</td>
<td>← and →</td>
<td>Source and target element with compatible structure references</td>
</tr>
<tr>
<td>&amp;&lt;Structure&gt;-&lt;Table-Field&gt;&amp;</td>
<td>← and →</td>
<td>Source and target element with compatible field references</td>
</tr>
<tr>
<td>Const</td>
<td>←</td>
<td>Assignment of a constant</td>
</tr>
</tbody>
</table>

It is not possible to assign the value of an element in the target container to an object attribute in the source container.
Alternative Binding Definition

Use
As an alternative to describing a binding in the binding editor, you can define a binding by specifying a function module. This type of binding definition is particularly useful as an alternative to the line-oriented binding definition described up to now, if:

- You work on both sides of the binding definition with structure references and the binding editor does not enable the desired assignments
- You want to perform calculations or checks on the container elements before assignment
- You want to combine two or more container elements in some way before assignment

Features
To execute the binding, the system calls a function module, to which the two containers are passed as table parameters. This function module is programmed especially for this binding and is entered when the alternative binding is defined.
The binding definition based on container elements is deleted.

Activities
You define an appropriate function module for each binding direction. The interface for these function modules is defined. Use the function module SWA_TEMPLATE_EXECUTE_DATAFLOW as a template. The function module documentation contains all the required information on the interface and programming.
To enter the function modules, you choose *Alternative binding* in the binding editor context menu and make the entries in the relevant input field.
Using Expressions in Binding Definitions

Use
You use an expression to assign a value to a container element in the binding definition.

Features
Expressions must be assigned using the F4 input help. You have the following options for assigning values:

Constant
Fixed value or fixed object reference, which is assigned to a container element at runtime.

Container
Direct assignment of a container element or assignment using a container element from the source container.

- The value of the variable is derived directly from the container element
  
  The variable is created in the form &<ElementName>&

- The value of the variable is derived indirectly as an attribute or a key field of the object referenced in the container element.

  The variable is specified in the form &<ElementName>.<AttributeName>&.

  This indirect establishment of values can also have two or more levels. The variable then appears in the form &<ObjectReference>.<ObjectReference>.<...>.<AttributeName>&.

- The value of the variable is derived indirectly as a field value of the structure that is referenced in the container element.

  The variable is specified in the form &<ElementName>.<Table>-<Field>&.

System field
Field in table SYST.

The variable is specified in the form %<SystemFieldName>% Depending on the environment, not all the expression components named are always offered.

Activities
Use the F4 input help when making assignments. Bear in mind the color legend. The colors indicate whether an assignment is permitted as far as the data types of the two partners are concerned, and whether the assignment may be defined. As an alternative to the "technical" element ID, you can display the name of the container element by choosing .

Activity
Choose between constant, container and system field to specify the type of expression. If you specify the expression as the target of a binding definition, you can only choose container.

Entries can occur twice when container elements are displayed, the second entry having an addition.

Addition "with index"
The addition "with index" only appears for a multiline container element if:

- The container element is a source element of a binding definition.
- The container element is entered as a multiline container element on the tab page Other. If this entry is chosen, only one line of the multiline container element is assigned to the single-line target element in the binding. For further information, refer to Binding Definitions from Source Container to Target Container [Extern].

This application should only be used within dynamic parallel processing.

**Addition "add instead of change"**
The addition “add instead of change” only appears for a multiline container element, if the container element is a target element of a binding definition. This addition means that the value of the source element is appended to the content (if any) of the target element. For further information, refer to Binding Definitions from Target Container to Source Container [Extern].
Document Processing

Use
You can use a document in the following ways in SAP Business Workflow:

- Send it as a "mail" from a workflow
- Create, display or edit it in a workflow step
- Add it to a workflow as an attachment

Features

Sending or displaying a text created at definition time
A text entered at definition time of a workflow and supplemented at runtime with information from the context of the workflow can be sent as a mail, and is displayed in the work item preview of the Business Workplace.

Adding a document to a workflow as an attachment
A document can be assigned to a workflow as an attachment. The recipients of the work items in this workflow can display this attachment in the work item preview.

Processing standard texts in a workflow
A SAPscript document can be created, edited or displayed in a workflow step. It can be supplemented with information from the task container.
Processing SAPscript Texts as a Step in the Workflow

Use

SAPscript standard texts can be entered and displayed at runtime as part of a workflow.

Integration

The object type STD_TEXT (SAPscript standard text) is defined in the Business Object Repository with methods for creating, editing and printing.

Activities

If you want to enter a text as a step in a workflow at runtime and this text is to be read by another employee in one of the subsequent steps, the following activities are required:

1. Define a single-step task for entering a text. The user who executes the work item in question at runtime has to enter a text. For further information, refer to Defining the Standard Task "Create SAPscript Text" [Seite 1239].

2. Define a single-step task for displaying the text. For further information, refer to Defining the Standard Task "Display SAPscript Text" [Seite 1240].

3. Include both steps into a workflow definition. For further information, refer to Defining the Workflow (Section) "Process SAPscript Text" [Seite 1241].

Procedure at runtime

At runtime the recipient of the work item enters a SAPscript text in the first step. Text variables can be used to reference values from the task container of this single-step task. Knowledge of the task container is required to use this variable replacement option.

This text is displayed to the user who executes the work item in the second step.

Printing standard texts

The following methods of the object type STD_TEXT enable users to print the standard texts, and can be used in single-step tasks:

- ReplaceAndPrintBatch (Print text with symbol replacement in background)
  - The printer and print attributes must be passed as parameters.

- ReplaceAndPrint (Replace text symbols and print text)
Defining the Standard Task "Create SAPscript Text"

2. Select Standard task in the list field Task and then ▶️.
3. Enter the following data:
   - Abbreviation: Any abbreviation from your namespace [Extern]
   - Description: Any task description
   - Work item text: Any text to identify the work item in the Business Workplace
   - Object type: STD_TEXT
   - Method: CREATE

4. Select ▶️.

Agent assignment

5. Choose Additional data → Agent assignment → Maintain to specify the possible agents [Extern] for this single-step task.
6. Select ▶️.

Result

You have defined a single-step task that enables a SAPscript text to be created when it is executed. This creates an object of the type STD_TEXT (SAPscript standard text). The reference to this object is stored in the container element _WI_Object_ID of the task container after execution of the method. If you use this task as an activity in a workflow definition, you must pass this reference in the binding from the task container to the workflow container. To do this you require a container element in the workflow container with a data type reference to the object type STD_TEXT.

This text is displayed in another single-step task Display standard text. Therefore this single-step task should really only be used as a step in a workflow definition.
Defining the Standard Task "Display SAPscript Text"

**Prerequisites**
A SAPscript text has been entered and the object reference is known in the workflow.

**Procedure**
8. Select Standard task in the list field Task and then.
9. Enter the following data:
   - Abbreviation: Any abbreviation from your namespace [Extern]
   - Description: Any task description
   - Work item text: Any text to identify the work item in the Business Workplace
   - Object type: STD_TEXT
   - Method: REPLACEANDDISPLAY
10. Select.

**Agent assignment**
11. Choose Additional data → Agent assignment → Maintain to specify the possible agents [Extern] for this single-step task.
12. Select.

**Result**
You have defined a single-step task that enables a SAPscript text to be displayed when it is executed. The text to be displayed must be an object reference to an object of the type STD_TEXT in the container element _WI_Object_ID of the task container.
Defining the Workflow (Section) "Process Standard Text"

Prerequisites
The single-step task is defined, with which the method Create (object type STD_TEXT) is created. For a description of the procedure, refer to Defining the Standard Task "Create SAPscript Text" [Seite 1239].
The single-step task is defined, with which the method ReplaceAndDisplay (object type STD_TEXT) is executed. For a description of the procedure, refer to Defining the Standard Task "Display SAPscript Text" [Seite 1240].

Procedure
Add the following activities to the workflow definition. The two activities do not need to be consecutive.

Activity for creating a text
1. Define an activity in which you enter the single-step task for the method Create (object type STD_TEXT).
   This displays the dialog box Define container elements and binding, on which proposals for a binding and for new container elements of the workflow container are displayed.
2. Select ✓.
   The system creates the container element STANDARDTEXT in the workflow container, in which the object reference of the SAPscript text created is stored. A binding is defined from the container element _WI_Object_ID in the task container to the container element STANDARDTEXT in the workflow container for this.

Activity for displaying a text
1. Define an activity in which you enter the single-step task for the method ReplaceAndDisplay (object type STD_TEXT).
   This displays the dialog box Define container elements and binding, on which proposals for a binding and for new container elements of the workflow container are displayed.
2. Select ✓.
   The single-step task requires the object reference of the SAPscript text to be displayed.
   The system defines a binding from the container element STANDARDTEXT in the workflow container to the element _WI_Object_Id in the task container.
Specifying Agents of Subsequent Steps Dynamically

Use

The actual agent of the step Determine agents in a workflow can specify the recipients who are to receive the work item of the subsequent step Any activity in the same workflow. The step Any activity does not have to follow the step Determine agents directly.

You can only specify users as recipients if they are possible agents of the relevant task.

Use ad hoc agent assignment only if the user who starts the workflow is to specify recipients at runtime.

Features

You can use the wizard Include "Choose agent" to extend the workflow definition. In this wizard you specify the task for which you want to select recipients. The wizard extends the workflow definition by adding two steps:

- Activity Selection of agents for subsequent task
- Activity that references the subsequent task you specify

Activity Selection of agents for subsequent task

The system enters the workflow initiator as responsible agent for this task. The actual agent of this work item selects the recipients from the possible agents of the subsequent task. A list of the possible agents of the subsequent task is displayed if it is not a general task. If the subsequent task is a general task, the actual agent can select the recipients from all users of the system. The recipients are transported with a binding into a multiline workflow container element created by the system.

Activity that references the subsequent task you specify

The responsible agents of the subsequent task are read by the system from container element created.

The graphic below illustrates the extension of the workflow definition and the bindings created.
Specifying Agents of Subsequent Steps Dynamically

### Specifying Agents of Subsequent Tasks

<table>
<thead>
<tr>
<th>Activity</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Select agents for subsequent task</td>
<td>Responsible agent</td>
</tr>
<tr>
<td>Agents selected</td>
<td>Subsequent task</td>
</tr>
<tr>
<td>Task executed</td>
<td>Workflow container</td>
</tr>
</tbody>
</table>

#### Activities

Execute the wizard Include "Choose agent". You have to edit the steps created if:

- The recipients are not to be selected by the workflow initiator
- Deadlines are to be defined for the activities
- Other outcomes are to trigger a response in the workflow
- The work item display is to be formatted individually

You need your own task for the activity Selection of agents for subsequent task if, for example, you want to enter your own texts for the task.

For further information, refer to **Copying a Selection Task [Seite 1244]**.
Copying a Selection Task

Use

The wizard uses the standard task TS30200146 as a selection task. If you want to use another task, create it as a copy of the standard task.

The standard task references the object type WF_TASK [Seite 1163].

Procedure

1. Copy the standard task TS30200146. For further information, refer to Creating and Maintaining Tasks [Seite 1173].
2. Open the task in change mode and make your changes.
   Select No if the system asks whether missing elements should be taken from the object method.
3. Go to the Workflow Builder and use the wizard to insert dynamic agent assignment into your workflow definition.
4. Open the step definition of the activity Selection of agents for subsequent task created by the wizard.
5. Go to the binding editor and note the binding created by the wizard.
6. Replace the standard task TS30200146 used by the wizard with your copy.
7. Define the same binding for the new task as the one the wizard created for the standard task TS30200146.
Copying Tasks into Another Plan Version

Use

You can copy customer tasks [Extern] and workflow tasks [Extern] to other plan versions.

Procedure


   The Copy plan version dialog box appears.

2. Enter the source and target plan version.

3. Enter the task type to be copied.

4. Select 🗓.
Workflow Builder

Use
This is the main *SAP Business Workflow* tool for creating, displaying and processing workflow definitions. You can test workflow definitions and generate operable versions.

Integration
The Workflow Builder provides a graphical view of the workflow definition. The alphanumeric workflow editor provides a textual view of the workflow definition in a tree display, as an alternative to the Workflow Builder.

Prerequisites
Each workflow definition has one multistep task. While the definition of the multistep task regulates the external relationship (call interface, triggering events), the workflow definition describes the actual implementation of a business scenario as a sequence of connected steps.

Features
The steps, outcomes and triggering events of the workflow definition are displayed graphically with their interconnections. You can choose the type of display in the Workflow Builder options.

The Workflow Builder screen is divided into the following areas.

Workflow
Here you can insert new steps into the workflow definition and process existing ones. Double-clicking on a step calls the associated step definition.
Overview

The overview graphic is displayed here. The part of the workflow graphic displayed in the screen area *Workflow* is marked with a green rectangle. Changing the size or position of the rectangle changes the display in the screen area *Workflow*.

Step types

All step types that you can use in the workflow are displayed here if you are in change mode. To insert a new step into the workflow, you click on the step type and insert it into the workflow definition.

Tasks

This screen area (which you can deactivate) can be used to display tasks in order to use them in the workflow definition. A displayed task can be inserted into your workflow definition as an activity by double-clicking. The binding between the task container and the workflow container as well as the other container elements required is generated automatically by the workflow system as far as possible.

Objects

The objects of the workflow are displayed here depending on the display of the workflow definition. The container elements of the workflow container and the document templates of the workflow are also displayed. You can change an object or create new ones directly from the list. A where-used list is available for document templates and container elements.

Messages

All messages generated in where-used lists and workflow tests are displayed here. Double-clicking on a message takes you to the relevant step definition.

For further information, refer to *Workflow Builder Functions*.[Seite 1006]
Calling the Workflow Builder

Procedure

You can call the Workflow Builder in two ways:

- **Direct start:** Choose **Tools → Business Workflow → Development → Definition tools → Workflow Builder**
- **Start from task:** From a screen for processing or displaying a multistep task

**Direct start**

This loads the workflow definition you last saved. If you want to switch between change mode and display mode, choose 🍀. To load another version of the workflow definition or a new workflow definition, you have the following options:

- If the workflow definition is displayed in the task area in the folder *Workflows last processed*, the version number of the workflow definition is also specified in parentheses. To load it, select the multistep task and choose **Load workflow definition** in the context menu.

- If the multistep task is displayed in a different folder in your task area, you can select the version that you want to load in the dialog box *Open another workflow definition*. Select the multistep task and choose **Load workflow definition** in the context menu.

- If you want to load another workflow definition, select 📜. In the dialog box *Open another workflow definition*, you can select the workflow and version that you want to load.

- If you want to create a new workflow definition, choose **Workflow → New**. You make specifications for the multistep task when the workflow definition is first saved.

**Start from task**

If you start the Workflow Builder from a screen for processing or displaying a multistep task, choose **Workflow Builder**. The active version of the workflow definition is loaded. If there is no active version, the version with the highest version number is loaded.

To call another version of the workflow definition, choose **Goto → Workflow Builder → Choose version**. This takes you to a dialog box in which you can choose a version of the workflow definition. For further information, refer to [Versions of a Workflow Definition][1].

**Result**

The workflow definition is displayed and can be processed as applicable. If the Workflow Builder is being called to process a newly-created, initial workflow definition for the first time, the following representation is displayed:

- The start of the workflow definition is indicated by 🌟 **Start workflow**. If the new workflow definition was called from the definition of a multistep task in which triggering events [Extern] are defined, these are represented by the symbol 🔄 and their description.

- The end of the workflow definition is indicated by 🌟 **Workflow terminated**.

- The area in which the new workflow definition can be inserted is indicated by an undefined step with an outcome. Steps are represented by symbols. The name of an outcome is represented in the standard view on the arrow.

---

[1]: #Versions_of_a_Workflow_Definition_Seite_1019
# Step Types and Their Symbols

<table>
<thead>
<tr>
<th>Symbol</th>
<th>Step type</th>
<th>Runtime function</th>
</tr>
</thead>
<tbody>
<tr>
<td>🔄</td>
<td>Process control</td>
<td>The functions <em>Cancel work item</em> or <em>Set work item to obsolete</em> are applied to other work items of the current workflow.</td>
</tr>
<tr>
<td>🔄️</td>
<td>Activity</td>
<td>Execution of a task. This task can be a single-step task or a multistep task.</td>
</tr>
<tr>
<td>🔄️️</td>
<td>Condition</td>
<td>One of two branches defined in the workflow definition is processed. The system makes the decision based on defined conditions. When doing this, this system takes account of information from the context of the workflow or the application objects processed.</td>
</tr>
<tr>
<td>🔄️️️</td>
<td>User decision</td>
<td>The process flow of the current workflow is controlled on the basis of a decision made by the current agent.</td>
</tr>
<tr>
<td>🔄️️️️</td>
<td>Document from template</td>
<td>A document is created from a document template.</td>
</tr>
<tr>
<td>🔄️️️️️</td>
<td>Container operation</td>
<td>Elementary arithmetic operations or value assignments are applied to individual elements of the workflow container.</td>
</tr>
<tr>
<td>🔄️️️️️️</td>
<td>Event creator</td>
<td>An event is created.</td>
</tr>
<tr>
<td>🔄️️️️️️</td>
<td>Multiple condition</td>
<td>One of several branches defined in the workflow definition is processed. To do this, the system checks the value of a container element in the workflow container.</td>
</tr>
<tr>
<td>🔄️️️️️️️</td>
<td>Fork</td>
<td>The processing that follows takes place in parallel branches. You can define how many branches have to be processed for the fork to be successfully completed or define an end condition in the condition editor.</td>
</tr>
<tr>
<td>🔄️️️️️️️</td>
<td>Undefined step</td>
<td>No function at runtime. Undefined steps always have an outcome.</td>
</tr>
<tr>
<td>🔄️️️️️️️</td>
<td>Loop (UNTIL)</td>
<td>A sequence of steps is processed repeatedly until the defined termination condition occurs.</td>
</tr>
<tr>
<td>🔄️️️️️️️</td>
<td>Wait for event</td>
<td>The system waits for a specific event. The work item is only completed if the expected event occurs.</td>
</tr>
<tr>
<td>🔄️️️️️️️</td>
<td>Loop (WHILE)</td>
<td>A sequence of steps is processed repeatedly as long as the defined comparisons apply. The system leaves the loop when the basis of comparison does not agree with any of the comparison values.</td>
</tr>
</tbody>
</table>
Workflow Builder Functions

Use
You need these functions for the following activities:
- Adding, reassigning or removing steps in a workflow definition
- Processing Steps
- Checking, testing or printing the workflow definition
- Processing another workflow definition
- Processing objects in the object area

Features
If not otherwise specified, you will find these entries in the context menu of a selected node.

Block operations: Cut, copy, paste, delete
The functions always refer to the complete block in which the selected node is located, including all subordinate blocks in the hierarchy. Therefore, always consider the scope of a block. Loops and forks, in particular, can cover large sections of your workflow definition. If necessary, choose Graphic → Blocks → Show to see which steps are in the block.

⚠️ You can only change or delete an outcome in the step definition of the preceding step. If you choose Change for an outcome, only the name is changed.

Functions for changing, creating and displaying a step
If you choose Change or Display, you branch to the step definition.
If you want to change the step type of the step, you must delete the old step and insert a new step of the desired step type. To change an undefined step into another step type, select the undefined step by double-clicking and choose the new step type.
For further information, refer to Creating Steps [Seite 1008].

Inserting Multistep Tasks
The workflow definition of the chosen multistep task is inserted into the current workflow definition at the selected point. Triggering events of the inserted workflow are not transferred. Choose Edit → Insert multistep task.

Undoing changes
You can undo up to ten changes. Select 🔄 and in the list the action up to which you want to undo changes.

Restoring changes
You can restore changes that have been undone. Select 🔄.

Where-used list
The Workflow Builder has two where-used lists:
- You can generate a where-used list for container elements of the workflow container and for document templates. To do this, you select the container element or the document template in the object area and choose Where-used list in the context menu. All nodes in which the
object is used are listed in the message area. From here you can branch to the relevant step definition by double-clicking on a usage.

- You can create a where-used list for tasks listed in the task area. To do this, you select the task and choose Where-used list in the context menu. All workflow definitions in which the task is used within a step definition are displayed.

**Processing objects in the object area**

You can display the object definition for all objects displayed in the object area. If you are in change mode, you can process all objects except the workflow system variables in the workflow container. For further information on processing container elements, refer to [Definition of Workflow Container](#). When you create a new document template, the application last used for a document template is started. You can use the function Change document class to start a different application. For further information, refer to [Creating Document Templates](#).

**Checking the workflow definition**

You select to check the workflow definition. All problems recognized are classified as errors or warnings and are output in the message area. You can process the node in which the error occurs if you double-click on the message. Workflow definitions that contain errors cannot be activated. For further information, refer to [Versions of a Workflow Definition](#). You can also start the workflow definition test from the display mode. The active version of the workflow definition is used for the test.

**Testing the workflow definition**

You select to execute the workflow definition on a test basis. This classifies the multistep task temporarily as a general task and saves and activates the workflow definition. If no error occurred, the multistep task is started in dialog. For further information, refer to [Start Workflow (Test Environment)](#).

**Printing the workflow definition**

For more information, refer to [printing workflow definition](#).

**Processing another workflow definition**

In the Workflow Builder, you can go directly into the processing of another workflow definition. If the workflow definition that you want to process is displayed in the task area, select it and choose Load workflow definition in the context menu. If it is not displayed, select . You can then choose the desired workflow definition.
Create Step

Use
You create a step in a workflow definition in order to execute tasks or to control the workflow, for example. A step is always created in a block [Extern]. This block contains at least the created step and its defined outcomes.

Procedure
You create a new step in the workflow definition by either changing an undefined step or creating a new step between two existing nodes.

Changing undefined steps
1. Open the dialog box Choose Step/Operator by double-clicking on the undefined step.
2. Choose a step type.

Creating new steps
1. Select a step or an outcome in the workflow area. Consider the positioning rules [Extern] for new steps.
2. Choose Create in the context menu. The dialog box Choose Step/Operator opens.
3. Choose a step type.

Inserting tasks from the task area
You can insert a task displayed in the task area into the workflow definition as an activity.
1. Select a step or an outcome in the workflow area. Consider the positioning rules [Extern] for new steps.
2. In the task area, select the task that you want to insert.
3. In the context menu, choose Include as an activity.

If you have selected a workflow definition (multistep task), you can also choose Include workflow definition. The steps of the selected workflow definition are then inserted.

Result
If you have created a new step, a step of the chosen step type is inserted at the chosen position/ replaces the undefined step. You go directly to the step definition. Undefined steps are also inserted in the workflow definition for some step types.
For further information, refer to Step Types and Their Symbols [Seite 1005].
If you have inserted a task from the task area, an activity which references the chosen task is inserted in the workflow definition.

The binding is created automatically by the system and you can adapt it to suit your requirements.
Special Functions in the Workflow Builder

Use
These functions are for processing a workflow definition in the Workflow Builder. The way the functions described are used depends on the options set [Seite 1016].

Features

Drag and drop
You can use drag and drop to create a new step. To do this, you select the desired step type in the step type area. This moves the cursor to the center of the work area and changes its form. You then choose a step or outcome [Extern], which inserts the new step in accordance with the positioning rules [Extern].

Double-click
Double-clicking on the various steps and outcomes in the workflow area has different effects:

<table>
<thead>
<tr>
<th>Double-click on</th>
<th>Effect</th>
</tr>
</thead>
<tbody>
<tr>
<td>Undefined step or its outcome</td>
<td>A new step is inserted</td>
</tr>
<tr>
<td>Any other type of step</td>
<td>The step definition is opened.</td>
</tr>
<tr>
<td>Outcome</td>
<td>The name of the outcome can be edited and, depending on the outcome type, additional information about the outcome is displayed.</td>
</tr>
</tbody>
</table>

If you have selected a step or an outcome in the workflow area, double-clicking on a task in the task area will transfer it into the workflow definition as an activity in accordance with the positioning rules [Extern].

Showing blocks
The individual blocks [Extern] that make up the workflow definition can be displayed using various gray scales. To do this, choose Graphic → Blocks → Show.
Nested blocks are represented with darker gray scales. Blocks on the same level have the same gray scale. The blocks each have a heading derived from the name of their first step and displayed in a different color in the upper-left part of the block.

Aligning
This function can be used to reformat the display in the workflow area. To do this, you select ☐.
Inserting Multistep Tasks

Prerequisites
You can insert a workflow definition into the workflow definition currently being processed.
You insert a workflow definition (as with creating steps) according to the rules of block orientation.

Procedure
1. Select a step or an outcome. Consider the positioning rules [Extern].
The remaining procedure depends on whether the workflow definition to be inserted is displayed in the task area.

   **Variant A: Workflow definition displayed in task area:**
   2. Select the workflow definition to be inserted in the task area.
   3. In the context menu, choose Include workflow definition.

   **Variant B: Workflow definition not displayed in task area:**
   2. Choose Edit → Insert multistep task. The dialog box Insert workflow definition in workflow definition is opened.
   3. Choose the multistep task whose workflow definition you want to insert.
   4. Choose the version of this workflow definition.

   If the workflow definition to be inserted has container elements in the workflow container, which exist under the same name (but maybe a different type) in the workflow definition currently being processed, you are given the opportunity to rename these container elements. If you rename the container element, all the references throughout the workflow definition are also renamed.

Result
The workflow definition inserted has now become an indistinguishable part of the original workflow definition.

If you want to insert a workflow definition that is to remain recognizable as an independent unit, you must insert an activity [Seite 1026] with reference to a multistep task into your workflow definition.
Printing Workflow Definitions

Printing structures
Select to print the graphical representation of a workflow definition. You can specify the size and arrangement of the printed workflow definition in a dialog box.

If you select in the alphanumeric workflow editor, the textual structure of the workflow definition is printed. This corresponds to the display in the object area.

Printing details
Choose Workflow → Print → Details to print a list of all the steps involved in the workflow definition with their most important properties. A dialog box WF: Print workflow definition opens, in which you define the scope of the list.
Select to print the list directly, or to display the list on the screen first.
Condition Editor

Use
The subsequent flow of a workflow depends at various points on a condition. The evaluation of a condition always returns either the result true or the result false. You formulate the condition with the condition editor.

<table>
<thead>
<tr>
<th>Step type that uses the condition editor</th>
<th>Detail</th>
</tr>
</thead>
<tbody>
<tr>
<td>Condition [Seite 1061]</td>
<td>The result of the evaluation decides which of two modeled branches the workflow processes.</td>
</tr>
<tr>
<td>Loop (UNTIL) [Seite 1087]</td>
<td>The result of the evaluation decides whether the loop is processed again or not.</td>
</tr>
<tr>
<td>Fork [Seite 1089]</td>
<td>The result of the evaluation decides whether parallel processing is ended, even if the number of necessary branches is not yet reached.</td>
</tr>
</tbody>
</table>

Prerequisites
You are editing a step type that requires a condition. You start the condition editor by double-clicking in the condition table.

Features
There are two modes for working with the condition editor, which differ only in the display and the way you select expressions and operators. To change modes, you select 📦. The following modes are available for display:

Direct text entry
You can select the expressions and operators for constructing the condition using the F4 input help.

Mouse-oriented
The screen of the mouse-oriented condition editor is divided into the following areas:
Condition Editor

**Editing**
The current condition is displayed and can be edited here. Every condition has the general form:

\[
[\text{<Not>}] \text{ <Expression> <Operator> <Expression> [<And/Or> <Not> <Expression> <Operator> <Expression> ...]}
\]

**Expression**
Here you will find the following input options for expressions.

- System fields
- Container elements of the workflow container
  
  You can also access the fields of the referenced structure or the attributes of the referenced object via a container element.

**Operators**
The operators available are displayed here. When an operator is chosen, it is inserted into the current line of the condition and replaces an operator that is already there.

<table>
<thead>
<tr>
<th>Operator</th>
<th>Meaning</th>
<th>Remarks</th>
</tr>
</thead>
<tbody>
<tr>
<td>=</td>
<td>Equals</td>
<td></td>
</tr>
<tr>
<td>≠</td>
<td>Does not equal</td>
<td></td>
</tr>
<tr>
<td>&lt;</td>
<td>Less than</td>
<td></td>
</tr>
</tbody>
</table>
Logical operators

<table>
<thead>
<tr>
<th>Operator</th>
<th>Meaning</th>
<th>Remarks</th>
</tr>
</thead>
<tbody>
<tr>
<td>AND</td>
<td>And</td>
<td></td>
</tr>
<tr>
<td>OR</td>
<td>Or</td>
<td></td>
</tr>
<tr>
<td>NOT</td>
<td>Not</td>
<td>The whole condition line is evaluated and the result negated.</td>
</tr>
</tbody>
</table>

**NOT** is stronger than **AND**, and **AND** is stronger than **OR**.

A condition of the form **A OR B AND C OR D** is equivalent to **A OR (B AND C) OR D**

**Constants**

You can enter a constant in this input field for expression 2. The format of the constant entered depends on expression 1.

**Parentheses**

You can structure the condition by using parentheses and activate the display of the contents of the parentheses selectively.

The mouse-oriented mode requires a 32-bit platform.
Activities

Editing conditions
The description below applies to operation in mouse-oriented mode.
The expression you can enter or change is selected using . To create or edit a condition, you first select the relevant expression field in the condition by double-clicking and then define the content.
Usually at least one of the two expressions refers to a container element of the workflow container.
If you choose an operator or a logical operator, it is transferred automatically into the line in which the selection is located. If you delete a condition line, the logical operators AND and OR of the previous line are also deleted. If you select a NOT operator that already exists in a line again, it is deleted.

Using parentheses and comments
To insert parentheses into an existing condition, you select the condition lines around which parentheses are to be inserted and choose ( ).
You can change the comment for parentheses. Note that this text is not translated.

Expanding or collapsing parentheses
You can display or hide the contents of all or individual pairs of parentheses. The position of the cursor has the following effects:
- In the input field outside all parentheses: All parentheses are affected
- Inside parentheses or in a line with parentheses: One pair of parentheses is affected
To hide the content of parentheses you select ( ) and to display it ( ).

Syntax check
To check the condition for syntax errors, you select .

Testing conditions
You can test your condition at definition time.

Test data
You create the test data required as follows:
- Use real data from a work item container. To do this, choose Retrieve test data for work item in the context menu.
- Enter the values directly. To do this, choose Enter test data directly in the context menu.

Testing
Choose Evaluate condition with test data in the context menu.
After the test you can use Test analysis to see how the test result arose, and Display test messages to display the test messages again.
Workflow Definition Extras

Features

The following functions are located in the *Extras* menu in the Workflow Builder:

- **Workflows of this definition**
  
  The system calls the work item selection [Seite 1490]. The current multistep task is preset. Define the search period.

- **Business Workplace [Seite 1368]**
- **Business Object Builder [Seite 1100]**
- **Create event [Seite 1365]**
- **Technical information**
  
  The technical information contains the table entries with the definition and runtime structures on which the workflow definition is based.

  If you contact SAP about any errors that may have occurred, you should have this information to hand.

- **Expand subworkflows**
  
  Subworkflows used in the workflow definition are inserted directly into the workflow definition. If there are naming conflicts between the container elements of the existing workflow container and the container elements to be inserted, the system informs you and gives you the opportunity to rename these container elements.

- **Translation**
  
  Here you can translate all language-dependent texts that appear in the workflow definition. This applies to:

  - Step and outcome names
  - Decision texts and titles
  - Container element names

- **Ad hoc functions**

  You can use this to change the workflow type to *ad hoc*. For further information, refer to *Ad Hoc Workflow [Seite 1095]*.

- **Special functions**

  These enable you to correct the container structure or block structure if the system points out problems.

- **Call workflow configuration**

  This enables you to display of the configuration of the workflow definition that is currently loaded. For further information, refer to *Workflow Configuration [Seite 1171]*.

Effects of setting options

The tab page *General* is used to specify whether the tips and tricks are to be displayed when the Workflow Builder is started.
The tab page *Graphic* is used to determine which view is used for the display:

- **Without event-driven process chains**
  All steps, triggering events and the start and end of the workflow are displayed with symbols. Outcomes are displayed in text on the connecting lines.

- **Classic event-driven process chains (ClassicEPCs)**
  All triggering events, outcomes, steps and operators are displayed. Steps are displayed as green rectangles, triggering events and outcomes as red hexagons and operators as circles. The node number and the node type are displayed with the step name in the graphical symbols. The step type area is not displayed.

- **Event-driven process chains (EPCs)**
  All triggering events, outcomes, steps and operators are displayed. Steps are displayed as green rectangles, triggering events and outcomes as red hexagons and operators as circles. The step name is displayed in the graphical symbols.
Definition of Workflows

Purpose
You define workflows:

- If you want to model business processes that include more than one step or more than one agent, and the workflow definitions of the workflow supplied by SAP are not sufficient.
- If you want to extend or modify the workflow supplied by SAP.

The process description below assumes that you are creating a completely new workflow definition. Logically, it applies if you are processing or displaying an existing workflow definition.

If you only want to change the responsible agents [Extern] and deadlines for a SAP workflow definition, then you can extend the workflow definition [Seite 1171].

Prerequisites
Before defining a workflow, you must do the following:

- Analyze the content of the process to be modeled
- Check whether the objects involved and the methods they require are defined

The tasks you want to use in the steps of the workflow definition do not necessarily have to be defined beforehand. You can also define these tasks from the workflow definition.

Process Flow
1. Start the Workflow Builder. You can start with processing of the workflow definition and create the required container element of the workflow container in the container area. When saving for the first time you determine the abbreviation and the name.

   For further information, refer to Calling the Workflow Builder [Seite 1004].

2. You check, save and activate to conclude your work.

   For further information, refer to Workflow Builder Functions [Seite 1006].

3. If you require the triggering events [Extern], switch to processing of the relevant Multistep task [Extern]. Here you define this including the binding definition.

   For further information, refer to Definition of a Multistep Task [Seite 1194].
Versions of a Workflow Definition

Use
The system administers several versions of one workflow definition within a validity period for a multistep task. One of the versions of a workflow definition is its active version.

- The import and export parameters of the workflow container are not subject to any versioning.

To check which version of the workflow definition you are processing, select in the Workflow Builder. The version number is displayed on the tab page Control. An overview of all versions of the workflow definition can be found on the tab page Overview. The status of the workflow definition is displayed in the header.

A workflow always refers to the version of the workflow definition active at the time it is started.

Even if subsequently a new version of the workflow definition becomes the active version, workflows still running continue to refer to the version active when they were started.

Features

Calling the Workflow Builder
When the Workflow Builder is called, the active version is displayed as standard. If there is no active version, the version with the highest version number is displayed. For further information, refer to Calling the Workflow Builder [Seite 1004].

Saving and activating workflow definitions
When a workflow definition is saved, the old version is overwritten automatically. After being saved, the workflow definition has the status revised.

To set the workflow definition to status active, you select . The workflow definition is first subjected to a syntax check. Any errors and warnings discovered are displayed in the message area of the Workflow Builder. Only if errors are found, is the workflow definition not activated. If no errors are found (no red traffic lights in the message area), the workflow definition is saved and activated.

If you save and activate a workflow definition, current workflows that refer to this version of the workflow definition may no longer be executable.

Generating a runtime version of a workflow definition
You can generate a runtime version without activating the workflow definition. To do this, you choose Workflow → Generate runtime version.

New version of a workflow definition
To generate a new version, you choose Workflow → Generate version. The workflow definition is set to status new, saved.

In order not to increase the memory space requirements for workflow definitions excessively, you should only generate a new version under the following circumstances:
Versions of a Workflow Definition

- If you have made incompatible changes
- If there are productive workflows running that refer to the current version.

As long as you are still in the test and development phase, you do not usually have to create new versions.

**Transporting a workflow definition**

If a workflow definition is transported into another system, only the active version is transported. In the target system, an existing workflow definition with the same version number is overwritten if there are no current workflows for it. Otherwise, the transported workflow definition is saved with a free version number. The transported workflow definition becomes the active workflow definition in the target system.
Basic Data of a Workflow Definition

Use
You can process the basic data of the workflow definition by selecting ``` in the Workflow Builder. General information about the workflow definition is also displayed here, and you can create and process triggering events [Extern], start conditions and start transactions.

Features

Tab page Start
In the area `Workflow start by triggering events...`, you can define and remove triggering events for the workflow, as well as create and process start conditions for the triggering events. To define a triggering event, enter the relevant object type and the event into the appropriate fields in a line. You can use the F4 input help. You can activate a defined triggering event and process the binding from the event container to the workflow container.

You can process the properties of a triggering event by selecting it and choosing ```.

You can create start conditions for each triggering event defined. The workflow then only starts if the triggering event has occurred and one of the start conditions defined for it is fulfilled. Use the condition editor [Seite 1012] to define the start condition. For further information, refer to Binding Definitions from the Event Container [Seite 1227] and Start Conditions for a Workflow [Seite 1458].

In the area `...and/or start transaction(s)`, you can create a start transaction for the workflow. This start transaction is executed if you start the workflow manually. There are input fields in the start transaction, in which you can assign data to the import parameters of the interface. You can process the layout of the start transaction generated in the Screen Painter [Extern].

Tab page Control
You can make settings for the workflow definition here. If you set the indicator `Advance with dialog`, you can set the indicator individually for each step definition.

If you do not specify anything for maximum node number or maximum subworkflow level, the Customizing settings [Extern] apply. In the field Performance optimization, you specify what data is to be logged by the system. The more data you have logged, the worse the system performance is.

If you choose `Only log errors`, the workflow log does not contain any attachments, objects, forwarding information or information in the area Details.

Tab page Responsibilities
Here you define the system administrator for this workflow. If no entry is made, the entry for the party responsible in the definition of the relevant multistep task is evaluated first, then the relevant SAP Business Workflow Customizing entry [Extern] if necessary.

You can also define a standard recipient for all missed deadlines. This party is notified if the following apply:

- The workflow definition is used as a subworkflow.
- A monitored deadline with notification is activated in the step definition, but no recipient is entered.

For further information, refer to Defining Responsibilities [Extern].
Tab page Notification
Here you define the message recipient for completion of the workflow. This party is notified if the workflow is used as a subworkflow and there is no recipient specified in the task definition. Upon completion of the workflow, the text displayed here is sent as an e-mail. For further information, refer to Defining Responsibilities [Extern].

Tab page Change data
All entry, change and activation data of the workflow definition is displayed here.

Tab page Technical information
Here you will find general data about the workflow and a version overview for the workflow definition. The version number of the version being processed in the original development system is displayed in the field Original version. If a workflow is transported into another system, in which workflows are already running for this definition, the version numbers are reconciled. The indicator Locked against instantiation is maintained in the classification of the multistep task.
Using Wizards

Use
You use wizards when you want to simplify the definition of certain steps within a workflow definition. The number of wizards supplied is increased on a regular basis. Therefore, look in the system to see which wizards are currently available. The wizards are called in the Workflow Builder under Wizards. You can also call the wizard Include "Send mail" from the step type area.

You usually have to process the parts of your workflow definition created by the wizard yourself afterwards. The appropriate explanatory text in the wizard will point this out.

Integration
The wizards only ever create parts of an overall workflow definition. You can also call the Workflow Wizard Explorer from the Workflow Builder. You can use the workflow wizards available there to extend your workflow definition.
Maintenance of Activities

**Purpose**
You define or process an activity in the following cases:

- If you want to add a new activity to a workflow definition.
- If you want to change an existing activity with regard to its agent assignment or its deadline monitoring.
- If you want to analyze a workflow definition.

In the workflow definition an activity is represented with the symbol.

**Prerequisites**
One of the following prerequisites must be fulfilled when you create a new activity:

- You know the task (single-step or multistep) that you want to reference in the activity.
- You know an object type and one of its methods that has the required functionality.

During definition of an activity, you can find a defined single-step task via the method. If there are no appropriate single-step tasks, you can define one from the workflow definition.

**Process Flow**
The maintenance of an activity is spread across several tab pages. You can process the properties of the activity here, for example:

- Task selection
- Data for deadline monitoring
- Outcomes
- Responsible agents

The order of the tab pages defines the recommended sequence of processing. You should start the definition of a new activity on the tab page Control, since certain entries are made automatically on other tab pages when the task is entered.

On the tab page Control, you can make all the entries required to define an executable activity in a sequence.

**Result**
The system inserts a step of the type Activity into the workflow definition. To ensure the consistency of the workflow definition, all active outcomes are inserted with the actual step. For each outcome, the system adds a branch in the workflow definition and puts an undefined step into this branch. To ensure the consistency of the block structure, all branches created are brought together before the next step of the workflow definition.

At runtime, an activity is represented with a work item of one of the following types:

- Dialog work item if the activity references a single-step task with dialog.
- Background work item if the activity references a single-step task processed by the system without dialog.
Maintenance of Activities

- Workflow work item [Extern] if the activity represents a multistep task.
Maintenance of Tab Page Control

Use

With the specifications you make on this tab page, you define:

- Which task is executed in this activity.
- Which properties this step has.
- Who is responsible for processing the activity.

When you define a new activity, you must make entries about the task. This supplements entries on the tab page Outcomes.

Activities

Process the following settings:

Task

Define the task to be executed. An existing task can be selected via:

- Parts of names.
- The application component assigned.
- The method used.

Select if no suitable task exists yet.

If you enter another task in an existing step definition, the outcomes of the original single-step task are not assigned automatically to the outcomes of the newly entered single-step task. You are asked to assign the selected outcomes of the new single-step task to the outcomes of the single-step task originally entered. Superfluous outcomes and the corresponding branches of the Workflow definition are deleted.

The system creates new branches in the workflow definition for additional or unassigned outcomes of the new single-step task.

The properties and descriptive text of the selected task are displayed. However, you can only maintain these within the task definition.

Step properties

You can define the following step properties:

- The task is to be determined with an expression [Extern].

You can determine which task is referenced in this activity dynamically. To do this, you define a task in the activity as usual (basic task). Select and specify the container element containing the alternative task. This task must be identical to the basic task in all interfaces. If there is no result or an error is found when the expression is resolved, the basic task is used. If you use task TS70008069 in a preceding step, the task can be evaluated easily. To do this, you must define a task group containing all tasks that could be executed in this activity. You pass this task group to the method used in task TS70008069. This method provides an expression that contains the selected task and can be used in this activity.

- Processing can be rejected
If the indicator is set, processing can be rejected in the Business Workplace. The workflow definition is extended by the addition of a branch in which you must model the reaction to this rejection.

- Work items for this step appear in the [workflow log](Seite 1420).
- The following step is to be executed immediately by the same [recipient](Extern). (For further information, refer to [Advancing with Immediate Dialog](Seite 1453).)

**Binding**

You can process the *binding definition* between task container and workflow container. When you enter a task for the first time, the system proposes the following:

- New container elements for the workflow container.
- A binding definition.

You decide whether you want to accept the proposal in full or in part. To process the binding later, select ![Binding]. For further information, refer to [Rule-Based Proposal for Binding Definitions](Seite 1221).

Even if you reject the proposal at first, you can define the binding automatically at a later date. To do this, select ![binding].

**Agents**

You process the [responsible](Extern) agents and the [excluded](Extern) agents for this step if you are referencing a dialog task. These specifications only apply to this step and complement the specifications on organizational responsibility for the task. For further information, refer to [Defining Responsibilities](Extern).
Maintenance of Tab Page Outcomes

Use
The specifications you make on this tab page determine the outcomes of the step for which you want to model a reaction. If reactions to missed deadlines are modeled, the outcomes defined for the relevant missed deadline are only displayed here.

Features
The system determines the outcomes that are displayed on this tab page from the following specifications:
- Possible values of the result [Extern] of the method used in the task.
- Terminating events [Extern] of the task.
- Exceptions [Extern] for the method used in the task.
- Step definition.

You decide which outcomes you want to take into account in the workflow when modeling. The only outcomes that have to be taken into account whenever they are displayed on this tab page are the outcomes for missed deadlines Task executed synchronously and Processing rejected.

If you have not included an outcome that is required during execution into the definition, the associated workflow work item is forced into status error.

All possible outcomes are listed in the following table:

<table>
<thead>
<tr>
<th>Outcome</th>
<th>This outcome exists if ...</th>
<th>Notes and comments</th>
</tr>
</thead>
<tbody>
<tr>
<td>Terminating event of task</td>
<td>The task was defined with terminating events.</td>
<td>If the underlying method is an asynchronous method, you must select at least one event as an outcome.</td>
</tr>
<tr>
<td>Possible value of result of object method</td>
<td>The synchronous object method is defined with a result, for which fixed values or a check table are maintained.</td>
<td>If you deactivate all the values of the result, the system activates the outcome Task executed synchronously.</td>
</tr>
<tr>
<td>Object method exception</td>
<td>The object method is defined with exceptions.</td>
<td>Exceptions in the Workflow Definition [Extern]</td>
</tr>
<tr>
<td>Task executed synchronously</td>
<td>• The activity refers to a synchronous object method without result.</td>
<td>If this outcome is displayed, it is always active.</td>
</tr>
<tr>
<td></td>
<td>• The activity refers to a synchronous object method with result, but no result is selected.</td>
<td></td>
</tr>
<tr>
<td>Processing rejected</td>
<td>The indicator Processing can be rejected is set.</td>
<td>If processing of the relevant work item is rejected at runtime (via Reject execution in the Business Workplace), the steps defined after this outcome are executed.</td>
</tr>
<tr>
<td>Processing obsolete</td>
<td>The work item can be set to obsolete via a step of the type</td>
<td>The steps defined after this outcome are executed.</td>
</tr>
</tbody>
</table>
### Maintenance of Tab Page Outcomes

<table>
<thead>
<tr>
<th>Process Control</th>
<th>This outcome should only be activated if the modeled reaction to a missed deadline contains a step that sets the work item for this step to obsolete.</th>
</tr>
</thead>
<tbody>
<tr>
<td>Requested end</td>
<td>The relevant deadline monitoring is activated and a modeled reaction required.</td>
</tr>
<tr>
<td>Latest end</td>
<td>Within these branches, you can model a process control step that sets the work item of the step to obsolete. You cannot deactivate these outcomes.</td>
</tr>
<tr>
<td>Latest start</td>
<td></td>
</tr>
</tbody>
</table>

At runtime, the system enters an event receiver linkage only for those terminating events that you select in the definition of an activity and therefore include in the workflow definition.

No other events terminate the activity, even if they are entered as terminating events for the single-step task.

### Activities

You activate the outcomes that you want to actually transfer into the workflow definition. Select 🗡️ in the relevant line. Active outcomes are marked with 📜️ and have a modeling branch in the workflow definition.

⚠️ If you deactivate an active outcome, all steps already modeled in this branch are deleted.

For further information about an outcome, select it and choose Detail in the context menu.
Maintenance of Tab Page Notification

Use
The specifications you make on this tab page define the notification agent for completion. This entry is optional.
These notification agents receive a notification of completion [Extern] when the relevant work item is set to status completed.

Activities
Enter a message recipient for completion. For further information, refer to Defining Responsibilities [Extern].
If no recipient is entered, no notification of completion is sent. The notification of completion contains the completion text that can only be processed [Seite 1201] in the task definition, and a reference to the work item display of the completed work item.
Maintenance of Deadline Tab Pages

Use
You can define all the deadlines for this step with the specifications you make on these tab pages. All these entries are optional.

Features
You can set the following deadlines on the corresponding tab pages:

- Requested start [Extern]
- Latest start [Extern]
- Requested end [Extern]
- Latest end [Extern]

You can define reactions to a missed deadline for the last three deadline types. The system offers the following possible reactions:

- Notification
- Free modeling

Activated deadlines are marked with in the tab page index.

You define deadlines with respect to a reference date/time. The system offers the following reference date/times:

- The creation date/time of the work item.
- The creation date/time of the workflow to which the monitored work item belongs.
- A date in the form of an expression, which is derived from the context of the application during execution of the workflow.

Activities
You activate monitoring of the relevant deadline by selecting a reference date/time for the deadline. If you choose expression, you have to define the reference date/time by specifying expressions [Extern] for the date and time yourself. Use the F4 input help for entering expressions.

The value referenced using the expression must be of data type D for the date and data type T for the time. If you specify a date but no time, the system sets the time to 00:00:01 (requested and latest start) or 23:59:59 (requested and latest end).

Process the deadline by entering a duration and an appropriate unit. Negative durations only apply if you define the reference date/time yourself via an expression. The reaction to a deadline being reached depends on the type of deadline:

Requested start
If the work item is created before the requested start, it is assigned the status waiting. When the requested start is reached, the system sets the work item to status ready. If the work item is created after the requested start, it is assigned the status ready immediately.

Latest end, latest start, requested end
You can either notify someone or model a reaction in the workflow definition for when the deadline is reached.
- **Display text (standard escalation)**
  
  If you want to notify someone, you must specify a recipient [Extern] on the tab page Display text. The message text is displayed in the display area. It can only be changed in the task definition.

- **Modeled**

  If you want to model your own reaction, enter a name for the new outcome on the tab page Modeled. A reaction branch is added to the workflow definition, which is processed if the deadline is missed.

  If a deadline is missed, the original step is not yet completed. The steps following this "missed deadline event" do not end the original activity.

  For further information, refer to Modeled Deadline Monitoring or Standard Escalation [Extern].
Maintenance of Tab Page Methods

Use
In addition to the task's method, you can define the following methods:

- **Before method [Extern]**: Executed before execution of the method.
- **Secondary method [Extern]**: Executed at the same time as the method.
- **After method [Extern]**: Executed after execution of the method.

Prerequisites
The methods and their object types are defined in the *Business Object Repository*. The objects on which the methods are executed must be referenced in a container element in the workflow container.

⚠️ If you define one of these methods for a user decision, you must delete the entry in the field *Function module* on the tab page *Work item display*.

Features

**Calling before and after methods**
When executing a work item with before or after methods, the system first calls the before methods synchronously in any order and processes them. Then the system starts the task method and secondary methods in different sessions. Secondary methods are only for display. Any changes carried out in them have no (direct) effect on the workflow. Finally, the system calls the after methods synchronously in any order and processes them.

The after methods being called is not dependent on the result of the task method. They are called even if the task method terminated with an error.

The after methods being called is not dependent on whether or not the task method is terminated with an event. They are called immediately after the synchronous part of the method.

With work items that are set to status *completed* by a terminating event without method execution, the before, secondary and after methods are not called. Before methods cannot create the object on which the task method is to run. However, after methods can run on a object created by the task method.

**Data transfer into and out of the methods**
There is no separate binding between task container and method container for these methods. The methods are given the method container of the task method. If no binding is defined between the task and the method, the system passes the task container. Secondary methods cannot return data to the workflow nor influence the process control. Before and after methods can evaluate and manipulate the method container in order to influence the subsequent method or steps.
Error handling
As with secondary methods, errors in before and after methods have no influence on the execution of a work item. The processing logic of the work item depends solely on the task method. The system records errors in the before and after methods as warnings in the workflow log.

Activities
To define an object method as a before, an after or a secondary method, select (in the column Object) a container element in the workflow container, which contains the object to be processed. Then select the object method required in the column Method.
Maintenance of Tab Page Work Item Display

Use
On the tab page *Work item display*, you can individualize the work item display with an additional tab page, and the work item preview with graphics or text.

Features

**Step-specific tab page for work item display**
An individual tab page is added to the work item display. This tab page is displayed in addition to the three tab pages *Basic data, Activities* and *Available objects* as the first tab page of the work item display. This additional tab page only appears when work items that represent this step are displayed. This tab page can be used, for example, to:

- Display important information from the current process, which does not appear in the work item display as standard.
- Provide functions that the end user has to execute often.
- Create a customer-specific 'look and feel'.

**Adapting the work item preview in the Business Workplace**
A user exit is available for individualizing the work item preview. For further information, refer to the User Exit for Work Item Preview [Extern].

A user decision can be executed in the work item preview as standard. The function module required for this is entered automatically.

If secondary methods are to be executed or methods before or after the work item execution in a user decision, delete the entry in the field *Function module*.

Activities

**Step-specific tab page for work item display**
Enter the program and the number of the screen into the relevant input fields. For further information, refer to Programming a Step-Specific Tab Page [Extern].

**Adapting the work item preview in the Business Workplace**
Enter the function module that you want to use in the input field *Function module*. You can create a function module using a template by selecting 📋.
Maintenance of Tab Page Other

Use
On the tab page Other, you can change the priority and repeated execution of the work item.

Features

Step priority
Here you define what priority the work items created for this step have. Dialog work items of priority 1 are displayed in the Business Workplace as express work items.

Table-Driven Dynamic Parallel Processing
You can process this step several times in parallel. The multiline container element entered here controls this parallel processing. For further information, refer to Table-Driven Dynamic Parallel Processing [Extern].

Repeat counter for work items with temporary errors
Maximum number of attempts made by the system to restart a background work item [Extern] with temporary errors. If all of the attempts are unsuccessful, the work item status is set to:

- Error (if the temporary exception is not modeled)
- Completed (if the temporary exception is modeled)

This function is only available for steps that refer to a single-step task without dialog ("background task"). If the repeat counter has the value 0, the maximum number of retries is determined from the settings in Customizing [Extern].

For further information, refer to Error Handling for Background Work Items [Seite 1388].
Maintenance of Documents from a Template

Purpose
You use this step type to create workflow documents of various PC applications and process them within the workflow. The work item recipient creates the document based on the document template which can also contain container elements of the workflow container.

In the workflow definition such a step type is represented with the symbol 📝.

Prerequisites
The PC application for which you want to create a document must be installed on your PC.

Process Flow
The maintenance is spread across several tab pages. You can process the properties here, for example:

- Template selection
- Responsible agents [Extern]
- Deadline monitoring [Extern]

The order of the tab pages defines the recommended sequence of processing.

When a document from template is created, the system enters the SAP document task *Create document from template* (TS70008298) into the field *Task* on the tab page *Control* as standard. You can change this entry at any time.

You should only create your own document task if:

- You want to restrict the possible agents of the task.
- You want to display a different text in the work item preview of the Business Workplace.

The generated document is passed to a container element of the workflow container, from which it is available to other steps.

Result
The system inserts a step of the type *document from template* into the workflow definition. To ensure the consistency of the workflow definition, all active outcomes are inserted with the actual step. For each outcome, the system adds a branch in the workflow definition and puts an undefined step into this branch.

To ensure the consistency of the block structure, all branches created are brought together before the next step of the workflow definition.

At runtime, this step is represented by a dialog work item [Extern] in the Business Workplaces [Seite 1368] of its recipients [Extern].

When a recipient chooses the work item created for this step for processing in the Business Workplace, the relevant PC application is opened with the chosen template. If several or no templates are defined in the step definition, the recipient can choose the desired template in a dialog box.
Maintenance of Tab Page Document Templates

Use
With the specifications you make on this tab page, you define:

- Which document templates are offered to the user.
- The container element of the workflow container, in which the document is stored.
- Who the responsible agent [Extern] of the step is.

Activities
Make the following settings:

Document template
You must enter the document templates that should relate to the document to be created. If you enter more than one template, the user can choose which template is to be used when the step is executed. If you do not enter any templates, the user is offered all possible templates for selection. If the template required does not exist, select [ ]. For further information, refer to Creating Document Templates [Seite 1047].

Element for document
You must choose a container element in the workflow container for the new document. Use the F4 input help. If you enter a container element that does not yet exist, it is created automatically.

Agent determination
You process the responsible [Extern] agents and the excluded [Extern] agents for the document from template. These specifications only apply to this step and complement the specifications on organizational responsibility. For further information, refer to Defining Responsibilities [Extern].

If you only make entries on this tab page, a step of the type Document from template on the basis of the standard task supplied by SAP is transferred into the workflow definition.
Creating Document Templates

Use
Document templates determine the appearance of PC documents created in the workflow. The document created is based on one of the templates available, and can contain container elements of the workflow container or system fields as variables.

Prerequisites
The PC application on which the template is to be based must be installed locally on your PC.
In order to display the Document template folder in the object area of the Workflow Builder, the workflow definition must have been saved at least once.

Procedure
1. Open the PC application by double-clicking on the Document templates folder in the object area or select in the step definition of document generation.

   The PC application which you last used to create a template is opened in the Workflow Builder. The system fields and the container elements of the workflow container are offered for selection in the object area. They can be transferred into the template by double-clicking. All relationships created between container elements/system fields and the document template are displayed in the task area.

   It is not possible to insert system fields and container elements into all types of document template.

2. If you want to create a document template of a different class, select Change document class and choose the new type of document template.

3. Select to assign a name for the document template.

4. Create the document template in the usual way in your PC application. You can use all the functions of the PC application.

5. Insert container elements and/or system fields from the object area into your template by double-clicking. These fields are replaced with the content of the container elements/system fields at runtime.

6. Select to save your document template.

   You can only save the document template from your PC application as a local copy on your PC.
Maintenance of Tab Page Control

Use
The specifications you make on this tab page enable you to create a document from a template.

Activities
Make the following settings:

Task
A document task must be entered in this field. You should only use another document task if you want to restrict the selection of possible agents [Extern] or display another text in the work item preview. Select to create a new document task on the basis of the standard task. The properties and task description of the selected task are displayed.

Step properties
You can define the following step properties:

- The task is to be determined with expression [Extern].
  You can determine which document task is used dynamically. To do this, select and specify the container element that references the alternative task. This task must be a decision task and should have been developed from a copy of the standard document task.

- Processing can be rejected
  If the indicator is set, processing can be rejected in the Business Workplace. The workflow definition is extended by the addition of a branch in which you must model the reaction to this rejection.

- Work items for this step appear in the workflow log [Seite 1420].

- The following step is to be executed immediately by the same recipient [Extern] (for further information, refer to Advancing with Immediate Dialog [Seite 1453]).

Binding
Process the binding between task container and workflow container. This is necessary if the agent is to create a document that is based on a template as an attachment, and if text variables are used in the task description. To do this, select Binding (existing). For further information, refer to Binding Definition for Document from Template [Seite 1230].
Maintenance of Tab Page Outcomes

Use
The specifications you make on this tab page determine the outcomes of the step for which you want to model a reaction. If reactions to missed deadlines are modeled, the outcomes defined for the relevant missed deadline are only displayed here.

Features
On this tab page you can only activate the outcomes *Processing obsolete* and *Document could not be created*. All other outcomes are determined by the system and activated automatically.

<table>
<thead>
<tr>
<th>Outcome</th>
<th>This outcome exists if ...</th>
<th>Notes and comments</th>
</tr>
</thead>
<tbody>
<tr>
<td>Document could not be created</td>
<td>Always</td>
<td>Defined exception for the method of the standard document task.</td>
</tr>
<tr>
<td>Task executed synchronously</td>
<td>Always</td>
<td>You cannot deactivate this outcome.</td>
</tr>
<tr>
<td>Processing rejected</td>
<td>The indicator <em>Processing can be rejected</em> is set.</td>
<td>If processing of the relevant work item is rejected at runtime (via <em>Reject execution</em> in the Business Workplace), the subsequent steps defined after this exception are executed.</td>
</tr>
<tr>
<td>Processing obsolete</td>
<td>The work item can be set to obsolete via a step of the type <em>process control</em>.</td>
<td>The subsequent steps defined after this exception are executed. This outcome should only be activated if the modeled reaction to a missed deadline contains a step that sets the work item for this step to obsolete.</td>
</tr>
<tr>
<td>Requested end</td>
<td>The relevant deadline monitoring is activated and a modeled reaction required.</td>
<td>Within these branches, you can model a <em>process control</em> step that sets the work item of the step to obsolete. You cannot deactivate this outcome.</td>
</tr>
<tr>
<td>Latest end</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Latest start</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Activities
You activate the outcomes that you want to actually transfer into the workflow definition. Select 📆 in the relevant line.
Active outcomes are marked with 📆 and have a modeling branch in the workflow definition.

⚠️

If you deactivate an active outcome, the whole of the associated branch is deleted.

For further information about an outcome, select it and choose *Detail* in the context menu.
Maintenance of Tab Page Notification

Use
The specifications you make on this tab page define the *notification agent for completion*. This entry is optional. These notification agents receive a notification of completion when the relevant work item is set to status *completed*.

Activities
Enter a message recipient for completion. For further information, refer to Defining Responsibilities.

If no recipient is entered, no notification of completion is sent. The notification of completion contains the *completion text* that can only be processed in the task definition, and a reference to the work item display of the completed work item.
Maintenance of Deadline Tab Pages

Use
You can define all the deadlines for this step with the specifications you make on these tab pages. All these entries are optional.

Features
You can set the following deadlines on the corresponding tab pages:

- Requested start [Extern]
- Latest start [Extern]
- Requested end [Extern]
- Latest end [Extern]

You can define reactions to a missed deadline for the last three deadline types. The system offers the following possible reactions:

- Notification
- Free modeling

Activated deadlines are marked with 🔄 in the tab page index.

You define deadlines with respect to a reference date/time. The system offers the following reference date/times:

- The creation date/time of the work item.
- The creation date/time of the workflow to which the monitored work item belongs.
- A date in the form of an expression, which is derived from the context of the application during execution of the workflow.

Activities
You activate monitoring of the relevant deadline by selecting a reference date/time for the deadline. If you choose expression, you have to define the reference date/time by specifying expressions [Extern] for the date and time yourself. Use the F4 input help for entering expressions.

➤

The value referenced using the expression must be of data type D for the date and data type T for the time. If you specify a date but no time, the system sets the time to 00:00:01 (requested and latest start) or 23:59:59 (requested and latest end).

Process the deadline by entering a duration and an appropriate unit. Negative durations only apply if you define the reference date/time yourself via an expression.

The reaction to a deadline being reached depends on the type of deadline:

**Requested start**
If the work item is created before the requested start, it is assigned the status waiting. When the requested start is reached, the system sets the work item to status ready. If the work item is created after the requested start, it is assigned the status ready immediately.

**Latest end, latest start, requested end**
You can either notify someone or model a reaction in the workflow definition for when the deadline is reached.


- **Display text (standard escalation)**
  
  If you want to notify someone, you must specify a recipient [Extern] on the tab page *Display text*. The message text is displayed in the display area. It can only be changed in the task definition.

- **Modeled**
  
  If you want to model your own reaction, enter a name for the new outcome on the tab page *Modeled*. A reaction branch is added to the workflow definition, which is processed if the deadline is missed.

  If a deadline is missed, the original step is not yet completed. The steps following this "missed deadline event" do not end the original activity.

  For further information, refer to Modeled Deadline Monitoring or Standard Escalation [Extern].
Maintenance of Tab Page Methods

Use
In addition to the task's method, you can define the following methods:

- **Before method [Extern]**: Executed before execution of the method.
- **Secondary method [Extern]**: Executed at the same time as the method.
- **After method [Extern]**: Executed after execution of the method.

Prerequisites
The methods and their object types are defined in the *Business Object Repository*. The objects on which the methods are executed must be referenced in a container element in the workflow container.

> If you define one of these methods for a user decision, you must delete the entry in the field *Function module* on the tab page *Work item display*.

Features

Calling before and after methods
When executing a work item with before or after methods, the system first calls the before methods synchronously in any order and processes them. Then the system starts the task method and secondary methods in different sessions. Secondary methods are only for display. Any changes carried out in them have no (direct) effect on the workflow. Finally, the system calls the after methods synchronously in any order and processes them.

> The after methods being called is not dependent on the result of the task method. They are called even if the task method terminated with an error.

> The after methods being called is not dependent on whether or not the task method is terminated with an event. They are called immediately after the synchronous part of the method.

With work items that are set to status *completed* by a terminating event without method execution, the before, secondary and after methods are not called. Before methods cannot create the object on which the task method is to run. However, after methods can run on an object created by the task method.

Data transfer into and out of the methods
There is no separate binding between task container and method container for these methods. The methods are given the method container of the task method. If no binding is defined between the task and the method, the system passes the task container. Secondary methods cannot return data to the workflow nor influence the process control. Before and after methods can evaluate and manipulate the method container in order to influence the subsequent method or steps.
Error handling
As with secondary methods, errors in before and after methods have no influence on the execution of a work item. The processing logic of the work item depends solely on the task method. The system records errors in the before and after methods as warnings in the workflow log.

Activities
To define an object method as a before, an after or a secondary method, select (in the column Object) a container element in the workflow container, which contains the object to be processed. Then select the object method required in the column Method.
Maintenance of Tab Page Work Item Display

Use
On the tab page *Work item display*, you can individualize the work item display with an additional tab page, and the work item preview with graphics or text.

Features

**Step-specific tab page for work item display**
An individual tab page is added to the work item display. This tab page is displayed in addition to the three tab pages *Basic data*, *Activities* and *Available objects* as the first tab page of the work item display. This additional tab page only appears when work items that represent this step are displayed. This tab page can be used, for example, to:
- Display important information from the current process, which does not appear in the work item display as standard.
- Provide functions that the end user has to execute often.
- Create a customer-specific 'look and feel'.

**Adapting the work item preview in the Business Workplace**
A user exit is available for individualizing the work item preview. For further information, refer to the [User Exit for Work Item Preview](#).

A user decision can be executed in the work item preview as standard. The function module required for this is entered automatically.

If secondary methods are to be executed or methods before or after the work item execution in a user decision, delete the entry in the field *Function module*.

Activities

**Step-specific tab page for work item display**
Enter the program and the number of the screen into the relevant input fields. For further information, refer to [Programming a Step-Specific Tab Page](#).

**Adapting the work item preview in the Business Workplace**
Enter the function module that you want to use in the input field *Function module*. You can create a function module using a template by selecting the template icon.
Maintenance of Tab Page Other

Use
On the tab page *Other*, you can change work item priority.

Features
Step priority
Here you define what priority the work items created for this step have. Dialog work items [Extern] of priority 1 are displayed in the Business Workplace as express work items.
Maintenance of Conditions

Use
You use a condition [Extern] in a workflow definition if:

- At execution time, only one of two possible alternatives can be processed during the workflow, from a business point of view.
- The workflow system can make a decision based on the contents of the workflow container without user interaction.

In the workflow definition a condition is represented with the symbol 🔄.

Features
A condition is a branching in your workflow definition. The workflow system evaluates the condition comparing elements of the workflow container with constants, system fields or other fields. You define conditions in the condition editor, in which you can also define complex conditions. The evaluation of a condition can result in the values true or false.

Activities
You maintain conditions on the screen Create step: Condition. Here you have to assign a step name, define the condition [Seite 1012] and determine names for the outcomes true and false. These names are used to label the two outcomes of the condition in the workflow definition.

The currently valid condition is displayed on this screen. To change this or create a new condition, you must go to the condition editor by double-clicking.

Result
The system inserts a step of the type condition into the workflow definition. For both outcomes, the system adds a branch in the workflow definition and puts an undefined step into this branch. To ensure the consistency of the block structure, all branches created are brought together before the next step of the workflow definition.
Maintenance of Multiple Conditions

Use
You use a multiple condition in a workflow definition if:

- At execution time, only one of several possible alternatives can be processed during the workflow, from a business point of view.
- The workflow system can make a decision based on the contents of the workflow container.
- The decision can be formulated based on a comparison between an expression from workflow container and a finite amount of comparison values (constants, expressions).

You also have the option of going to an Others branch if none of the conditions agree with the value of the basis of comparison.

In the workflow definition a multiple condition is represented with the symbol 🔄.

Features
For the decision, the workflow system checks the value of a basis of comparison against the defined comparison values.

The basis of comparison can be a container element of the workflow container or a system field, and the comparison value can be a constant, a container element of the workflow container or a system field.

You cannot use multiline container elements.

Activities
You define the basis of comparison as an expression with reference to the workflow container. You enter the comparison values into the table as expressions and assign a name with which the outcome is displayed in the workflow definition. For comparison values that are not taken into account, you can include a branch into the workflow definition for Other values. To do this, you assign a name in the relevant input field.

If the basis of comparison has a value that does not agree with any comparison value and if there is no branch Other values, the workflow assumes the status error.

For each outcome, the system adds a branch in the workflow definition and puts an undefined step into this branch. To ensure the consistency of the block structure, all branches created are brought together before the next step of the workflow definition.
Maintenance of Event Creator

Use

Use an event creator if you want to publish an event from a workflow. This event can be used as follows:

- To start other workflows or tasks.
  
  For this, the event must be entered as a triggering event [Extern] for the relevant workflow or task.

- As a terminating event [Extern] of a task.

- For internal communication and synchronization.
  
  To react to an event in a workflow, you use a wait step [Extern].

In the workflow definition an event creator is represented with the symbol 📣.

Features

At runtime the system creates an event in accordance with your specifications. The system caters automatically for the workflow system variables in the event container [Extern] as well.

If the event container contains other, non-standard elements, you must define a binding from the workflow container to the event container [Seite 1223].

Activities

You give a name to the step and to the outcome. The other fields are as follows:

- **Container element**
  
  This container element of the workflow container must contain an object reference. The referenced object is the object to which the event refers.
  
  If the event is created at runtime, the system writes this object reference into the container element _Evt_Object in the event container.

- **Object type**
  
  This system automatically puts the type of the object referenced in the container element into this field.

- **Event**
  
  This event is created by the system at execution time of the workflow. The event must be defined for the object type in the Business Object Repository.

Select 🌐 to see whether all the receivers you expected were started. For further information, refer to Event Simulation [Seite 1363].
Maintenance of Wait Steps

Purpose
You use a wait step [Extern] in a workflow definition:

- To suspend the entire execution of the workflow until a defined event [Extern] has occurred. A workflow is waiting for requested documents to be provided for a particular applicant activity.
  Once the requested documents are received in the company, the transaction used for inbound processing and archiving creates an event which the wait step has been waiting for as a receiver.

- To wait for an event in parallel processing branches, which renders processing in the other branches superfluous.
  The event waited for should not be created in the other branches of the fork.

In the workflow definition a wait step is represented with the symbol [Wait Step].

Process Flow
The maintenance of a wait step is spread across several tab pages. You can process the properties of the wait step here, for example:

- Event to be waited for.
- Number of events to be waited for.
- Missed deadlines
- Outcomes

The order of the tab pages defines the recommended sequence of processing.

On the tab page Control, you can make all the entries required to define an executable wait step.

Result
The system inserts a step of the type wait step into the workflow definition. To ensure the consistency of the workflow definition, all active outcomes are inserted with the actual step. For each outcome, the system adds a branch in the workflow definition and puts an undefined step into this branch. To ensure the consistency of the block structure, all branches created are brought together before the next step of the workflow definition.

At runtime, a wait step is represented by a wait step work item [Extern].
Wait step work items are not displayed in the Business Workplace but can be found using the work item selection [Seite 1490].
Maintenance of Tab Page Control

Use
With the specifications you make on this tab page, you define the event you want the workflow to wait for.

Activities
You must describe the expected event uniquely. Specify the following:

Container element
In this step, the workflow waits for an event created by a specific object. As this is not yet known at definition time, you specify a container element of the workflow container, which will contain an object reference at runtime. With this information, the workflow system enters the instance linkage [Extern].

The entries in the tables for event receiver linkage (linkage tables [Extern]) are made exclusively by the workflow system itself.

Object type
Here the workflow system automatically enters the type of the object referenced in the container element.

Event
Choose the event to be waited for in this step. The event must be defined for the object type in the Business Object Repository.

Number of events
You enter how many times the same event must occur for execution of the workflow to be continued.

Binding
Process the binding definition from the event container [Seite 1227] to the workflow container if you want to use information supplied with the event in the workflow. To do this, select Binding.

In general, this binding definition is not required.

Name
Enter the ID of the outcome of the wait step, with which processing is to be continued after the event occurs.
Maintenance of Tab Page Outcomes

Use
The specifications you make on this tab page determine the outcomes of the step for which you want to model a reaction. The outcome entered on the tab page Control is only displayed here. If reactions to missed deadlines are modeled, the outcomes defined for the relevant missed deadline are only displayed here.

Features
On this tab page you can only activate the outcome Processing obsolete. All other outcomes are determined by the system and activated automatically.

All possible outcomes are listed in the following table:

<table>
<thead>
<tr>
<th>Outcome</th>
<th>This outcome exists if ...</th>
<th>Notes and comments</th>
</tr>
</thead>
<tbody>
<tr>
<td>Wait step completed</td>
<td>Always</td>
<td>You cannot deactivate this outcome.</td>
</tr>
<tr>
<td>Processing obsolete</td>
<td>The work item can be set to obsolete via a step of the type process control.</td>
<td>The subsequent steps defined after this exception are executed. This outcome should only be activated if the modeled reaction to a missed deadline contains a step that sets the work item for this step to obsolete.</td>
</tr>
<tr>
<td>Requested end</td>
<td>The relevant deadline monitoring is activated and a modeled reaction required.</td>
<td>Within these branches, you can model a process control step that sets the work item of the step to obsolete. You cannot deactivate these outcomes.</td>
</tr>
<tr>
<td>Latest end</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Latest start</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Activities
You activate the outcomes that you want to actually transfer into the workflow definition. Select 📒 in the relevant line.
Active outcomes are marked with 🆗 and have a modeling branch in the workflow definition.

⚠️
If you deactivate an active outcome, the whole of the associated branch is deleted.

For further information about an outcome, select it and choose Detail in the context menu.
Maintenance of Tab Page Notification

Use
The specifications you make on this tab page define the notification agent for completion. This entry is optional.
These notification agents receive a notification of completion when the relevant work item is set to status completed.

Activities
Enter a message recipient for completion. For further information, refer to Defining Responsibilities.
If no recipient is entered, no notification of completion is sent. The notification of completion contains the completion text that can only be processed in the task definition, and a reference to the work item display of the completed work item.
Maintenance of Deadline Tab Pages

Use
You can define all the deadlines for this step with the specifications you make on these tab pages. All these entries are optional.

Features
You can set the following deadlines on the corresponding tab pages:

- **Requested start** [Extern]
- **Latest start** [Extern]
- **Requested end** [Extern]
- **Latest end** [Extern]

You can define reactions to a missed deadline for the last three deadline types. The system offers the following possible reactions:

- Notification
- Free modeling

Activated deadlines are marked with 📅 in the tab page index.

You define deadlines with respect to a reference date/time. The system offers the following reference date/times:

- The creation date/time of the work item.
- The creation date/time of the workflow to which the monitored work item belongs.
- A date in the form of an expression, which is derived from the context of the application during execution of the workflow.

Activities
You activate monitoring of the relevant deadline by selecting a reference date/time for the deadline.

If you choose **expression**, you have to define the reference date/time by specifying expressions [Extern] for the date and time yourself. Use the F4 input help for entering expressions.

The value referenced using the expression must be of data type D for the date and data type T for the time. If you specify a date but no time, the system sets the time to 00:00:01 (requested and latest start) or 23:59:59 (requested and latest end).

Process the deadline by entering a duration and an appropriate unit. Negative durations only apply if you define the reference date/time yourself via an expression.

The reaction to a deadline being reached depends on the type of deadline:

**Requested start**
If the work item is created **before** the requested start, it is assigned the status waiting. When the requested start is reached, the system sets the work item to status ready. If the work item is created **after** the requested start, it is assigned the status ready immediately.

**Latest end, latest start, requested end**
You can either notify someone or model a reaction in the workflow definition for when the deadline is reached.
• Display text (standard escalation)
  
  If you want to notify someone, you must specify a recipient [Extern] on the tab page Display text. The message text is displayed in the display area. It can only be changed in the task definition.

• Modeled
  
  If you want to model your own reaction, enter a name for the new outcome on the tab page Modeled. A reaction branch is added to the workflow definition, which is processed if the deadline is missed.

  If a deadline is missed, the original step is not yet completed. The steps following this "missed deadline event" do not end the original activity.

  For further information, refer to Modeled Deadline Monitoring or Standard Escalation [Extern].
Maintenance of Process Control

Use
You use this step type to complete or cancel other work items of the current workflow or the workflow itself at runtime.
In the workflow definition a process control step is represented with the symbol 🏃.

Features
You determine the type of process control in the field Function. This step always has an outcome.

Process control influences other work items

Cancel work item
The workflow system forces another work item of the same workflow into the status logically deleted.
This completes this other work item. Defined subsequent steps of this work item are not executed.
To indicate the work item to be canceled, you specify the node number of a step from the current workflow definition in the field Workflow step.

This function can only be used if the process control step and the step to be canceled are located in a fork. Note that the other work item is only forced into the status logically deleted, if it has the status in process when the process control step is executed.

Set work item to obsolete
The workflow system forces another work item of the same workflow into the status completed and continues the processing of the workflow in the branch processing obsolete. This other work item must therefore support the exception processing obsolete.
To indicate the work item for which processing is to be set to obsolete, you specify the node number of a step from the current workflow definition in the field Workflow step.
You can use this function to define modeled deadline monitoring. The work item whose deadline has been missed is forced into status completed. For further information on this type of modeling, refer to Modeled Deadline Monitoring or Standard Escalation [Extern].

You should not use this function in forks to force work items in other branches of the fork into status completed.

Process control influences the workflow itself

Terminate workflow
The workflow system terminates the current workflow. If there are any incomplete work items, for example in parallel branches of this workflow, they are forced into status logically deleted.
The workflow is in its defined end status. If the terminated workflow was a subworkflow of a superordinate workflow, the system executes the binding between the container of the subworkflow and the container of the superordinate workflow in accordance with the definition, and continues the superordinate workflow.
Cancel workflow
The workflow system cancels further execution of the current workflow. The workflow work item is forced into status *logically deleted*. If there are any incomplete work items, for example in parallel branches of this workflow, they are also forced into status *logically deleted*.
If the canceled workflow was a subworkflow of a superordinate workflow, the superordinate workflow is **not** continued.

Activities
You select the function to be performed in this step and enter the node number of the workflow step to be cancelled if applicable. Assign a name for the step outcome.
Maintenance of Container Operations

Use
You use a container operation to change a container element of the workflow container at runtime.
In the workflow definition a container operation is represented with the symbol 🎨.

Features
A container operation has one outcome. The following operation types are available:

Arithmetic operation
In an arithmetic operation, the result of a simple calculation is assigned to a container element of the workflow container. The arithmetic operation requires two expressions.

Assign/Append
In assigning or appending, the content of a container element or expression is assigned to a container element of the workflow container. The assignment only requires an expression. The second expression and the operator are not required.

Assign
The expression is assigned to the container element. This content of the container element is deleted first. The assignment is possible both for single-line and for multiline container elements.
When assigning to multiline container elements, the following apply:
- `<MultilineElement> =`  
  The multiline element is initialized.
- `<MultilineElement> = <MultilineExpression>`  
  The content of `<MultilineElement>` is deleted.  
  The content of `<MultilineExpression>` is assigned to `<MultilineElement>`.
- `<MultilineElement> = <Single-LineExpression>`  
  The content of `<MultilineElement>` is deleted.  
  The content of `<Single-LineExpression>` is entered into the first line of the multiline container element.

Append
The expression is appended to the container element. This extends the content of the multiline container element. Appending is only possible for multiline container elements.
When assigning to multiline container elements, the following apply:
- `<MultilineElement> <- <MultilineExpression>`  
  The content of `<MultilineExpression>` is appended to the content of `<MultilineElement>`.
- `<MultilineElement> <- <Single-LineExpression>`  
  The content of `<Single-LineExpression>` is appended to the multiline container element as the last line.
Maintenance of User Decisions

Purpose
In a user decision [Extern], the task description is displayed to the current agent together with the possible decision options during execution. You can use a user decision in the following situations:

- **Only one of several** possible alternatives can be processed in the workflow from a business point of view at execution time..
- A decision about the progression of the workflow must be made in dialog with a user.
- **An instruction** (= user decision with only one decision option) is required to continue the workflow.
- **For approval and release steps**, possibly in connection with a secondary method (the latter is possible without the application having to cater for these functions).

Prerequisites
The user decision refers to a decision task [Extern]. In order for an individual text to be displayed in the dialog box for the user decision, you must create your own decision task. You specify the individual text as task description in the task definition. For further information, refer to Creating Decision Tasks [Seite 1078].

If you do not think an individual text is necessary, you do not generally need your own decision task. In this case, you use the task **generic decision task** (TS00008267).

Process Flow
The maintenance of a user decision is spread across several tab pages. You can process its properties here, for example:

- Decision options
- Responsible agents [Extern]
- Deadline monitoring [Extern]

The order of the tab pages defines the recommended sequence of processing.

On the tab page Decision, you can make all the entries required to define an executable user decision.

When a user decision is created, the system enters the SAP decision task **generic decision task** (TS00008267) into the relevant field on the tab page Control as standard. You can change this entry at any time.

You can change this setting for the decision task in the Workflow Builder Customizing [Extern].

If you use a decision task you defined in a user decision and have text variables for substitution, you must provide the relevant elements in the task container with values via a binding from the workflow container. For further information, refer to Binding Definition in Decision Tasks [Seite 1231].
Secondary methods with user decision

If you use the user decision as a step for approval or release of an object, you can display the object in question as a secondary method, hence supplementing the functions of the user decision. For further information, refer to Maintenance of Tab Page Methods [Seite 1083].

Result of user decision

If you want to store the result of a user decision in an element of the workflow container in order to use it again in a later step (for example, loop or condition), you must define a binding from the task container to the workflow container. For further information, refer to Binding Definition in Decision Tasks [Seite 1231].

Result

The system inserts a step of the type user decision into the workflow definition. To ensure the consistency of the workflow definition, all active outcomes are inserted with the actual step, whereby every decision alternative is represented by an outcome. For each outcome, the system adds a branch in the workflow definition and puts an undefined step into this branch. To ensure the consistency of the block structure, all branches created are brought together before the next step of the workflow definition.

At runtime, this user decision is represented by a dialog work item [Extern] in the Business Workplaces of its recipients [Extern].

When a recipient chooses the work item created for this step for processing in the Business Workplace, a screen is displayed with the description text and the defined decision texts and one of the alternatives offered can be selected.

The recipient can also display or create attachments and objects on this screen.
Maintenance of Tab Page Decision

Use
With the specifications you make on this tab page, you define:

- The decision text used when the user decision appears to the recipient at runtime.
- The decision options offered to the recipient for selection.
- Who the responsible agent of the decision is.

Activities
You make the following specifications to define a decision:

Decision title
You can include up to four system fields or expressions from the workflow container in the title as parameters. To do this, you mark the points at which the parameters are to be inserted with placeholders & and enter the expressions in the fields Parameter1 to Parameter4.
At runtime, the system replaces the placeholders with the current values.

Decision options
For each option, specify a name and a decision text that is displayed to the recipient. All options defined here are outcomes of the step and are transferred with their names to the tab page Outcomes.

Agent determination
Define the responsible agents and the excluded agents. These only apply to this workflow definition. For further information, refer to Defining Responsibilities.

If you only make entries on this tab page, a user decision on the basis of the decision task supplied by SAP is transferred into the workflow definition.
Maintenance of Tab Page Control

Use
With the specifications you make on this tab page, you define:

- Which decision task is used.
- How the user decision is presented.

Activities
Make the following settings:

Task
A decision task [Extern] must be entered in this field. You should only use another decision task if you want to restrict the selection of possible agents [Extern] or use another decision text. Select to create a new decision task. For further information, refer to Creating Decision Tasks [Seite 1078].

The properties and task description of the selected task are displayed. The task description is displayed to the decision-maker as a decision text at runtime.

Step properties
You can define the following step properties:

- Work items for this step appear in the workflow log [Seite 1420].
- The following step is to be executed immediately by the same recipient [Extern] (for further information, refer to Advancing with Immediate Dialog [Seite 1453]).

Binding
Process the binding between task container and workflow container. This is necessary if the agent is to create a document that is based on a template as an attachment, and if text variables are used in the task description. To do this, select Binding (existing). For further information, refer to Binding Definition in a Decision Task [Seite 1231].

Template for attachments
You can determine a container element in the workflow container that contains a template for an attachment that is to be created. If an attachment is created during the execution of the step, the user receives the template and can add to it.
Creating Decision Tasks

Use
A decision task is only required for steps of the type \textit{user decision}. You should only create a new decision task if:

- You want to restrict the possible agents.
- You require another decision text.

A decision task must always refer to the object type \texttt{DECISION} with the method \texttt{Process}.

Procedure
Always create a new decision task as a copy of the standard decision task supplied by SAP. This is task \textit{generic decision task} (TS00008267).

Go into the step definition of a decision task. A decision task is referenced there in the field \textit{Task}.

If this is task TS00008267 or a copy of this task, select \textbullet.

Specify a new abbreviation and new name for the task and assign it to a development class.

Process the relevant text types on the tab page \textit{Description}. The task description is displayed to the recipient when the user decision is executed. For further information, refer to \textit{Maintaining Task Texts} [Seite 1201].

Add the elements that you inserted into the description text as text variables (or object references) to the task container as \textit{mandatory import elements}. To do this, select \textbullet\textbullet.

In order for the text variables to be filled with values at runtime, you must define the binding from the workflow container to the task container in the definition of the user decision.

Process the possible agents [Extern] of your decision task. Choose \textit{Additional data} \rightarrow \textit{Agent assignment} \rightarrow \textit{Maintain}.
Maintenance of Tab Page Outcomes

Use
The specifications you make on this tab page determine the outcomes of the step for which you want to model a reaction. The outcomes entered on the tab page Decision for the decision options are only displayed here. If reactions to missed deadlines are modeled, the outcomes defined for the relevant missed deadline are only displayed here.

Features
On this tab page you can only activate the outcome Processing obsolete. All other outcomes are determined by the system and activated automatically.

All possible outcomes are listed in the following table:

<table>
<thead>
<tr>
<th>Outcome</th>
<th>This outcome exists if ...</th>
<th>Notes and comments</th>
</tr>
</thead>
<tbody>
<tr>
<td>Decision option</td>
<td>You created options on the tab page Decision.</td>
<td>These outcomes cannot be deactivated. To remove them, you must delete the decision option.</td>
</tr>
<tr>
<td>Processing obsolete</td>
<td>The work item can be set to obsolete via a step of the type process control.</td>
<td>The steps defined after this outcome are executed. This outcome should only be activated if the modeled reaction to a missed deadline contains a step that sets the work item for this step to obsolete.</td>
</tr>
<tr>
<td>Requested end</td>
<td>The relevant deadline monitoring is activated and a modeled reaction required.</td>
<td>Within these branches, you can model a process control step that sets the work item of the step to obsolete. You cannot deactivate these outcomes.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Latest end</th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Latest start</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Activities
You activate the outcomes that you want to actually transfer into the workflow definition. Select ✅ in the relevant line.

Active outcomes are marked with ✅ and have a modeling branch in the workflow definition.

⚠️
If you deactivate an active outcome, the whole of the associated branch is deleted.

For further information about an outcome, select it and choose Detail in the context menu.
Maintenance of Tab Page Notification

Use
The specifications you make on this tab page define the notification agent for completion. This entry is optional. These notification agents receive a notification of completion when the relevant work item is set to status completed.

Activities
Enter a message recipient for completion. For further information, refer to Defining Responsibilities. If no recipient is entered, no notification of completion is sent. The notification of completion contains the completion text that can only be processed in the task definition, and a reference to the work item display of the completed work item.
Maintenance of Deadline Tab Pages

Use
You can define all the deadlines for this step with the specifications you make on these tab pages. All these entries are optional.

Features
You can set the following deadlines on the corresponding tab pages:

- **Requested start [Extern]**
- **Latest start [Extern]**
- **Requested end [Extern]**
- **Latest end [Extern]**

You can define reactions to a missed deadline for the last three deadline types. The system offers the following possible reactions:

- Notification
- Free modeling

Activated deadlines are marked with 🕒 in the tab page index.

You define deadlines with respect to a **reference date/time**. The system offers the following reference date/times:

- The creation date/time of the work item.
- The creation date/time of the workflow to which the monitored work item belongs.
- A date in the form of an expression, which is derived from the context of the application during execution of the workflow.

Activities
You activate monitoring of the relevant deadline by selecting a reference date/time for the deadline. If you choose **expression**, you have to define the reference date/time by specifying expressions [Extern] for the date and time yourself. Use the F4 input help for entering expressions.

The value referenced using the expression must be of data type **D** for the date and data type **T** for the time. If you specify a date but no time, the system sets the time to 00:00:01 (requested and latest start) or 23:59:59 (requested and latest end).

Process the deadline by entering a duration and an appropriate unit. Negative durations only apply if you define the reference date/time yourself via an expression.

The reaction to a deadline being reached depends on the type of deadline:

**Requested start**
If the work item is created before the requested start, it is assigned the status **waiting**. When the requested start is reached, the system sets the work item to status **ready**. If the work item is created after the requested start, it is assigned the status **ready** immediately.

**Latest end, latest start, requested end**
You can either notify someone or model a reaction in the workflow definition for when the deadline is reached.
• Display text (standard escalation)

  If you want to notify someone, you must specify a recipient [Extern] on the tab page Display text. The message text is displayed in the display area. It can only be changed in the task definition.

• Modeled

  If you want to model your own reaction, enter a name for the new outcome on the tab page Modeled. A reaction branch is added to the workflow definition, which is processed if the deadline is missed.

  If a deadline is missed, the original step is not yet completed. The steps following this "missed deadline event" do not end the original activity.

  For further information, refer to Modeled Deadline Monitoring or Standard Escalation [Extern].
Maintenance of Tab Page Methods

Use
In addition to the task's method, you can define the following methods:

- **Before method [Extern]**: Executed before execution of the method.
- **Secondary method [Extern]**: Executed at the same time as the method.
- **After method [Extern]**: Executed after execution of the method.

Prerequisites
The methods and their object types are defined in the *Business Object Repository*. The objects on which the methods are executed must be referenced in a container element in the workflow container.

If you define one of these methods for a user decision, you must delete the entry in the field *Function module* on the tab page *Work item display*.

Features

Calling before and after methods
When executing a work item with before or after methods, the system first calls the before methods synchronously in any order and processes them. Then the system starts the task method and secondary methods in different sessions. Secondary methods are only for display. Any changes carried out in them have no (direct) effect on the workflow. Finally, the system calls the after methods synchronously in any order and processes them.

- The after methods being called is not dependent on the result of the task method. They are called even if the task method terminated with an error.
- The after methods being called is not dependent on whether or not the task method is terminated with an event. They are called immediately after the synchronous part of the method.

With work items that are set to status *completed* by a terminating event without method execution, the before, secondary and after methods are not called. Before methods cannot create the object on which the task method is to run. However, after methods can run on a object created by the task method.

Data transfer into and out of the methods
There is no separate binding between task container and method container for these methods. The methods are given the method container of the task method. If no binding is defined between the task and the method, the system passes the task container. Secondary methods cannot return data to the workflow nor influence the process control. Before and after methods can evaluate and manipulate the method container in order to influence the subsequent method or steps.
Error handling
As with secondary methods, errors in before and after methods have no influence on the execution of a work item. The processing logic of the work item depends solely on the task method. The system records errors in the before and after methods as warnings in the workflow log.

Activities
To define an object method as a before, an after or a secondary method, select (in the column Object) a container element in the workflow container, which contains the object to be processed. Then select the object method required in the column Method.
Maintenance of Tab Page Work Item Display

Use

On the tab page Work item display, you can individualize the work item display with an additional tab page, and the work item preview with graphics or text.

Features

Step-specific tab page for work item display

An individual tab page is added to the work item display. This tab page is displayed in addition to the three tab pages Basic data, Activities and Available objects as the first tab page of the work item display. This additional tab page only appears when work items that represent this step are displayed. This tab page can be used, for example, to:

- Display important information from the current process, which does not appear in the work item display as standard.
- Provide functions that the end user has to execute often.
- Create a customer-specific 'look and feel'.

Adapting the work item preview in the Business Workplace

A user exit is available for individualizing the work item preview. For further information, refer to the User Exit for Work Item Preview [Extern].

A user decision can be executed in the work item preview as standard. The function module required for this is entered automatically.

If secondary methods are to be executed or methods before or after the work item execution in a user decision, delete the entry in the field Function module.

Activities

Step-specific tab page for work item display

Enter the program and the number of the screen into the relevant input fields. For further information, refer to Programming a Step-Specific Tab Page [Extern].

Adapting the work item preview in the Business Workplace

Enter the function module that you want to use in the input field Function module. You can create a function module using a template by selecting the button.
Maintenance of Tab Page Other

Use
On the tab page Other, you can change the priority and repeated execution of the work item.

Features

Step priority
Here you define what priority the work items created for this step have. Dialog work items of priority 1 are displayed in the Business Workplace as express work items.

Table-Driven Dynamic Parallel Processing
You can process this step several times in parallel. The multiline container element entered here controls this parallel processing. For further information, refer to [Table-Driven Dynamic Parallel Processing](#).

Repeat counter for work items with temporary errors
Maximum number of attempts made by the system to restart a background work item with temporary errors. If all of the attempts are unsuccessful, the work item status is set to:

- **Error** (if the temporary exception is not modeled)
- **Completed** (if the temporary exception is modeled)

This function is only available for steps that refer to a single-step task without dialog ("background task"). If the repeat counter has the value 0, the maximum number of retries is determined from the settings in Customizing [Extern].

For further information, refer to [Error Handling for Background Work Items](Seite 1388).
Maintenance of UNTIL Loops

Use
You use an UNTIL loop [Extern] in a workflow definition if:

- At execution time with one of two possible alternatives the workflow is to “jump back” to process sections of the workflow definition again.

- The workflow system can make a decision based on the contents of the workflow container.

In the workflow definition an UNTIL loop is represented with the symbol 🔄.

Features
An UNTIL loop consists of a sequence of steps processed by the workflow system until a termination condition is recognized as true by the workflow system. In an UNTIL loop, the steps are processed at least once before the loop can be terminated.

Outcome of UNTIL Loops
The evaluation of a condition can return the values true or false.
You assign an event name for each of the two values. These names are used to label the two outcomes of the loop in the workflow definition.
In an UNTIL loop, the return jump always occurs when the condition evaluation returns the value false. The workflow execution is continued if the condition evaluation returns the value true.

Activities
To define the loop you must assign names for the step and enter the outcomes. Use the condition editor to enter the termination condition. For further information, refer to Condition Editor [Seite 1012].
The system creates a loop containing an undefined step in the workflow definition. You must then define the loop content.
Maintenance of WHILE Loops

Use
You use a WHILE loop [Extern] in a workflow definition if:

- At execution time, only one of several possible alternatives can be processed in the workflow from a business point of view, and then the comparison is to be performed again.
- The workflow system can make a decision based on the contents of the workflow container.
- The decision can be formulated based on a comparison between an expression from workflow container and a finite amount of comparison values (constants, expressions).

In the workflow definition a WHILE loop is represented with the symbol 🔄.

Features
At runtime, the workflow system compares the value of an expression of the workflow container with the defined comparison values. If the system cannot establish any agreement, the workflow is continued in the Other values branch.
If the value of the expression of the workflow container agrees with one of the comparison values, the steps modeled in the relevant branch are processed. Then the return jump occurs and the system runs the loop check again.

Activities
Assign a step name and define the basis of comparison as an expression of the workflow container. Enter a name for each comparison value. These names identify the various processing branches in the workflow definition.
The branch Other values is created automatically by the system even if you do not assign a name. This ensures that the workflow definition is executable.

Make sure that in each branch of the loop you implement steps that change the basis of comparison in an appropriate way.
Maintenance of Forks

Use
You use a fork [Extern] in a workflow definition when the business process can be continued by several users at the same time. You can also configure the fork in such a manner that not all branches have to be processed.

In the workflow definition, the start of a fork is shown with the symbol and the end with .

Integration
In addition to this explicitly-modeled parallelism, there is also table-driven, dynamic parallel processing and the work queue. For further information, please refer to Implementation Options for Parallel Processing [Extern].

Features
You can create any number of branches in a fork. All branches of the fork flow into a join operator ( ). When one branch of a fork reaches this join operator at runtime, the end conditions are checked.

The system checks whether the number of branches processed agrees with the number of branches required as specified in the definition. The system then checks whether a condition produces the result true.

If one of the end conditions is fulfilled, any existing work items of the fork are set to status logically deleted, and the workflow is continued after the join operator.

The individual branches of a fork should be functionally independent.

Activities
To define a fork, you specify the number of parallel branches required with the step name. The fork, consisting of the start operator, the branches each with an undefined step [Extern] and the join operator, is inserted into the workflow definition.

To define the termination of the fork, you enter the number of branches required and the condition. For further information on defining the condition, please refer to Condition Editor [Seite 1012].

If you make no entry for the number of branches required, the system takes the number of branches.

You can transfer an existing modeled block into a fork as a new branch as follows:

1. Copy or cut the block.
2. Select and choose Paste block in the context menu.
Top-Down Modeling

Use
Top-down modeling allows you to create workflow definitions taking into account block orientation without knowledge of the individual tasks used. At an early modeling stage, it is therefore possible to create incomplete but consistent workflow definitions. It may well be appropriate to define the flow logic initially for complex workflows for example. Top-down modeling is also useful for presentation purposes.

Prerequisites
To model a workflow "top-down", start the Workflow Builder [Seite 1002] and select ☐ in the application toolbar.

Features
Top-down modeling
You can use all the step types available when modeling. The following top-down task can be used when describing an activity in the workflow definition:

<table>
<thead>
<tr>
<th>Task</th>
<th>TS30100074</th>
</tr>
</thead>
<tbody>
<tr>
<td>Abbreviation</td>
<td>WF_MODTASK</td>
</tr>
<tr>
<td>Name</td>
<td>Modeling task for TOP/DOWN approach</td>
</tr>
</tbody>
</table>

You should overwrite the proposed step name with a name that describes the significance or function of this step in your scenario.

The top-down task references an object method with a result [Extern]. The result can assume six values, which are offered in the workflow definition as outcomes [Extern]. You can then choose the number of outcomes required for your scenario and name them. You must deactivate possible outcomes that you do not require. The top-down task can be incorporated in a workflow definition more than once.

It is also possible to maintain the agent assignment to the individual steps.

Productive use
To use the "top-down" modeled workflow productively, you must overwrite the task ID TS30100074 in the definition of the activity with the new ID. The definition of the task entered is read and the binding definition between the workflow container and task container generated if applicable. The outcomes of the activity can be described in the normal way. For further information, refer to Maintenance of Tab Page Outcomes [Seite 1028] When you return from the definition of the activity to the display of the workflow definition, superfluous outcomes of the top-down task are deleted. New branches are created in the workflow definition for additional or unassigned outcomes of the task entered.
Starting Tasks and Workflows

Use
There are various options for starting tasks and workflows. You must be aware of these options when defining a task or workflow.

Integration
A started workflow is represented by a workflow work item [Extern], and a started task by a dialog work item [Extern] or a background work item [Extern].

Features
Tasks and workflows can be started in the following ways.

By an event
The system starts the task or workflow when a triggering event defined for this purpose occurs. The following requirements must be met:

- The event must be defined as a triggering event in the definition.
- The event receiver linkage must be activated.

The linkage may be deactivated by the system if the start was unsuccessful. If this occurs, try to analyze and resolve the error by simulating an event [Seite 1363].

To use data stored with the event in the execution, a binding can be defined from the event container to the workflow container or task container.
A task or workflow can have several triggering events. If one of the triggering events is generated in the system, the relevant execution is started.

Triggering events are only required for a task if the task is to be executed as a single step in response to an event. If you only want to use the task as a step in a workflow, the task does not require any triggering events.

For further information, refer to Creating Triggering Events [Seite 1206].

You can also start a task or workflow defined with a triggering event on a test basis using this event. For further information, refer to Creation of Events for Test Purposes [Seite 1365].

Directly in dialog
You start the task or workflow yourself. For further information, refer to:

- Starting Workflows in Dialog [Seite 1450] for the standard environment at runtime and starting workflows with reference to an application object.
- Starting Workflows (Test Environment) [Seite 1452] for a test environment that allows tasks and workflows to be started.
- Definition of Start Transactions [Seite 1455] for workflows.

A task or workflow started by a triggering event is normally provided with input data from the event container. If you start a task or workflow of this kind in dialog, you
must enter the input parameters manually. Therefore, you will usually only start a task or workflow with a triggering event in dialog for test purposes.

**From a message long text**
This function is only available with messages for which an appropriate assignment is maintained. A user can start the relevant workflow when the message is triggered. For further information, refer to Starting Workflows or Tasks from Message Long Texts [Seite 1456].

**By a workflow**
The system starts a task or workflow if the task or workflow is integrated in another workflow as a step. If the task is executed in dialog, possible agents [Extern] must be assigned to it, or it must be classified as a general task.

> To execute a task as a step in a workflow, the task does not require triggering events.

For further information, refer to Maintenance of Activities [Seite 1024].
Starting Workflows in Dialog

Use
This function can be used to start the workflows you have created in dialog. "In dialog" means that the workflow is not started by a triggering event but manually. It is irrelevant whether triggering events are defined for the workflow.
For further information, refer to Advancing With Immediate Dialog [Seite 1453].

Prerequisites
You can only start tasks in dialog if you are entered as a possible agent [Extern] for them.

Features
Starting workflows
The screen for starting workflows has two parts. The workflows you have defined are listed on the left. The description of the workflow selected on the left is displayed on the right.
You can use the standard toolbar to go to the workflow outbox [Seite 1441] and the Business Workplace [Seite 1368].
You can start a workflow by selecting it and then choosing either in the left screen area or the Start function in the right screen area.

Starting with details
The function Start with details can be used to add information such as attachments, notes or deadlines to the workflow via three tab pages. The attachments and notes are available to all agents of the workflow.

- Tab page Properties
  You can maintain the priority of the workflow, its latest end and its requested start here. If you set the indicator Create flag for started workflow in inbox, you get a work item in your inbox when the workflow is started, which calls the relevant workflow log during execution. The system deletes this flag automatically when the workflow is terminated.

- Tab page Note at start
  The note that you create here is added to the workflow as an attachment.

- Tab page 0 Attachments
  Here you can create ( ), display ( ), import ( ) and delete ( ) attachments for the workflow.

  You start the workflow with the function Start workflow.

Starting via start transaction
If a start transaction [Seite 1455] is defined for the workflow, the defined transaction is started instead of detail maintenance.
As with detail maintenance, you can change the priority, add attachments and maintain deadline data via . In contrast to detail maintenance, you cannot create any new attachments. However, it is possible to create ad hoc objects (in contrast to attachments, these are assigned to the container element _Adhoc_Objects).
Make the entries required in the input fields you have created. This assigns values to the import elements of the interface of the workflow to be started. Use the F4 input help if required.
Starting workflows with object references

Starting a workflow with an object reference allows you to extend a process at runtime and initiate additional subsequent activities on an ad hoc and flexible basis. This function is not available everywhere. The function assumes that the system recognizes as a reference object a particular application object (for example document, invoice, item of master data) that you are processing.

The process flow is as follows:

- The system establishes a reference object [Extern] from the current call context. The current call context is, for example, a work item selected in the workflow inbox of the Business Workplace. This work item can then contain the notification of absence xy as a reference object.

- The system establishes the object type for this reference object.

- The system displays all workflows that refer to this object type so you can make your selection. This uses the same function as Starting workflows (see above). If there is no appropriate workflow, you can create a new workflow ad hoc. For further information, refer to Defining Workflows Ad Hoc [Seite 1098].

- The system starts the selected workflow and passes the reference object to the task.

Activities

The function Start workflow can be accessed via Office → Start workflow or via Tools → Business Workflow → Development → Runtime tools → Start workflow.

The function Start workflows with object reference is available within the Business Workplace in the workflow inbox, the workflow outbox and the workflow resubmissions under Environment.
Starting Workflows (Test Environment)

Use
You should only use this environment for starting tasks in a development system. Here you can start all tasks and workflows, even if you are not one of the possible agents.

Features
In the Task field you can enter the ID of the task [Extern] to be started. This ID consists of two characters for the task type and an eight-digit number. The task type is identified by:
- T (for customer tasks)/TS (for standard tasks)
- WF (for workflow tasks)/WS (for workflow templates)

If you enter a customer task, there must be a blank between the T and the 8-digit number.

To ensure that the respective current organizational plan is taken into account when the task is started, refresh the organizational environment.

The function Input data can be used to enter initial values required for execution into the task's container. This displays a screen with the mandatory import parameters of the task. To display the rest of the import parameters as well, choose Show elements.

The function Deadline data can be used to enter deadlines for the execution of the task. The relevant missed deadline message recipient [Extern] is declared in the definition of a single-step task as the default role, or in a workflow definition as part of the basic data.

The selected workflow is started using . After the start you can branch to the Business Workplace via and the workflow log via .

Activities
You can access the test environment for starting workflows via Tools → Business Workplace → Development → Runtime tools → Start workflow (test environment) or from the Workflow Builder via .
Advancing with Immediate Dialog

Use
The runtime system of SAP Business Workflow supports advancing through work items with immediate dialog as standard.

Features
Advancing with immediate dialog means that when the current agent of a work item in the workflow has executed this work item, the method of the next work item is presented to them directly in dialog provided they are one of the recipients of the next work item. The runtime system behaving in this way means that a user does not have to "detour" via the workflow inbox of the Business Workplace, which would first have to be refreshed, in order to reserve or execute the relevant work item.

A user decides in a user decision to revise the notification of absence rejected by their superior. Since the user is therefore also the recipient of the next activity "Revise notification of absence", the relevant method is executed for them immediately.

The runtime system behaving in this way also, however, means that a work item is no longer offered to all recipients for processing in the status ready, since the first agent in such a chain of work items automatically becomes the current agent of the next work items.

Starting workflows
If Advance with immediate dialog is activated, the method for the first work item appears immediately when a workflow is started in dialog. The initiator of the workflow must, of course, be one of the recipients of this work item.

Since advancing with immediate dialog can only be activated or deactivated for workflows, work items for single-step tasks are always started with immediate dialog.

For further information, refer to Starting Tasks [Seite 1448].

Conditions
The following conditions must be met:

- The task underlying the step references a synchronous method [Extern].
- The step does not have a requested start date in the future, but can be started immediately.

Technical details
At runtime, when a step has been completed (= work item has status completed), the system checks whether the next work item can be started immediately with dialog and whether the actual agent is also a recipient of the next work item.

If this is the case, the work item is reserved for the agent as soon as it is created and the relevant object method is executed.

This procedure continues until more than one work item or no work items are created that can be executed immediately by the actual agent. More than one can occur when forks are used, for example. A user can also leave the "chain" by canceling execution of the method.
Activities

You activate or deactivate *advance with immediate dialog* in the basic data of a workflow definition. This setting then refers to advancing for all steps in the workflows relating to this definition. The first time the Workflow Builder is called, *advance with immediate dialog* is always activated. You can also activate or deactivate *advance with immediate dialog* within the definition of an activity in a workflow definition. This setting then refers only to advancing from the preceding step to this step.

If you have deactivated advancing for the workflow in the basic data, you cannot activate it at the step level.
Defining Start Transactions

Use
The workflow system can create an individual transaction to start a particular workflow. A start transaction provides input fields that are used to assign values to the import parameters of the workflow container. You can edit this transaction at a later date using the ABAP Workbench, for example to change the names of input fields or adapt the layout.

If you select a workflow by choosing Office → Start workflow, for which a start transaction is defined, this start transaction is executed.

For further information, refer to Starting Workflows in Dialog [Seite 1450].

Prerequisites
You have defined a workflow and marked the container elements required as import parameters.

Features
The workflow system generates a transaction to start a given workflow. The input fields are derived from the import parameters of the workflow container.

Container elements that reference a structure cannot be assigned values using input fields.

- The workflow system generates the name of the transaction automatically from the ID of the workflow. A successfully-generated transaction can be recognized by the symbol adjacent to the transaction name.

- The workflow system also generates the name of the module pool automatically if you do not specify anything yourself. You determine the screen number. In this manner, you can create various screens for various workflows in the same module pool.

Activities
Choose Tools → Business Workflow → Development → Definition tools → Workflow start transactions to access the workflow start transactions function.

You specify a workflow ID, a screen number and, optionally, a module pool in the field Program. Select .

Alternatively, you can generate a start transaction for a workflow from the Workflow Builder. To do this, select on the tab page Start in the basic data of the workflow in the Workflow Builder. You can also process existing start transactions from here.

You can use the Screen Painter functions to revise the transaction generated. For further information, refer to the Screen Painter [Extern] documentation.

If you want to change the user interface status of the transaction generated, you must define a separate user interface status. You can use the user interface status DYNPRO as a model.
Starting Workflows or Tasks from Message Long Texts

Use

Workflows or tasks can be started from message long texts.

If there are appropriate (target) elements in the workflow container or task container, the system passes the following data to the started workflow or started work item:

<table>
<thead>
<tr>
<th>Container element</th>
<th>Data type reference</th>
<th>Explanation</th>
</tr>
</thead>
<tbody>
<tr>
<td>ARBGB</td>
<td>T100-ARBGB</td>
<td>Message ID</td>
</tr>
<tr>
<td></td>
<td></td>
<td>(application area, for example 'WO')</td>
</tr>
<tr>
<td>MSGNR</td>
<td>T100-MSGNR</td>
<td>Message number (for example '430')</td>
</tr>
<tr>
<td>MSGTY</td>
<td>SYST-MSGTY</td>
<td>Message type ('E', 'A', 'S', etc.)</td>
</tr>
<tr>
<td>MSGV1</td>
<td>SYST-MSGV1</td>
<td>Message variable 1</td>
</tr>
<tr>
<td>MSGV2</td>
<td>SYST-MSGV2</td>
<td>Message variable 2</td>
</tr>
<tr>
<td>MSGV3</td>
<td>SYST-MSGV3</td>
<td>Message variable 3</td>
</tr>
<tr>
<td>MSGV4</td>
<td>SYST-MSGV4</td>
<td>Message variable 4</td>
</tr>
<tr>
<td>TEXT</td>
<td>T100-TEXT</td>
<td>Message text in which the variables (&amp;) are not replaced</td>
</tr>
<tr>
<td>MESSAGE</td>
<td>Without type, that is C(255)</td>
<td>Message text in which the variables are replaced</td>
</tr>
</tbody>
</table>

This data is available in the container of the task started and can be used to control the process flow.

An application document is incomplete. The application communicates this via a message.

The Start workflow function is available to the user in the dialog box with the message long text. They can use it to start the linked workflow that will help solve the problem.

Features

You are supported in the functions described below by a Workflow Wizard. Choose Tools → Business Workflow → Development → Definition Tools → Wizards → Create "Call Workflow From Message".

How do you link a message to a workflow?

To implement this functionality, you enter the assignment between task and message in table T100W. Maintenance of this table is client-specific.

If the error message long text is displayed and a T100W assignment maintained, a function is displayed with which the relevant task can be executed.

A message can be linked to all possible task types (WS, WF, T, TS).

It is important that the container elements described above have been added to the container concerned (workflow container or task container). Otherwise, the information from the message cannot be evaluated in a meaningful way in the task started.

Workflow definition details

When defining a task that is to be started from a message long text, you create the elements specified above in the container. At runtime, the system then fills the container elements with the relevant values from the message.
Starting Workflows or Tasks from Message Long Texts

The object reference to the application object to which the error refers, is not directly available via the message variables as a rule.

If the message variables are to contain the values of object key fields, the object reference must first be created within the workflow from these key fields. To do this, you must provide an appropriate instance-independent method with import parameters as shown in the following example (object method \texttt{Build}):

\begin{verbatim}
BEGIN_METHOD BUILD CHANGING CONTAINER.
    DATA: KEYFIELD LIKE ....
    SWC_GET_ELEMENT CONTAINER 'MSGV1' KEYFIELD.
    SWC_SET_OBJECTKEY KEYFIELD.
END_METHOD.
\end{verbatim}
Start Conditions for Workflows

Use
You use this function if you want to define additional conditions for the start of a workflow.

You can only define start conditions for workflows that are started by an event.

Integration
Refer also to the following for workflow start conditions:
- Event [Seite 1321]
- Starting Tasks [Seite 1448]

Features
This function enables the definition of start conditions for a workflow. This means that the start of a workflow no longer depends only on whether a specific event is created, but also on the occurrence of additional conditions.

A workflow is to be started when a notification of absence has been created and the creator of the notification of absence belongs to a specific group of people.

Selection area
The following selection criteria are available for selecting a particular workflow:
Start Conditions for Workflows

- **Start condition**
  Here you can search for the name of a particular start condition defined for a workflow.

- **Business object**
  Here you can search for workflows with start conditions using a particular object type.

- **Event**
  All the events for the chosen object type, which are linked with workflows as triggering events are offered for selection here.

- **Workflow**
  Here you can search directly for workflows with start conditions.

If a newly defined triggering event [Seite 1206] is not displayed, check that the event linkage was activated when the workflow was defined.

**Definition area**
In this area you can define additional start conditions for workflows or view and change existing start conditions.
The individual lines of a start condition provide the following information:
- Name of the start condition
- Currency
- Triggering event of workflow
- Start condition

To change the start condition, the name of the condition or the currency, single-click on the relevant area.

The condition editor [Seite 1012] is used to determine conditions.

Select ☑ to activate the start condition (or ☒ to deactivate it). Select ☐ to delete the start condition.

**Personal settings**
You can display and edit personal settings by choosing Extras → Settings. In particular, you can set whether the maintenance of start conditions is to be cross-client or client-specific.

In the cross-client view you cannot activate or deactivate start conditions if they are redefined in a client. They are marked with the symbol 🚧. Start conditions that are not redefined are only activated or deactivated for the current client in the cross-client view.

**Activities**
Choose the following for workflow start conditions: Tools → Business Workflow → Development → Definition Tools → Events → Event Linkages → Workflow Start Conditions.

To create a new start condition, select ☐.
You can also create start conditions directly from the Workflow Builder. To do this, go to the tab page Start in the basic data of the workflow. Click on the cell Start condition in the line of the relevant triggering event. This displays the condition editor directly.
Organizational Plan

Definition
Representation of the task-related, functional structure of your enterprise, created using tools from the Organizational Management component.
This functional organizational plan differs from the administrative enterprise structure and the personnel structure whose elements are relevant to Payroll Accounting (company code, personnel subarea or employee group, for example). These structures are found in their corresponding components.

Use
You can create several organizational plans in different plan versions, this provides you with the following options in Organizational Management:
- in one plan version, you depict your current valid organizational plan which you use for your current business processes (evaluations, Workflow, personnel planning, for example).
- In additional plan versions, you can depict organizational plans as planning scenarios (for Business Process Re-Engineering, for example).
You can compare the current organizational plan with the planning scenarios and transfer data from the simulated structures into the current organizational plan.

Structure
Organizational plans are normally created by assigning objects of the following types to each other:
- Organizational Unit
- Position
- Job
- Task
If you are using your organizational plan for Workflow, the following object types are also available:
- Standard task
- Workflow template
- Task group
The main elements of an organizational plan are:

- an organizational structure [Extern], with which the reporting structure and task distribution are created using organizational units (departments, for example)

- Staff assignments [Extern] for each organizational unit, in which the current persons (employees), users and vacancies are listed

**Integration**

Organizational plans are generally related to objects from other components.

- If Integration with Personnel Administration [Extern] is active, the personal data for the staff assignments comes from the Personnel Administration component.

  These functional (organizational plan) and administrative (personnel and enterprise structures) structures come into contact if a person is assigned to an organizational plan (as the holder of a position) as well as an enterprise or personnel structure (that is, to a personnel subarea etc.).

- If the enterprise structure [Extern] is active, account assignment data can be obtained from **Controlling**.
Working with Screen Areas

Purpose
You use the user interface in the Organization and Staffing or Organization and Staffing (Workflow) view to create, display and edit organizational plans.
The user interface is divided into various areas, each of it which fulfills specific functions.

Together, the search area and the selection area make up the Objektmanager [Seite 1257]
In the search area you can choose between different object types. Depending on whether you are in the Organization and Staffing or Organization and Staffing (Workflow) view, you are offered different object types to choose from.

Each time you access the Organization and Staffing or Organization and Staffing (Workflow) views, the objects that you edited last are automatically called up, though (if appropriate) with a new date. You can continue editing the objects directly. In addition, the last user-specific settings relating to the size of the screen, and in some cases the preview period, are available again.

You can reset user-specific settings in the object manager using the report RH_DELETE_OM_USER_SETTINGS. For more information, see the report documentation.

You can reset the last object selection and the preview period using the report RH_DELETE_NF_USER_SETTINGS. For more information, see the report documentation.

Note that the reset takes place across all applications.
Prerequisites
Before you work with screen areas in the Organization and Staffing or Organization and Staffing (Workflow) views, you must be familiar with Validity [Seite 1251].
You have decided to use either the Organization and Staffing or the Organization and Staffing (Workflow) view. If necessary, choose the required view over Goto → Change view.

Process Flow
A typical work process could look like the following:
2. In the Search area, find [Seite 1259] one or more objects that you want to display or edit, for example:
   - a complete organizational structure
   - all objects of a certain object type, positions, for example
   - one or more objects of a certain object type, for example, a particular organizational unit
3. The objects found are listed in the Selection area. Select one of these objects
   - by double clicking on it, if you want to display [Seite 1263] the object itself including its environment in the overview area and its characteristics in the detail area
   - by clicking on it, if you want to assign [Seite 1269] it to another object using drag & drop; you can assign a position to an organizational unit, for example
4. In the Overview area
   - the selected object including its environment is displayed.
     According to the object type the environment can be displayed as
     • an organizational structure
     • a report hierarchy
     • staff assignments in the form of a list or a tree structure
     • task assignments
     • organizational assignments
     • job assignments
     • a task hierarchy
     • an agent
     • account assignments
   - you can, if required
     • switch between these representations of the object’s environment using ➤. 
     • create [Seite 1266], copy [Seite 1268], assign/move/reposition [Seite 1269], or delimit/delete [Seite 1271] new objects.
     • select another object
5. In the Detail area, characteristics of the selected object are displayed [Seite 1263] on tab pages. You can edit [Seite 1263] the characteristics of this object or add new ones, as required.
Using [icon], you can hide the detail area, so that you can enlarge the overview area. You can display the detail area again using [icon].

In customizing you can define which tab pages should be displayed. For further information, see the Implementation Guide (IMG) under *Personnel Management* → *Organizational Management* → *Hierarchy Framework.*
Validity

Use
An enterprise’s organizational plan is constantly undergoing change. For this reason, *Organizational Management* allows you to edit the organizational structure, staff assignments as well as individual objects according to key dates.

You must be familiar with the concept of validity as it is relevant to every step in the *Organization and Staffing* and *Organization and Staffing (Workflow)* views.

Prerequisites
So that you can use validity completely, make sure

- that you know the period for which you want to find and select objects
- when you are creating objects and object characteristics, that you specify the period in which they are to be valid
- when you are assigning objects and object characteristics that you specify the period in which the assignments are to be valid

Features
You determine the validity of an object or assignment when you create it. You control the display of objects by entering key dates and preview periods. For more information, see [Validity of Objects and Assignments](#)

You determine the validity of object characteristics when you create them. Changes to data are presented in periods. For more information, see [Validity of Object Characteristics](#).
Validity of Objects and Assignments

Use
So that the presentation of your organizational plan can be exact, objects and assignments between objects exist for a specific period of time. You determine the validity period of objects and assignments when you create them, as required.

Features

Key Date and Preview Period
A key date and a preview period are always set in the Organization and Staffing and the Organization and Staffing (Workflow) views.

- Every time you log on, the current date is set as the key date. You can change the key date. Data valid on the date you have selected is displayed.

- When you logon initially, a preview period of 3 months is set, that is, all changes to data that happen in this period are displayed. You can change this preview period. Next time you log on, the preview period which you selected is set.

System Response
When you create [Seite 1266] an organizational unit or position in the overview area,
- the validity of the object begins with the key date set, this can be moved forward in the detail area
- the assignment begins with the key date set
If you want the validity of the organizational unit, position or assignment to be restricted, you can terminate them. For more information, see Terminating Organizational Units [Extern], or Terminating Positions [Extern].
If you are creating a job or a task, specify the validity date in the dialog box. For more information, see Creating Jobs [Extern] or Creating Tasks [Extern]. Task and job assignments begin on the key date set.
If you are finding objects [Seite 1259] in the search area, the system selects all the objects that exist on the key date and preview period you set, and presents them for selection in the selection area.
Objects whose validity begins only in the preview period set, are indicated by a in the selection area.
Objects, whose validity ends in the preview period, are indicated by a .
If you have selected an object to be displayed [Seite 1263], in the overview area, according to object type
- the relevant organizational structure and the organizational units assigned to it on the key date or in the preview period is displayed.
- the relevant staff assignments and the positions, persons (employees) or users assigned to it on the key date or in the preview period are displayed.
- the relevant staff assignments and the positions, persons (employees) or users from an organizational unit assigned to the selected job on the key date or in the preview period are displayed.
- the relevant task assignment and the objects assigned to it on the key date or in the preview period is displayed.
Objects whose assignment to another object begins only in the preview period set, are indicated by a .
Objects whose assignment to another object ends in the preview period, are indicated by a .
If you assign an object to another object in the overview area or detail area, the new assignment applies from the key date to the end of the validity of an object. If you have activated Query time period for organizational changes, assignments of organizational units and positions are valid in the period you entered in the Query time period for organizational changes dialog box.

Assignments can only be made in the detail area in key date mode. Change the mode by choosing .

**Activities**

Before you perform a step, check the key date and preview period that are set. Change the settings as required. For more information, see Selecting a Date and Preview Period. Activate/deactivate Query Time Period for Organizational Changes.
Selecting a Date and Preview Period

Use
To display and edit data according to key dates and time periods, select a key date and preview period.

Procedure
5. Choose Settings → Date and Preview period (or 
   The Define Date and Preview Period dialog box appears.
6. Enter the key date in the Start Date field.
7. In the Preview period field, enter the preview period.
   When you confirm your entries with Return, the preview period you selected is displayed.
   Correct your entries as required.
8. Confirm your entries by choosing .

Result
You have set a date and a preview period. You can change these settings at any time.
Activating/Deactivating Query Time Period

Use
The validity of assignments normally begins on the key date you have specified. Alternatively, you can activate a dialog box, in which you can enter any validity date when assigning objects (this does not apply when you are assigning persons (employees) or users to positions).

Procedure
4. Choose Settings → Query time for organizational changes
   The Query time for organizational changes dialog box appears.
5. Select the Query time for organizational changes field, if you want to enter a validity date when you are assigning objects. Otherwise, you can deselect this field.
6. Confirm your entries by choosing ✓.

Result
If you have activated the time period query, a dialog box will appear when you move or assign objects, you can enter the validity of the assignment in this dialog box.
If you have deactivated the time period query, the validity of the assignment is determined according to the key date.
Validity of Object Characteristics

Use
You store the characteristics of an object on tab pages in the detail area. The characteristics of an object can change with time. For this reason, an exact representation of your organizational plan is available according to a specific time period. You determine the validity of object characteristics, according to your requirements, when you create them.

Features

Key Date and Periods
In the Organization and Staffing and the Organization and Staffing (Workflow) views
- period mode is always set, which displays all existing periods
- key date mode is set, this displays the key date and the current period

System Response
Each time data is changed using a tab page in the detail area, a new period is created for this tab page. It is valid from the date in the Valid from field and until the date in the Valid to field. The previous period is automatically terminated on the day before the new period.
A new period is also created, if you make a change outside the detail area, which affects the tab pages, for example, if you create a new position in the overview area.
In key date mode, if more than one period exists for a tab page, this is indicated by a 📅. This indicated that assignments which you are making are valid alongside the current period. If there is only one period, this is indicated by a 📅. In this case, the duration of the new assignment includes the duration of the current period.

Activities
Scroll through the existing periods for object characteristics using ⬅️ and ➤️ or ⬅️ and ➤️. Alternatively, you can use 📅 to select a period directly.
When you are changing data, enter the date from when the data is to be valid in the Valid to field, and the date until it is to be valid in the Valid to field.
By selecting 📅 switch to key date mode, to assign objects in the detail area.
Object Manager

Use
With the object manager you can search for and select objects that you want to display or edit.

Prerequisites
You are familiar with the validity concept of the application. The validity concept determines which objects you can find during a search.

Features
The object manager consists of the search area and the selection area.

9. In the search area are one or more search functions for each object type, for example the Search Term and Structure Search functions. These search functions are marked with an icon. In addition, the object type itself can contain a search function. The object types are marked with the respective object type-specific symbol.

   If necessary, you can add more object types and search functions in customizing. You can also change the sequence of the search functions. For further information see the Implementation Guide (IMG) under Personnel Management → Global Settings in Personnel Management → Settings for Object Manager or the IMG for Organizational Management.

10. In the search area you can create search variants, so that you can reuse search criteria you have grouped together, or hits. These search variants are marked with an icon.
11. In the selection area the system displays the objects that you searched for and actually found. According to the search function, this can be either a hit list or a structure.

12. You can scroll through search results in the selection area using ← and →.

13. You can completely hide or display the object manager, so that the other screen areas get correspondingly bigger or smaller. To do that, choose Settings → Show Object Manager or Hide Object Manager.

14. With \[→\] you can increase or reduce the size of the selection area, in order to show more hits. As you do that, the search area is hidden or displayed accordingly.

\[→\]

The system saves the last settings relating to screen size and the last object selection user-specifically, and they are available next time you call up the transaction.

You can reset these settings and the last object selection using the report RH_DELETE_OM_USER_SETTINGS. Note that the reset takes place across all applications. For more information, see the report documentation.

- You can call up generic object services for a selected object using the right mouse button menu. To do that, select an object in the selection area with the right mouse button and choose Generic Object Services. For further information about generic object services, see List of Object Services [Extern].
Finding/Selecting Objects

Use
In the Search area you can search for one or more objects that you want to display or edit. For each object type there are various search functions, for example:

- Search Term
- Structural Search
- Free Search

Prerequisites
Objects you are searching for must already exist. Create new objects as required. The required object types and search functions are set up.

Procedure
2. In the search area select one of the available search functions and if necessary enter the required selection criteria.
   In the selection area the system displays all found objects with the corresponding object type, either as a list or as a structure.
3. Double-click on the required object.
Using Search Tools

Use
With the search tools for each object type you can search for objects in various object type-specific ways. These search tools are marked with \[\text{\textsuperscript{1}}\]. In addition, the object type itself can contain a search tool. The object types are marked with the respective object type-specific symbol.

Prerequisites
You are familiar with how the search tools are assigned.

15. In the search area, the object type itself can contain a search tool. The object types are marked with \[\text{\textsuperscript{1}}\].

16. Search tools can be positioned under the corresponding object type. In that case it is marked with \[\text{\textsuperscript{1}}\].

Procedure

Searching for Objects Using a Search Term
2. Choose \textit{Search Term} (or the required object type, if it has this search function).
   The \textit{Search for <object type>} dialog box appears.
5. Enter a name. This can be a name, abbreviation or numeric ID. You can also search using the entry \[\text{\textasteriskcentered}\].
6. Restrict the number of hits, if required. Enter whether the object you are looking for is \textit{directly} or \textit{indirectly} assigned to another object.
7. Choose \textit{Search}.
   The results of the search are displayed in the \textit{selection area}. The display in the overview and detail areas does not change.
7. If necessary, you can start another search for the same object type and then choose \textit{Insert} to add the new hits to the first results in the selection area.
8. Double-click on the required object in the selection area.

Searching for Objects Using Free Search
The \textit{Free Search} search tool uses the \textit{InfoSet Query}.
3. Choose \textit{Free Search} (or the required object type, if it has this search function).
   The \textit{Find Objects of Type <Object type>} dialog box appears. For further information, see \textit{HR in the InfoSet Query} \([\text{Extern}]\). After the search, the system displays the search results in the \textit{selection area}.
4. Double-click on the required object in the selection area.

Searching for Objects Using Structure Search
2. Choose \textit{Structure Search} (or the required object type, if it has this search function).
   In the \textit{selection area} the system displays all found objects of the relevant object type in a tree structure, ordered according to their assignment in the organizational plan.
If necessary, you can refresh the hits displayed in the selection area using \( \text{refresh} \). This is recommended, for example, if you have created new objects shortly before.

3. Expand the structure until the required object is revealed.

4. Double-click on the required object in the selection area.
Using Search Variants

Use
In some search functions, for example *Search Term*, you can restrict the number of hits by using a combination of selection criteria. You can then save such a combination as a search variant so that you can use it again. You can also delete a search variant again, if necessary.

Procedure

Creating Search Variants
2. Use one of the search functions to search for objects.
   The system displays the hits in the *selection area*.

3. Choose and enter a name. Choose ✓.
   The system saves the search criteria as a search variant and assigns them to the corresponding object type in the *search area*. The search variant is marked with ✓.

4. Select the search variant you created and choose ✓. You can check your search criteria.

Searching for Objects Using a Search Variant
3. Select a search variant.
   The hits are displayed in the *selection area*.

4. Double-click on the required object.

Deleting Search Variants
3. Select the search variant that you want to delete.

4. Choose ✓.
   The search variant is deleted.
Displaying/Editing Objects

Use
You can allocate numerous characteristics to the organizational objects in an organizational plan. Select an object in the **selection area** or **overview area** so that you can display the characteristics of the object in the **detail area** and, if required, edit them.

Prerequisites
You are familiar with **Validity [Seite 1251]**. Select a key date and preview period. It can be that the detail area is hidden. In order to display and edit object characteristics, you may need to redisplay the detail area using ![Redisplay](image).

Procedure

Selecting and Displaying Objects
You can select objects either in the selection area or in the overview area:

- In the **overview area**, select an object by double-clicking on it.

You can only select an object from the selection area if you have already searched for one or more objects in the **search area [Seite 1259]**.

The object itself is displayed in the **overview area**. The following display types are possible:

- **Organizational Unit**
  - **Organizational structure**: structural display showing the assignment of an organizational unit to other organizational units.
  - **Task assignment**: structural display showing the assignment of an organizational unit to tasks and activity groups.
  - **Report hierarchy**: structural display showing the assignment of an organizational unit to a chief position and its holder (person or user).
  - **Staff assignments (list)**: list display showing the assignment of an organizational unit to the subordinate positions and their holders.
  - **Staff assignments (structure)**: structural display showing an organizational structure and the assignment of each organizational unit to the subordinate positions and their holders.

- **Position**
  - **Task assignment**: structural display showing the assignment of positions to tasks and activity groups.
  - **Report hierarchy**: structural display showing the higher- and lower-level assignments of positions.
  - **Organizational assignment**: structural display showing the assignment of a position to higher-level organizational units.
Job

- Task assignment: structural display showing the assignment of jobs to tasks and activity groups.
- Job usage: structural display showing the assignment of a job to positions.

Person

- Organizational assignment: structural display showing the assignment of a person over positions to higher-level organizational units.
- Task assignment: structural display showing the assignment of persons to tasks and activity groups.
- Report hierarchy: structural display showing the higher- and lower-level assignments of positions and their holders.

User

- Organizational assignment: structural display showing the assignment of a user over positions to higher-level organizational units.
- Task assignment: structural display showing the assignment of users to tasks and activity groups.

Task, Standard Task, Workflow Template, and Task Group

- Task hierarchy: structural display showing the assignment of a task to other tasks and activity groups.
- Agent: structural display showing the assignment of a task to other organizational objects.

You can display the exact evaluation path for the current display using .

To switch between displays, select the required object and choose . The system proposes the various display types available.

To move up a level within a displayed structure, choose .

- In the overview area, double-click on an object that is displayed in a structure or list. The characteristics of the selected object are displayed in the detail area.

Scrolling

In the overview area you can scroll through the displays using and . If you have already made changes to data but have not yet saved them, you can use and to undo or recreate the last change.

In the detail area, you can use or and to scroll through the periods that exist for an object.

Editing the Characteristics of an Object

In the detail area you can edit the characteristics of the object you selected.

4. Select a tab page.

5. Make the appropriate entries. For more information, see

   Editing Organizational Units [Extern]
   Editing Positions [Extern]
6. Save your entries by choosing ☑.

On tab pages containing characteristics there is a ☑.
Creating Objects

Planstelle bearbeiten [Extern]
Stelle bearbeiten [Extern]
Aufgabe bearbeiten [Extern]

Use
You depict the functional structure of your enterprise as an organizational plan [Seite 1708], by creating organizational objects (organizational units, positions etc.) in the overview area and assigning them to objects that already exist.

Prerequisites
To create a root organizational unit for a new organizational plan, you must be in Create mode (Organization and Staffing Create or Organization and Staffing (Workflow) Create).

Create additional organizational objects
• in Create mode, if you are going to create additional objects straight away once you have created the root organizational object, without leaving the mode.
• in Change mode (Organization and Staffing Change or Organization and Staffing (Workflow) Change), if you have left create mode.

You are familiar with Validity [Seite 1251]. Select a key date and preview period.
Which objects you can create depends on which display type is currently in the overview area. If necessary, check which assignments can be displayed in your particular case, using . If necessary, use to switch to a display type in which you can carry out the required assignment.
For more information on the display types, see Displaying/Editing Objects [Seite 1263]

Procedure

Creating Root Organizational Units
You have chosen create mode.
2. In the Create a root organizational unit dialog box, enter the required validity period of the root organizational unit and confirm with .
   • The system creates a root organizational unit with the provisional name “New Organizational Unit”.
4. In the detail area, replace the preset entries for object name and object description on the Basic data tab page with the name of your choice.
5. Save your entries by choosing .
   • You have created your root organizational unit.

Creating Organizational Objects in the Structural Display
5. In the overview area, select the object to which you want to assign the object being created.
6. Choose .
   • The Choose Relationship dialog box appears.
Creating Objects

7. Select an object type.
   The system creates the object.

8. Over the tab pages in the **detail area**, edit the characteristics of the new object.
   For further information according to object type, see:
   - [Editing Organizational Units](#)
   - [Editing Positions](#)
   - [Editing Jobs](#)
   - [Editing Tasks](#)

**Creating Positions in Staff Assignments (List)**

2. Choose 📐.
   The system creates the new position.

3. Over the tab pages in the **detail area**, edit the characteristics of the new object.

   You can also create new objects by copying. For more information, see [Copying Objects](#).

   You can always assign objects to other objects – not just when you create them. For more information, see [Assigning Objects](#).
Copying Objects

Use
You can create a new object by copying an object which already exists. When you do this, all the characteristics of the object are copied.

Prerequisites
You are familiar with Validity [Seite 1251]. Select a key date and preview period.

Procedure
3. In the overview area, select the object which you want to copy.
4. Choose 
   The Copy Object dialog box appears.
4. Specify the number of copies you wish to make and the validity period which is to apply to the new objects.
5. Choose 
   The new object is created. The object’s tab pages will be filled with the characteristics of the template. The characteristics can be changed or enhanced as required.
6. Save your entries by choosing .
Assigning/Moving/Repositioning Objects

Use
An organizational plan depicts the functional relationships in your enterprise. To depict such relationships (the hierarchy of organizational units or the staffing of positions by persons, for example), you assign objects to each other. Objects can be

- newly assigned if they are not yet assigned.
- moved, that is, you end the object’s current assignment in a structure and create a new assignment within the same structure.
- repositioned within a hierarchy level in a structure.

The following can be target positions:

- An organizational unit can be assigned to objects of the following types:
  - Organizational Unit, Task, Standard Task, Workflow Template, Task Group

- A position can be assigned to objects of the following types:
  - Organizational Unit, Task, Standard Task, Workflow Template, Task Group

- A person (employee) or user can be assigned to objects of the following types:
  - Position, Task, Standard Task, Workflow Template, Task Group

- A position can be assigned to objects of the following types:
  - Position, Task, Standard Task, Workflow Template, Task Group

- A task, a standard task, a workflow template or a task group can be assigned to objects of the following types:
  - Organizational Unit, Position, Job, Person, User, Task, Standard Task, Workflow Template, Task Group.

Prerequisites
You are familiar with Validity [Seite 1251]. Select a key date and preview period.

You can only assign standard tasks, workflow templates and task groups in the Organization and Staffing (Workflow) view.

Which objects you can assign/move/reposition depends on which display type is currently in the overview area. If necessary, check which assignments can be displayed in your particular case, using If necessary, use to switch to a display type in which you can carry out the required assignment.

For more information on the display types, see Displaying/Editing Objects [Seite 1263]

Procedure
Assigning Objects

5. Search for an object that already exists, but does not yet belong to the current structure. For more information, see Finding Objects [Seite 1259].

6. In the selection area, select the object that you want to assign to another object.
Assigning/Moving/Repositioning Objects

7. Holding down the left mouse button, drag the object to the target position. The target position can be an object in a structural display or a field in the staff assignments (list) display.
   The system assigns the object.
8. Save your entries by choosing ![save_icon].

Moving Objects
4. In the overview area, select the object that you want to move.
5. Hold down the left mouse button and drag the object to the object to which you want to assign it.
   The object has moved and is therefore newly-assigned.
6. Save your entries by choosing ![save_icon].

Repositioning Objects
4. In the overview area, select the object that you want to reposition.
5. Choose ![up_icon] or ![down_icon] to reposition the object in a higher or lower position within the hierarchy level.
   The object has been repositioned. Therefore, the sequence of objects on a hierarchy level has changed.
6. Save your entries by choosing ![save_icon].

Assigning/Moving Using a Pushbutton
As an alternative to assigning or moving with drag & drop, you can also use the ![assign_icon] pushbutton.
5. In the overview area, select the object that you want to assign to another object.
6. Choose ![assign_icon].
   The Restrict allowed values dialog box appears.
7. Search for the object that you want to assign, select it and then choose ![assign_icon].
8. Save your entries by choosing ![save_icon].
Terminating/Deleting Objects or Assignments

Use
You can:
- Terminate organizational objects if you want to limit their validity.
- Terminate the assignment of an organizational object to another organizational object if you want to limit the validity of this assignment.
- Delete an organizational objects, if you want to delete them completely, including their history.
- Delete the assignment of an organizational object to another organizational object, if you want to delete this assignment and its history completely.

Prerequisites
You are familiar with Validity [Seite 1251]. Select a key date and preview period. Choose key date mode in the detail area.
Which objects/assignments you can terminate/delete depends on what is displayed in the overview area. Check which assignments can be depicted in a particular case by choosing . If necessary, switch to a display in which you can terminate/delete the object or assignment by choosing .

For more information on the displays, see Displaying/Editing Objects [Seite 1263].

Procedure
5. In the overview area, select the object which you want to terminate/delete or whose assignment you wish to delete.
6. Choose if you want to terminate, or if you want to delete.
7. Choose whether you want to terminate/delete the object or the assignment.
   - If you want to terminate, enter the date from which the object/assignment is to be terminated in the dialog box, and choose .
   - The object / the assignment is terminated/deleted.
8. Save your entries by choosing .
Undoing/Recreating Changes

Use
If you have already made changes to data in the overview area, but have not yet saved them, you can use and to undo or recreate them.

Prerequisites
You have changed data but you have not yet saved.
A change (no assignment) in the detail area: You have confirmed the change with Return, or you carried out another activity after the change, chosen another tab page, for example.

Procedure
2. Choose to undo the last change.
   
   The change is undone.
3. Choose to recreate the last change.
   
   The change is recreated.
4. Save your entries by choosing when you want to finally confirm your changes.
**Confirmation Prompt**

**Use**
Once you have carried out a specified number of changes to data (specified in Customizing), a dialog box appears asking whether you want to save these changes. Thus you can avoid losing data.

**Prerequisites**
You define the number of steps that are carried out before the confirmation prompt appears in Customizing. For further information, see the Implementation Guide (IMG) under *Personnel Management* → *Organizational Management* → *Framework* → *Set Up Confirmation Prompt*.

**Activities**
Save data as required.
Configuring Columns

Use
You can decide which columns should be displayed in different screen areas.

Prerequisites
The columns that are available are determined in Customizing. For further information see the Implementation Guide (IMG) under Personnel Management → Global Settings in Personnel Management → Column Framework or the IMG for Organizational Management.

Procedure
4. Choose the Column configuration dialog box appears.
5. Select the columns that you want to display.
   Some entries represent column groups, that is, more than one column will be displayed if you select one of these entries.
6. Confirm your entries by choosing ✓.

Result
The columns you selected are displayed.
You can reset all of a user’s column configurations using the report RH_DELETE_COL_USER_SETTINGS. Note that the reset takes place across all applications. For more information, see the report documentation.
Switching Maintenance Interfaces

Use
You can replace the Organization and Staffing and the Organization and Staffing (Workflow) views with Simple Maintenance. You may want to switch if, for example, you have already used Simple Maintenance and you want to work with the maintenance interface you are used to. For more information on Simple Maintenance, see Simple Maintenance [Extern]. In addition, you can jump to Infotype maintenance from a selected object. For more information, see Infotype Maintenance [Extern].

Procedure

**Situation A:** You are in Organization and Staffing or Organization and Staffing (Workflow) view and want to switch to Simple Maintenance. Choose Settings → Maintenance Interface. You access Simple Maintenance.
If you do not change this setting, you will automatically access Simple Maintenance when you call up a maintenance interface in the future.

**Situation B:** You want to reverse this setting. You are in Simple Maintenance. Choose Settings → Maintenance Interface. The Organization and Staffing interface appears.
When you call up a maintenance interface in future, the Organization and Staffing view will appear.

**Situation C:** You are in neither of the two maintenance interfaces; you want to select an interface.
In the SAP menu choose Human resources → Organizational management → Settings → Set maintenance interface. The Set maintenance interface dialog box appears. Select a maintenance interface and confirm by choosing ✔. If you do not change this setting, you will automatically access the interface you select when you call up a maintenance interface in the future.

**Situation D:** You have selected an object in the Organization and Staffing maintenance interface and displayed its characteristics in the detail area. You want to go to the infotype maintenance for this object. Choose Goto → Detail object → Enhanced object description. To get back to the Organization and Staffing view, choose ✔.
Event

Definition
Occurrence of a status change in an object.
An event is created from any application program (the event creator) and "published" system-wide. Any number of receivers can respond to the event with their own “response mechanisms”.

Object: Invoice, vendor Miller
Event: Entered

Use
You can use an event:

- As a triggering event [Extern] of a task or a workflow.
  When the event occurs, the task or workflow is started as a response.

- As a terminating event [Extern] of a task.
  When the event occurs, the task is terminated as a response.

- In workflow definition steps of the types event creator [Extern] and wait for event [Extern].

Event receiver linkage at definition time
An event is published without the creating application being informed as to whether a receiver reacts to this event. The system enters potential receivers in a linkage table [Extern], which is evaluated by the event manager [Extern].

Events at runtime
In its event container, each event has information about the context of its creation, which is available to the event receiver. The information can be used for event-driven control and communication mechanisms. The most important event parameter is the reference to the object whose status change the event announces. (The information about the type of status change is part of the name of the event.)

Integration
An event is always defined as a component of an object type [Extern].

The actual creation of an event is not implemented in the program of the object type.

Creation of additional events
You can extend an object type on a customer-specific basis using the delegation concept by adding more events. You must then also cater for the creation [Seite 1322] of these additional events.
Event Creation

Use
Events are created when the relevant status change occurs. The event creation must be implemented by SAP or yourself, or catered for by appropriate table settings.
Usually, the event creation is implemented in the application programs for the events supplied by SAP.

Customer-specific events
You can also create additional events yourself for specific status changes, which are not provided by SAP in the standard version. The creation of these events is catered for by table settings.

Features
When an event is triggered, the workflow system checks the event receiver linkage table to see whether receivers are entered for this event. If any are found, these receivers are called using the function module also located in the linkage table.

There are various ways in which events can be created:

<table>
<thead>
<tr>
<th>Function module</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Event creation by calling function module SWE_EVENT_CREATE or SAP_WAPI_CREATE_EVENT. An event can be created from any program by calling the function module SWE_EVENT_CREATE or SAP_WAPI_CREATE_EVENT. Knowledge and experience of the programming environment are required. The ID of the event, the ID of the triggering object type and the object type-specific key are passed to this function module as import parameters, and the event container as a table parameter. For further information, refer to Creation of Events by Calling a Function Module [Seite 1340].</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Change documents</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Event creation when change documents are written You can set the following system behavior by making entries in the relevant control tables of change document management: Status changes of objects which are logged as change documents are reported automatically as events as well. For further information, refer to Creation of Events When Change Documents Are Written [Seite 1325].</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>General status management</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Event creation upon status changes You can set the following system behavior by making entries in the relevant control tables of status management: The system automatically creates an event for objects that use general status management when an object status changes. For further information, refer to Creation of Events Upon Status Changes [Seite 1334].</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Message control</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Event creation as message type Events can be created via a connection to Message Control. For further information, refer to Creation of Events via Message Control [Seite 1337].</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Logistics Information System (LIS)</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Event creation when an exception situation occurs (LIS exception) The occurrence of an exception situation defined within the LIS as an exception is to cause an event to be created. For further information, refer to Creation of Events Upon LIS Exceptions [Seite 1338].</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Business Transaction Events</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>For further information, refer to Creation of Events via Business Transaction Events [Seite 1339].</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>HR master data</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Event creation when HR master data changes To create events when HR master records change, you must carry out the following Personnel Management Customizing activities:</td>
<td></td>
</tr>
<tr>
<td>----------------</td>
<td>-------------</td>
</tr>
<tr>
<td>● Activate event linkage [Extern]</td>
<td></td>
</tr>
</tbody>
</table>

April 2001
**SR master data**  
Event creation when HR master data changes

| To create events when HR master records change, you must carry out the following **Personnel Management Customizing activities**:
<table>
<thead>
<tr>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Activate event linkage [Extern]</strong></td>
</tr>
<tr>
<td><strong>Define event types for customer-specific business objects [Extern]</strong></td>
</tr>
<tr>
<td><strong>Redefine event types for SAP business objects [Extern]</strong></td>
</tr>
</tbody>
</table>

These Customizing activities can be found in the SAP Reference IMG under **Personnel Management → Global Settings in Personnel Management → Business Workflow Events**.

### Activities

You can use a [wizard](#) for the creation of certain events. If you cannot use a wizard, you must adhere to the following procedure when creating customer-specific events:

1. Define the event that you want to create as a component of an object type which you have created in the customer namespace.

   Usually you create this object type as a subtype of an object type supplied by SAP. This extends the functionality inherited from the supertype. But you can also create your own object type in the **Business Object Repository** without using inheritance from a supertype.

   This procedure is necessary because customer-specific events cannot be defined for object types supplied by SAP. These object types have program character and you cannot change them.

   For further information on creating an object type as a subtype, refer to [Creating Object Types](#).

2. Make this subtype into a [delegation type](#) of the initial object type.

   For further information, refer to [Adapting Object Types on a Customer-Specific Basis](#).

3. Maintain the relevant table settings to create your event.

   For further information, refer to [Tutorial: Event Creation Upon Status Changes](#).
Wizards for Event Creation

Use
You can use these wizards to create events, which can then be used within the definition of workflows.

Integration
The wizards do not cover all aspects of event creation [Seite 1322]. For alternative ways of creating events, choose Tools → Business Workflow → Development → Definition tools → Events → Event creation.

Features
Wizards are available for the following:
- Change documents
- Logistics Information System
- Business Transaction Events

The wizards are mostly self-explanatory and have documentation. To display the relevant documentation, select Click here for further information...

Activities
To execute the individual wizards, choose Tools → Business Workflow → Development → Definition tools → Events → Event creation → Set up with wizard.
The wizard is set up with wizard.
Then select the wizard that you want to execute.
Creation of Events When Change Documents are Written

Use
Event creation can be connected to the writing of a change document without modifying an existing application.

Integration
Many business objects are changed frequently. It is often useful and even necessary to be able to trace the changes made. This logging is carried out with change documents. These can be created for changes already made as well as for planned changes.
To be able to log changes to a business object in a change document, an appropriate change document object must be defined in the system. In its definition, a change document object has tables, which represent a business object in the system. Changes to table fields designated as change document-relevant are logged by writing a change document.

Changes made to the material master data in the field MARA-WRKST (basic material) are logged as standard on the part of the application in a change document for the change document object MATERIAL.

For further information about change documents, refer to the documentation Change Documents [Extern].

Features
The assignment between a change document object and a business object type/event must be maintained in a system table. The system then creates the event whenever a change document is written for the change document object. The change document is written when the change is updated. The procedure described, putting the event after the logging, ensures that the event is not created until the change has actually been made.
The system fills the event container of the event created with information about the change document. For further information and advice on how to use these event parameters, refer to Creation of Additional Event Parameters With Event Creation [Seite 1332].

Activities
- Assign a change document object to an object type/event pairing and determine the action (create, change or delete) on the application object for which the event is to be created. You can classify the event in order to create three different events for a change document, if appropriate from a business point of view.
  For further information, refer to Maintaining the Assignment Between Change Document and Event Without Field Restrictions [Seite 1327].
- You may restrict the event creation with regard to whether the change document or change meets certain conditions.
  For further information, refer to Maintaining the Assignment Between Change Document and Event With Field Restrictions [Seite 1329].
You may specify a function module with which you can still perform operations on the event container immediately before the event manager is called, that is, before possible event receivers are determined.

This function module is only required under specific circumstances.
Maintaining the Assignment Between Change Document and Event Without Field Restrictions

Prerequisites
The event that you want to create is defined as a component of an object type.

Procedure
You make the following settings to create an event each time a particular change document is written.

2. In the first view Change "Events for Change Document": Overview, create a new entry. To do this, select the function New entries.
3. Specify a change document object in the field Chng.doc.obj.
4. In the field Business obj. type, enter the object type whose status change is to be indicated by the event created. Use the F4 input help.
   
   The object types offered to you by the system in the F4 input help usually have the same key structure as the change document object.

   It may, however, make sense from a business point of view to create the event for an object type whose key structure is different from the key structure of the change document object. In this case, you must first enter a structure for the object type key and a function module in the workflow-relevant settings for change documents. The function module "translates" the key of the change document into the key of the business object. For further information, refer to Maintenance of Workflow-Relevant Settings for Change Documents [Seite 1330].

5. Enter the ID of the event to be created in the field Event. The event must be defined for the specified object type. The name is taken automatically from the object type definition.
6. Select whether the event is to be created upon change, create or delete actions.

Changes to fields in various tables are generally logged with a change document object. Only one of these tables is the main table in the sense that the relevant change document is written with its key. The create, change and delete actions always associated with the main table.

Depending on how the change documents are used in the application, it may be that no change document is written upon create or delete actions.

When an item in a sales and distribution document (structure VBAP) is created, the header data of the document (structure VBAK = main table of the change document object) is changed. Therefore, in this case the change document written logs a change and not a create action.
Maintaining the Assignment Between Change Document and Event Without Field Restrictions

For further information on how you can react to the creation or deletion of non-main tables, refer to Maintaining the Assignment Between Change Document and Event With Field Restrictions [Seite 1329].

7. Save your entries and exit view maintenance.

Result
If you have catered for the creation of an event in the manner described above, you can use this event as a triggering event [Extern] of a task or workflow in the normal way.

If other conditions are to be checked for the linkage between the triggering event and the task to be started, use a check function module [Extern]. You must enter this manually in the type linkage table [Extern].

Event container of created event
In addition to other elements, the event container of the created event contains:

- The reference to the changed object in the element _Evt_Object.
- The reference to the user logged-on in the element _Evt_Creator.
Prerequisites
To restrict the event creation with regard to whether the change refers to specific fields or fulfills specific conditions, proceed initially as described under Maintaining the Assignment Between Change Document and Event Without Field Restrictions [Seite 1327].

Procedure
1. Select a change document-event assignment and call the dependent view Complex field restrictions. This displays the screen for changing a condition. You process the condition in the upper half of the screen. An existing condition may already be displayed there.
2. Determine the condition that is to trigger the event. Refer to the documentation on the condition editor [Seite 1012].
3. Exit the condition editor and choose Save.

Via the dependent view Field restrictions, you can determine a more simple field restriction (comparison of an "old" value with a "new" value). From there you can also branch to the condition editor via the function expert mode.
Maintenance of Workflow-Relevant Settings for Change Documents

Use
Certain settings must be made for each change document object before it can be used for event creation. This information is required, for example, for correct assignment between change document object and object type. For all change document objects supplied by SAP, the settings required are made.

- If you want to use the functionality of existing change documents, the entries displayed provide additional information on the existing change document objects. Do not change these entries!

- If you create your own change document object and want to prepare it for the purposes of flexible event creation, you must also make the relevant specifications in the view maintenance described below.

Features
The following information is stored in the table for each change document object.

Main table in change document
A change document usually refers to several tables. Changes to one of these tables (the "main table" of the change document object) determine the character of the change. Whether the change document is written when an entry is created, changed or deleted is hence derived from the changes to the main table in the change document.

Change document key with structure
The change document is written with the (key) fields of the table or structure specified in the field Change document key with structure. Using this, the system can check whether the key fields of the event-triggering object can be derived consistently (= domain same) from the key fields with which the change document is written. It is also used to set up the F4 input help for entering a business object type when creating events.

Indicator Action: Create, change, delete
Change documents are often written when table entries are changed, and less often when they are created or deleted. To construct a full reference, the actions for which a change document is written are specified. The creation of entries in a table (for example order items), which is not the main table, may lead to a change in the main table (for example order header) and the event only being created if a change action is executed.

Indicator Action: Create instance
Events created from the relevant change document when an object is created can only be used as terminating events if this indicator is set.

Structure
Structure (or table) whose fields define the key composition of an object. This structure must be specified if the key of the event-triggering object cannot be derived from the key of the change document object, but is instead determined with the function module (see below).
Function module
Function module called by the system with the import parameters key of change document and object type, and which returns the key of the event-triggering object. This function module is provided by the application for each change document object, if applicable. Its interface corresponds to the function module SWE_TEMPLATE_OBJECT_KEY_SET, which can be used as a template.

Activities
To maintain the workflow-relevant settings of change documents, choose Tools → Business Workflow → Development → Definition tools → Events → Event creation → Change documents → Define workflow properties.
Creation of Additional Event Parameters With Event Creation

Use
The system adds the old and new values of all changed fields to the event container of the event created. These values can then be read and evaluated in the workflow started by the event in order to influence the process flow of this workflow.

Prerequisites

Writing values into the event container
In order for the system to write the old and new values of a changed field into the event container, this field must be defined as a database field attribute [Extern] for the object type in the Business Object Builder.

Using values in the binding
To enable you to define a binding, with which this information is brought from the event container to the workflow container, the relevant fields must be defined as multiline event parameters [Extern] for the event in the Business Object Builder.

Features
After a change document has been written and the system has recognized from the assignment table that an event of a particular object type is to be created for it, the system checks whether the changed fields are defined as database field attributes of this object type. If this is the case, the changed value and the new value are written under the name of the attribute as a two-line element into the event container. (New value with index 0001, old value with index 0002.)

The values are put into the container as described even if they are not defined as event parameters. The definition as event parameters is required so that you can access it in a binding definition.

Activities
If you want to use the function of additional event parameters with event creation, you must proceed as follows:

- Define an appropriate database field attribute for each relevant field, a change in which can create an event. Ensure that this attribute refers to a table field for which a change is actually logged with the change document. Adopt the name proposed by the system.
  
  For further information, refer to Definition of Attributes for an Object Type [Seite 1123].

- In the same manner, define a multiline event parameter for the event to be created upon the change for each of the fields above. Adopt the name proposed by the system.
  
  For further information, refer to Definition of Events for an Object Type [Seite 1129].

- Define a binding between the event container and the container of the event receiver (usually the workflow container of the workflow to be started).
Creation of Additional Event Parameters With Event Creation

To test what the content of the event container is upon "arrival" at the event receiver, you can send the event container to yourself as a mail. To do this, you must enter a function module as event receiver, which converts the container content into a mail and sends it.

- Add a new entry to the type linkage table for the event receiver linkage.
- Enter the object type and the event that you assigned to the change document object.
- Enter your user name as receiver type.
- Enter **SWE_EVENT_MAIL** as receiver function module.
- Set the indicator **type linkage activated**.

For further information, refer to [Evaluation and Maintenance of the Type Linkage Table](#) [Seite 1358].

---

You have created a subtype for the object type **BUS1007 (customer)**. You have added the following components to this subtype:

- **Attribute PostalCode** with database field reference **KNA1-PSTLZ**
- **Attribute City** with database field reference **KNA1-ORT01**
- **Attribute District** with database field reference **KNA1-ORT02**
- **Event** changed with event parameters **PostalCode**, **City** and **District** (database field references as with attributes).

The container of the event created when a customer's address changes (relocation from Trier to Cologne) has not only the standard system elements but also the following multiline elements, each of which has a two-line value.

<table>
<thead>
<tr>
<th>Element</th>
<th>Value</th>
<th>Index</th>
</tr>
</thead>
<tbody>
<tr>
<td>PostalCode</td>
<td>50999</td>
<td>0001</td>
</tr>
<tr>
<td></td>
<td>54294</td>
<td>0002</td>
</tr>
<tr>
<td>City</td>
<td>Cologne</td>
<td>0001</td>
</tr>
<tr>
<td></td>
<td>Trier</td>
<td>0002</td>
</tr>
<tr>
<td>District</td>
<td>Rodenkirchen</td>
<td>0001</td>
</tr>
<tr>
<td></td>
<td>Pallien</td>
<td>0002</td>
</tr>
</tbody>
</table>
Creation of Events Upon Status Changes

Use
Event creation can be connected to changes in system status or user status without modifying an existing application.

Integration
General status management makes it possible to document the current processing state of an application object using statuses.
A status is an indicator that can have only the states set (active) or not set (not active). Statuses that can have more than these two states are not supported.
These statuses can be set by the system (system status) and by the user (user status).

- System statuses are indicators that are only set by the system. Their purpose is to document the current state of an object from the system viewpoint.
- User statuses are indicators defined by the user, which document the processing state of the object from the viewpoint of the user.

User statuses are only defined within a status profile. A status profile should be regarded as a group of user statuses.

Activities
If the status change of an object is logged as a change of a user status or system status, you can "report" this in a simple manner using an event. Certain entries must be made in a control table:

- To create an assignment between a status object type and an application object type/event.
- To initiate event creation when the status is either set or reset.
Maintaining the Assignment Between Status Change and Event

Prerequisites
The event that you want to create is defined as a component of an object type. The key fields of this object type must be compatible (= domain same) with the key fields of the status object type whose status change is to cause the creation of an event. The relevant information is located in the system table TBO00, which is maintained and supplied by SAP. This function is not available in a separated Basis system.

Procedure
Make the following settings to create an event if a status change occurs:

2. In the next dialog box, decide whether you want to implement the event creation by connecting to a system status (system settings) or a user status (customer settings). The basic procedure described below is not affected by this decision.
   The screen View: Change "Events for ...Status": Overview is displayed. If you are not in change mode, select Display → Change.
4. Enter a status object type in the field StatusOT.
5. Only for event creation via user status (customer settings): Enter the status profile for the user status in the field Profile.
6. In the field BusinessOT, enter the object type whose status change is to be indicated by the event created. Use the F4 input help.
7. Enter the ID of the event to be created in the field Event. The event must be defined for the specified object type. The name is taken automatically from the object type definition.

   The event entered is created upon each status change for this status object type.
8. If the event is not to be initiated for every status change but only for activation or deactivation of certain statuses, the view must be maintained for this status event.
   In the view "Status restrictions", you can define the statuses of the status object type, which must be set for the event to be triggered. If you select the field Inactive, the event is triggered when the relevant status is reset.

   If you accessed maintenance of the assignment between status change and event via customer settings, you can enter either a system status or a user status (for the status profile in the first view) in each input line for the status restrictions.
   You can also specify several status restrictions. In this way, you get an "AND" operation for the various restrictions.
Maintaining the Assignment Between Status Change and Event

You can get "OR" operations by defining several events for a status object type.

**Result**

If you have catered for the creation of an event in the manner described above, you can use this event as a triggering event [Extern] of a workflow or a task in the normal way.

**Event container of created event**

In addition to other elements, the event container of the created event contains:

- The reference to the changed object in the element _Evt_Object.
- The reference to the user logged-on in the element _Evt_Creator.
Creation of Events via Message Control

Use
One of the options for customer-specific creation of events is fully integrated in Message Control. For this, you must create a separate output type and declare 9 (workflow event) as transmission medium.

Prerequisites
This function is not available in a separated Basis system.

Features
Message Control is used for the exchange of information between different partners.

Delivery system
EVEN ("trigger event") has been created as output type.
The program RVNSWE01 with the form routine CREATE_EVENT is the standard processing program for sending the message.
The event ID and the object type that characterize the event created are assigned to a message application in the table NAST.
For further information, refer to the documentation Message Control [Extern].
Creation of Events Upon LIS Exceptions

**Use**
The processing following an exceptional situation defined as an *exception* recognized by the Early Warning System can be undertaken by a workflow. This *exception* must trigger an event entered as a triggering event of a workflow.

**Integration**
The Early Warning System is based on the key figures of the Logistics Information System (LIS). The Early Warning System can be used in decisions to make selections and check weak points within Logistics. The Early Warning System enables you to search for exceptional situations, thus helping you to recognize and rectify potential problems at an early stage.

You define the exceptional situations as *exceptions* within the Early Warning System. An *exception* consists of the specification of characteristics and/or characteristic values (such as vendor, material) and conditions. Conditions can be created as threshold values (such as materials/vendors with an order value greater than $5000), as trends (such as a positive trend in the order value) or as planned/actual comparisons.

For further information about the LIS and the Early Warning System, refer to the documentation [Early Warning System: Overview](mailto:Early%20Warning%20System%3A%20Overview%20%5BExtern%5D).

**Prerequisites**
This function is not available in a separated Basis system.

**Features**
The assignment between an *exception* and a business object type/event must be maintained in a system table. Then the system creates the event when an *exception* is recognized.

In connection with the event creation, the system fills the event container with the following information about the exception at runtime:

- The characteristic values when the *exception* was triggered
- The key figures when the *exception* was triggered

To enable you to define a binding, with which this information is brought from the event container to the workflow container, you must define the characteristics and key figures as event parameters for the event in the *Business Object Builder*.

**Activities**
You make entries in a *control table* to define which *LIS exception* is assigned to which object type/event. You may also specify a function module with which operations can be performed on the event container. The system calls this function module immediately before scanning the *type linkage table* for possible receivers.

For further information, refer to [Wizards for Event Creation](mailto:Wizards%20for%20Event%20Creation%20%5BSeite%201324%5D).
Creation of Events via Business Transaction Events

Use

G/L accounting and accounts receivable/payable accounting provide publish and subscribe interfaces, which inform other application components or external interested parties (partners, customers) that particular events (such as document entered) have taken place in financial accounting. The data this creates can be used by the interested parties for their own subsequent processing. But no data can be returned to G/L accounting or accounts receivable/payable accounting.

SAP Business Workflow uses this interface to create events of object types defined in the Business Object Repository from its events. These events can be used as triggering events of a workflow, for example.

Features

When a Business Transaction Event occurs the system calls a function module which creates the BOR event. To do this, this function module "converts" the data from the Business Transaction Event into the format suitable for the event and creates the appropriate event for the Business Transaction Event. The function module hence has a parameter interface and implementation which are appropriate for the Business Transaction Event.

For every Business Transaction Event that can be linked to an event, SAP provides a suitable function module with appropriate interface and implementation. The following relationships are currently possible:

<table>
<thead>
<tr>
<th>Business Transaction Event</th>
<th>Event</th>
<th>Event Description</th>
<th>Status</th>
</tr>
</thead>
<tbody>
<tr>
<td>00001030 POST DOCUMENT: Update standard data</td>
<td>BKPF Accounting document</td>
<td>CREATED</td>
<td></td>
</tr>
<tr>
<td>00001040 REVERSE CLEARING: After standard update</td>
<td>BKPF Accounting document</td>
<td>CLEARINGREVERSED</td>
<td></td>
</tr>
<tr>
<td>00001050 POST DOCUMENT: FI/CO interface</td>
<td>BKPF Accounting document</td>
<td>CREATED</td>
<td></td>
</tr>
<tr>
<td>00001110 CHANGE DOCUMENT: Save standard data</td>
<td>BKPF Accounting document</td>
<td>CHANGED</td>
<td></td>
</tr>
<tr>
<td>00001320 CUSTOMER MASTER DATA: Save</td>
<td>BUS3007 Customer account</td>
<td>CREATED</td>
<td></td>
</tr>
<tr>
<td>00001420 VENDOR MASTER DATA: Save</td>
<td>BUS3008 Vendor account</td>
<td>CREATED</td>
<td></td>
</tr>
<tr>
<td>00001520 CREDIT MANAGEMENT: Save</td>
<td>BUS1010 Customer credit account</td>
<td>CREATED</td>
<td></td>
</tr>
</tbody>
</table>

Activities

The Business Transaction Events and the events are already defined. You may create the linkage between them. You can use a wizard for event creation [Seite 1324].

⚠️

Finally, you must set the status of the event whose creation you have just configured to released.

To do this, call the Business Object Builder [Seite 1101] and change the release status of the event for the relevant object type.
Creation of Events by Calling a Function Module

Use
You must ensure that the event is triggered when the status change occurs. It may be necessary to trigger the event using a function module in your program.

Prerequisites
Note that events say something about object status changes that have actually occurred. Therefore ensure that the event is not created until the relevant status change has taken place. For this, the function module for creating an event should be called in the same logical unit of work (LUW) as the one in which the status change is made.

Features
You can create any event from any application or system program by calling the relevant function module:
This function module is called SWE_EVENT_CREATE.
For the following special cases, there are other function modules, which actually use the functionality of the above function module internally:
- SWE_EVENT_CREATE_IN_UPD_TASK
  This function module makes it possible to create events in an update task. In contrast to the function module SWE_EVENT_CREATE, it can be called with the addition of IN UPDATE TASK.
- SWE_EVENT_CREATE_FOR_UPD_TASK
  The event is created in the update task. (The function module is not called with the addition of IN UPDATE TASK.)
  In the case of methods for creating an object, the requester used for linking instance-related events is transported into the update session so that an event can be used as a terminating event in the workflow as well. (The particular problem here is that when the instance linkage for the terminating event is entered, the object instance is still unknown.)

When the function module SWE_EVENT_CREATE is called, the following operations are performed synchronously:
- Find requester
  If an event is created by an application that is executed as an asynchronous object method within a workflow, the work item that called this method can be established via internal queries.
- Determine all supertypes of the triggering object type
  The type linkage is not only taken into account for the triggering object type itself, but also for all supertypes of this object type.

  A type linkage is entered for an event of the object type person.
  If this event is created by the subtype applicant, the type linkage of the person should also be evaluated.

The evaluation of the linkage tables taking into account the import parameters of the function module for determining a possible receiver follows.
Creation of Events by Calling a Function Module

It is not possible to state whether the receivers were called successfully.
For further information, refer to Evaluation and Maintenance of the Type Linkage Table [Seite 1358].

Activities

Call the function module SWE_EVENT_CREATE or one of its modified versions mentioned above in a program of your application. The function module has the following interface (selection):

<table>
<thead>
<tr>
<th>Import parameters</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>OBJTYPE</td>
<td>SWETYPECOU-OBJTYPE</td>
</tr>
<tr>
<td>OBJKEY</td>
<td>SWEINSTCOU-OBJKEY</td>
</tr>
<tr>
<td>EVENT</td>
<td>SWETYPECOU-EVENT</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Export parameters</th>
<th></th>
</tr>
</thead>
</table>
| EVENT_ID | SWEDUMEVID-EVTID | The event number has a value other than zero if the event manager could establish one or more receiver function modules.
It is not possible to state whether the receivers were called successfully.
If no receiver could be established, zero is returned as the event number. |

<table>
<thead>
<tr>
<th>Table parameters</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>EVENT_CONTAINER</td>
<td>SWCONT</td>
</tr>
</tbody>
</table>

Since the asynchronous RFC for calling the receiver function module is not triggered until after the next COMMIT WORK, you must initiate the command COMMIT WORK in your application after the function module for creating an event is called in order for the events to actually be created.

The database commit performed automatically with a screen change does not trigger the asynchronous RFC.
**Event parameters**

If the event that you want to create has other event parameters defined in addition to the event parameters defined by the system, you must carry out the following before the function module is called:

1. Declare the event container
2. Initialize the event container
3. Assign values to the event container.

Macro instructions are available for carrying out these steps. For further information, refer to [Macro Instructions for Processing a Container Instance][1].

The container is then passed as a table parameter to the function module **SWE_EVENT_CREATE**. The name you give to the event container in your application is up to you.

---

[1]: Seite 1133
Definition of Event Container

Use
The event container contains the event parameters [Extern] as container elements. If you define triggering [Extern] events or terminating [Extern] events for a task, you can specify the binding from the event container to the task container. If you define triggering events for a workflow, you can specify the binding from the event container to the workflow container.

Features
The event container contains workflow system elements [Extern] as standard. You can add more container elements to the event container within object type definition in the Business Object Builder. This is not usually necessary, however.
Event Receiver

Definition
Program in which an event is evaluated.
From a technical point of view, the event receiver is started by an asynchronous RFC call of the receiver function module in the event receiver linkage table.

Use
Any application that wants to react to an event must provide a receiver function module and make an appropriate entry in the linkage table.

If you use events in the environment of SAP Business Workflow, special components in the workflow system are the receivers of the event.
You do not then need to concern yourself with the programming of the receiver function module and the entry in the linkage table.

Integration
The receiver function module is selected and called by the event manager once it has evaluated the type linkage table.
**Receiver Function Module**

**Definition**
Function module with defined interface for receiving events. The event manager calling the receiver function module initiates the event receiver’s reaction to the event.

**Use**
The container passed in the interface of the function module is read with macro instructions provided. If object references were passed in the container, they must first be converted into runtime handles. For further information and a full list of all macro instructions, refer to:
- [Macro Instructions for Processing a Container Instance in a Program](#)[Seite 1133].
- [Macro Instructions for Accessing Objects, Attributes, and Methods](#)[Seite 1146]

The workflow system uses the following function modules as standard:

<table>
<thead>
<tr>
<th>Function module</th>
<th>Use</th>
</tr>
</thead>
<tbody>
<tr>
<td>SWW WI CREATE VIA EVENT</td>
<td>Is entered for triggering events.</td>
</tr>
<tr>
<td>SWW WI COMP EVENT RECEIVE</td>
<td>Is entered for terminating events.</td>
</tr>
<tr>
<td>SWW EI EVENT RECEIVE</td>
<td>Is entered for wait steps.</td>
</tr>
</tbody>
</table>

**Structure**
The interface of the receiver function module is described with the following parameters:

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>OBJTYPE OBJTYPE</td>
<td>Type of the triggering object.</td>
</tr>
<tr>
<td>OBJKEY OBJKEY</td>
<td>Concatenated, object type-specific key of the triggering object.</td>
</tr>
<tr>
<td>EVENT EVENT</td>
<td>ID of the event.</td>
</tr>
<tr>
<td>RECTYPE RECTYPE</td>
<td>Receiver type.</td>
</tr>
<tr>
<td>EVENT_CONTAINER</td>
<td>Persistent event container of the event.</td>
</tr>
</tbody>
</table>

There is a template function module for receiver function modules (`SWE_CD_TEMPLATE_REC_FB`) in the function group `SWE_TEMPLATE`.

**Integration**
The receiver function module must be provided by the potential event receiver. The interface of this function module is standard and predefined. The receiver function module expects the ID of the event, the event container and the name of the receiver type as input parameters. Return parameters are not passed by the receiver function module.
**Receiver Function Module**

*Return parameters* and *exceptions* cannot be passed by the receiver function module. In addition, it cannot initiate *exceptions* (error messages and warnings) which lead to the function module being aborted.

The receiver function module must be flagged as *RFC-enabled*. 
Check Function Module

Use
If the check function module [Extern] terminates with an exception, the receiver function module is not called and the linkage between the event and the receiver therefore not established. Using a check function module gives the event receiver the opportunity to decide whether the receiver should actually be called for general or non-specific events before the call actually takes place.

The event Notification created should only actually lead to the receiver being called, if the notification is a service notification.

A check function module exception has no effect on the event creator. The container passed in the interface of the function module is read with macro instructions provided. If object references were passed in the container, they must first be converted into runtime handles.

For further information and a full list of all macro instructions, refer to:
- Macro Instructions for Processing a Container Instance in a Program [Seite 1133]
- Macro Instructions for Accessing Objects, Attributes, and Methods [Seite 1146]

Integration

Creator and Receiver Context

The check function module must be provided by the potential event receiver, if applicable. The check function module has the same interface as the receiver function module. This allows the contents of the event container to be checked.

Features
The interface of the check function module is described with the following parameters:
### Check Function Module

#### Import parameters

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>OBJTYPE</td>
<td>Type of triggering object</td>
</tr>
<tr>
<td>OBJKEY</td>
<td>Concatenated, object type-specific key of the triggering object</td>
</tr>
<tr>
<td>EVENT</td>
<td>ID of event</td>
</tr>
<tr>
<td>RECTYPE</td>
<td>Receiver type.</td>
</tr>
<tr>
<td>EVENT_CONTAINER</td>
<td>Persistent event container of the event.</td>
</tr>
</tbody>
</table>

#### Exception

- If the check function module terminates without an exception, the receiver function module is called by the event manager.
- If the check function module terminates with any exception, the receiver function module is not called.
- Exceptions are not treated in different ways.

There is a template function module for check function modules

(SWE_CD_TEMPLATE_CHECK_FB) in the function group SWE_TEMPLATE.
**Receiver Type**

**Definition**
Additional input parameter for the *receiver function module*, which determines the type of object that is manipulated (in the broadest sense) by the event receiver.

**Use**
The receiver type, like the receiver function module, is entered in the type linkage table by the event receiver.
A value does not have to be entered in the type linkage table for the receiver type. Accordingly, the receiver type is an optional import parameter of the *receiver function module*.

In the workflow environment (triggering events), the receiver type, which is passed to the function module `SWW_WI_CREATE_VIA_EVENT` as a parameter, is the unique ID of the task to be started.

**Integration**
The event receiver can provide a generic *receiver function module* that then branches with the input parameter *receiver type*.
The receiver type, like the receiver function module, is entered in the type linkage table by the event receiver. Specification of a receiver type is *optional*. 
Receiver Instance

Definition
Instance of an event receiver.

Use
The ID of a receiver instance (receiver ID) is entered in the instance linkage table at runtime by the potential event receiver.
When the instance linkage is implemented by the event manager at runtime, the event manager writes the receiver ID to the event container and passes it to the receiver function module declared in the type linkage table for further evaluation.

In the workflow environment (wait steps, terminating events), the number of a workflow is generally used as a receiver instance at runtime.
The work item manager [Extern] makes the entry in the instance linkage table.
Receiver Type Function Module

Definition
Function module for establishing the receiver type.

Use
It is possible that there is to be a receiver type, but that it cannot be established until runtime from the information accompanying the event. It is therefore possible to call a receiver type function module to evaluate the event container and determine the receiver type.

A workflow is to be started in response to an event.

The receiver function module is the generic function module for starting workflows, which expects the ID of the workflow task to be started as an input parameter in the receiver type.

The ID may not be known when the linkage is entered. This may be the case if the ID of the workflow task to be started depends on information from the event-creating application (for example an amount).

The relevant information is located in the event container. The ID of the workflow task is then determined by the receiver type function module evaluating the event container.

The container passed in the interface of the function module is read with macro instructions provided. If object references were passed in the container, they must first be converted into runtime handles.

For further information and a full list of all macro instructions, refer to:
- Macro Instructions for Processing a Container Instance in a Program [Seite 1133].
- Macro Instructions for Accessing Objects, Attributes, and Methods [Seite 1146]

Structure
The interface of the receiver type function module is described with the following parameters:

<table>
<thead>
<tr>
<th>Import parameters</th>
<th>Export parameters</th>
</tr>
</thead>
<tbody>
<tr>
<td>OBJTYPE</td>
<td>OBJTYPE</td>
</tr>
<tr>
<td>OBJKEY</td>
<td>OBJKEY</td>
</tr>
<tr>
<td>EVENT</td>
<td>EVENT</td>
</tr>
<tr>
<td>GENERIC_RECTYPE</td>
<td>GENERIC_RECTYPE</td>
</tr>
</tbody>
</table>
Receiver Type Function Module

<table>
<thead>
<tr>
<th>RECTYPE</th>
<th>Result of the evaluation by the receiver type function module.</th>
</tr>
</thead>
</table>

Table parameters

<table>
<thead>
<tr>
<th>EVENT_CONTAINER</th>
<th>SWCONT</th>
<th>Persistent event container of the event.</th>
</tr>
</thead>
</table>

There is a template function module for receiver type function modules (SWE_CD_TEMPLATE_RECTYPE_FB) in the function group SWE_TEMPLATE.

Integration

The evaluation of the event container with the aim of establishing the receiver type is performed in the receiver type function module.

This function module is called provided it is entered in the type linkage table. It returns the receiver type established. The check and receiver function modules are then called with this receiver type.
Maintenance of Linkage Tables

Use
The linkage tables contain the assignment of events to the event receivers interested in the events. The entries in the linkage tables must be made by the potential receiver of an event. This can be done using function modules provided or a maintenance transaction.

It is only necessary to maintain linkage tables manually if you use events within separate developments. If events are used exclusively within the SAP Business Workflow environment, all necessary entries (including transport) are made by the workflow system.

Integration
A distinction is made between type linkage and instance linkage. They are contained in separate tables.

Features
If there are appropriate entries in the linkage tables when the event is created, the event manager creates the linkage by calling the receiver function module.
If there are no appropriate entries in the linkage tables when the event is created, the event has no effect. Events are not buffered so that they can be produced at a later point in time.

It is, however, possible to temporarily store events for which a linkage is entered in the event queue.

The linkage between the creator and the receiver can be activated and deactivated by making a selection in the table (indicator Type linkage activated) at definition time and even at runtime for a specific situation or customer.

Activities
The maintenance transactions for linkage tables can be accessed by choosing Tools → Business Workflow → Development → Administration → Event manager → ...
Evaluation and Maintenance of Type Linkage Table

Use

The type linkage describes the assignment of a receiver function module [Extern] and a receiver type [Extern] to a particular combination of object type and event. This linkage is entered into the type linkage table. The table must be completed if a response is always to be made to an event of a particular object type, even if the linkage is to be instance-based. The type linkage table is evaluated at runtime by the event manager. The event receiver should make the entry in the type linkage table using a function module provided.

If events are used in SAP Business Workflow (as triggering and terminating events, for wait steps), the entries required in the linkage tables are made by the workflow system.

If you enter an event as a triggering event of a task, for example, the following entries are made automatically:

- Object type and event ID
- Task to be started as receiver type
- Function module SWW_WI_CREATE_VIA_EVENT as receiver function module
- Indicator global set in background

When the linkage is activated within task definition, the indicator enabled is also set in the type linkage table.

Integration

Transport and client copy

Entries in the client-specific type linkage tables are written automatically into a Customizing transport request if the client is configured (in table T000) for changes to be recorded automatically. All entries are then transported including activation indicators.

In the case of client copy, you should note that the event receiver linkages are copied into the target client, but are always deactivated in the target client.

The activation indicator of each individual event receiver linkage is only copied with client copy if explicitly requested (parameter option for copying tables of class A).

Process Flow

The event manager begins the evaluation of the type linkage table when it is notified of the ID of a created event. For event handling to take place, either the event created and its triggering object type or the created event and a supertype of the triggering object type must be entered in the type linkage table.

Only those entries in the type linkage table for which the indicator Event linkage activated is set are included by the event manager in the evaluation.
Type Linkage

An appropriate workflow is always started in response to an event of a particular object type.

The receiver type function module [Extern] for determining the receiver type is called if specified in the type linkage table.

If the event linkage is designated a global event linkage, the following occurs:

- The check function module [Extern] is called if specified in the type linkage table.
- If this function module terminates without an exception (or if no check function module is specified), the receiver function module is called with the transfer parameters event, receiver type and event container.
- The instance linkage table is not evaluated.

If the event linkage is not designated a global event linkage, the instance linkage table [Seite 1354] is evaluated.

If an error occurs within the function module $SWW\_WI\_CREATE\_VIA\_EVENT$ (task missing or has errors, problems with tRFC), a mail is sent to the workflow system administrator.

Features

Structure of the type linkage table
The following fields are available in the type linkage table:

- **Object type**
  
  Type of object for which the event is created.

- **Event**
  
  Event created.
Object type and event must be defined in the Business Object Repository.

- **Receiver type**
- **Receiver function module**
- **Check function module**
- **Receiver type function module**
- **Destination**

  Standard name of a logical RFC destination in which the receiver function module is called. If no destination is specified, the receiver function module is called locally in the destination `WORKFLOW_LOCAL_<Client>`.

  The workflow runtime system only uses events locally and does not complete this field. Do not enter a destination.

  For further information, refer to Destination Types [Extern].

- **Indicator: Global**

  If the event linkage is shown as *global*, the receiver function module specified is called each time the object type/event pair occurs, irrespective of the triggering object instance.

  The instance linkage table is not evaluated for this event.

  The indicator is set automatically by the workflow system:

  - The indicator is set for triggering events.
  - The indicator is not set for terminating events and wait steps.

  This indicator can only be set manually with the following function modules:

  - `SWE_EVENT_REC_TYPE_GLOBAL`
  - `SWE_EVENT_REC_TYPE_NOT_GLOBAL`

- **Indicator: Type linkage activated**

  If you set this indicator, the type linkage between the event creator and the event receiver is activated.

- **Indicator: Enable event queue**

  This indicator is only evaluated if the event queue is active [Seite 1519].

  If the indicator is set, the event receiver is started via the event queue. If the indicator is not set or the event queue is not active, the receiver is started immediately.

- **Behavior if error feedback**

  The system presetting maintained on the tab page Basic data [Seite 1517] in event queue administration can be adopted or revised.

- **Receiver status**
The status of the event receiver affects the starting of the receiver. In general, the receiver linkage must be active to start the receiver. The status of the receiver linkage has the following effects:

- **No errors**
  If the receiver linkage has the status *no errors*, the receivers are started by the system. They can be started directly or via the event queue.

- **Errors**
  If the receiver linkage has the status *errors*, the receiver is not started immediately, but is put into the event queue automatically instead. There the entry is assigned the status *Waiting because of errors*. So the receiver is not started, but there is an opportunity to start the receiver again after the error has been removed. Triggered events do not get lost even with linkages that have errors. The event receivers can be started again using the tab page *Linkages with errors* [Seite 1525].

### Maintenance of type linkage table

You have to process the type linkage table manually if:

- You use event control within your own non-workflow developments.
- You want to analyze the entries made by the workflow system.
- You want to add a check function module at a later date.
- You want to activate an event receiver linkage.

- You want the events of a particular linkage to use the event queue [Seite 1515].

Otherwise, the maintenance of the type linkage table (entering linkages for triggering or terminating events) is carried out by the workflow system.

It is possible to create a full entry in the type linkage table using the following function module: **SWE_EVENT_REC_TYPE_ENTER**.

APIs are also available for processing individual fields in the type linkage table.

<table>
<thead>
<tr>
<th>Name of function module</th>
<th>Explanation</th>
</tr>
</thead>
<tbody>
<tr>
<td>SWE_EVENT_REC_TYPE_CHECK_FB</td>
<td>Changing the check function module for the event receiver</td>
</tr>
<tr>
<td>SWE_EVENT_REC_TYPE_DISABLE</td>
<td>Deactivating a type linkage</td>
</tr>
<tr>
<td>SWE_EVENT_REC_TYPE_ENABLE</td>
<td>Activating a type linkage</td>
</tr>
<tr>
<td>SWE_EVENT_REC_TYPE_FB</td>
<td>Changing the receiver function module</td>
</tr>
<tr>
<td>SWE_EVENT_REC_TYPE_GET_FB</td>
<td>Changing the function module for the type determination of the event receiver</td>
</tr>
<tr>
<td>SWE_EVENT_REC_TYPE_GLOBAL</td>
<td>Setting a global event linkage</td>
</tr>
<tr>
<td>SWE_EVENT_REC_TYPE_NOT_GLOBAL</td>
<td>Deleting a global event linkage</td>
</tr>
<tr>
<td>SWE_EVENT_REC_TYPE_READ</td>
<td>Reading a type linkage</td>
</tr>
</tbody>
</table>

Use the following function module to delete an entry from the type linkage table: **SWE_EVENT_REC_TYPE_DELETE**.

If you make entries in the type linkage table, make sure you delete them again when a type linkage is no longer expected. Ensure that there are no instance linkages for this type linkage.
Activities

To maintain the type linkage table, choose Tools → Business Workflow → Development → Utilities → Events → Type linkages.

The overview displays the existing type linkages with details of object type, event and receiver type.

Choose Edit → New entries to add new entries to the type linkage table.

Choose Goto → Details to revise entries.
Evaluation and Maintenance of Instance Linkage Table

Use
The instance linkage describes the assignment of a receiver instance to a particular combination of object type, object key and event. This linkage is entered into the instance linkage table by the receiver instance itself using a function module provided. The entries in the instance linkage table are always made at event receiver runtime.

If events are used in SAP Business Workflow (as triggering and terminating events, for wait steps), the entries required in the linkage tables are made by the workflow system. An entry is made in the instance linkage table by the workflow system if work items wait for a terminating event. This can apply to the following work item types:

- Dialog work items (type W) and background work items (type B) if an asynchronous object method is executed.
- Wait step work items (type E)
- Work queue work items (type A)

Integration
The instance linkage enhances the type linkage such that the reaction to the event is only produced if the event is derived from a particular object. This object is specified in the instance linkage table using its object key. The unique ID (receiver ID) of a receiver instance can also be specified in the instance linkage table. This receiver ID is then written to the event container by the event manager and forwarded to the event receiver when the receiver function module is called.
Instance Linkage

The reaction to an event is made dependent on a particular object, as an instance of an object type

**Type linkage table (global linkage)**

- **Object type:** FI doc
- **Event:** created

**Instance linkage table**

- **Object type:** FI doc
- **Object key:** 12345
- **Event:** created

The instance linkage table is only evaluated by the event manager following the evaluation of the type linkage table if the linkage is **not** designated a global event linkage in the type linkage table. The process flow is as follows:

1. The event manager checks whether the **object type**, **object key** and **event** in the instance linkage table are in accordance with the event parameters of the created event.
   
   If this is the case, the ID of the **receiver instance** (receiver ID) is entered into the event container from the instance linkage table.

2. The **check function module** is called if entered in the type linkage table.

3. If this function module terminates without an exception (or if no check function module is specified), the **receiver function module** in the type linkage table is called with the transfer parameters **event**, **receiver type** and **event container**.

4. If it was possible to establish the instance linkage, the entry is deleted from the instance linkage table. The entry in the type linkage table remains.

The specifications made in the type linkage table at definition time are used again for the actual assignment of the event to the receiver. This double construction is necessary since the generic selection of the receiver function module must be defined before runtime, but the instance values are not yet known at definition time.

The linkage logic does not allow an actual receiver instance to wait for an event whose triggering object is not yet known. The key of the triggering object must also always be known for an instance linkage.

This problem occurs when an object is created in a work item by an asynchronous method and this is reported to the workflow system by an event. The work item, as an actual receiver instance, is to wait for an event whose triggering object is not yet known when the work item is entered in the instance linkage table.
Evaluation and Maintenance of Instance Linkage Table

For this event, there is the workflow requester via which a relationship can be created between the unknown object and the waiting work item.

Features

Structure of the instance linkage table

The following fields are available in the instance linkage table:

- **Object type**
  Type of the object that creates the event.

- **Event**
  Event created.

  Object type and event must be defined in the Business Object Repository.

- **Receiver type [Extern]**
- **Object key**
  Concatenated key of the object that creates the event.

- **Receiver instance**
  ID or instance of a receiver type.

The instance linkages (with associated type linkages) required for the wait steps and the terminating events of an activity in the workflow definition are entered automatically. Do not change these entries!

The following are entered in the type linkage table:

- **Object type and event ID**
- **WORKITEM** (for terminating events) or **EVENTITEM** (for wait steps) as **receiver type**
- **Function module** `SWW_WI_COMP_EVENT_RECEIVE` (for terminating events) or `SWW_EI_EVENT_RECEIVE` (for wait steps) as **receiver function module**
- **Indicator global** is not set
- **Indicator enabled** is set

The following are entered in the instance linkage table:

- **Object type and event ID**
- **Key** of the object processed in the task
- **Number of the work item to be terminated or waiting** as ID of receiver instance

If it was possible to establish the instance linkage, the entry is deleted from the instance linkage table. The entry in the type linkage table remains.

Evaluation of the instance linkage table

For the purpose of troubleshooting and error analysis, it may be useful to display the instance linkages [Extern] entered. To do this, you can display a section from the instance linkage table according to certain selection criteria.
Maintenance of the instance linkage table

You maintain the instance linkage table if you use event control within your own developments outside workflow. The instance linkage table can be maintained using the following function modules:

- **SWE_EVENT_REC_INST_ENTER**
  
  You can insert entries into the instance linkage table using the parameters of this function module.

- **SWE_EVENT_REC_INST_DELETE**
  
  You can delete an entry from the instance linkage table using this function module.

- **SWE_EVENT_REC_INST_DELETE_ID**
  
  You can delete all entries for a receiver from the instance linkage table using this function module.

If you make entries in the instance linkage table, ensure you delete them again when an instance linkage is no longer expected. You should also delete the associated entries in the type linkage table. Ensure that there are no (more) instance linkages for this type linkage.

Activities

To maintain the type linkage table, choose Tools → Business Workflow → Development → Utilities → Events → Instance linkages.
Event Simulation

Use
The creation of an event [Extern] is simulated and the result specifies which tasks and workflows would be started as receivers of this event.

Prerequisites

Authorization
You require appropriate authorization for the functions shown. This is the authorization (authorization object S_PROGRAM) for executing a program for authorization group SWU_DIAG.

Features
The result of the test also shows whether errors were diagnosed in connection with the simulated start of the task, which may result in the task or workflow not being started.
The respective error causes are designated with a letter. The legend can be consulted for a short explanation of the error causes. The most common error causes are:

- There is no workflow definition for the workflow to be started (error cause a).
- The workflow is not activated (error causes b, c).
- The linkage between the triggering event and the task or workflow has not been activated (error cause e).
- The binding definition from the event container to the workflow or task container has errors (error causes h, i). The task or workflow may nevertheless be started but will probably not be processed as expected.

To remove the error in all of the above cases, position the cursor on an entry and choose Goto → Task.
This takes you to the definition of the workflow or task in question, where you can take appropriate measures.

You activate workflows in the Workflow Builder.

- The linkage between the event and the task/workflow is entered in the linkage table but the task does not exist (error cause g).
  This error occurs if you delete a task/workflow defined with triggering events. You must then use transaction SWE2 to process the type linkage table manually and delete the relevant entry.
- The linkage between the triggering event and the task/workflow to be triggered has been created for the task/workflow, but has not been entered in the linkage table (error cause d).
  This error indicates an inconsistency between two tables. Call the task/workflow (Goto → Task) and save it again.
- The actual execution of the task to be started still depends on a check function module [Extern] being run without errors. It therefore cannot be finally assessed at this point (error cause f).
- Calls with errors or non-executable calls to tasks/workflows can, for technical reasons, also affect other calls, which may themselves be successful (error causes j, k).
Activities

Executing simulations

Choose Tools → Business Workflow → Development → Utilities → Event → Simulate event to simulate an event.

1. Describe the event to be triggered with object type and event ID.

2. Choose to simulate the event.

You can also simulate an event directly from a step of the type event creator.
Creation of Events for Test Purposes

Use
You can use this function to create an event for test purposes. This can be used, for example, to start a task or a workflow via an event, and hence check the definition and functionality of triggering events.

⚠️ This function is purely a test function that you do not use in normal operation. The object status change that is normally reported by the event did not occur.

Activities
To create an event for test purposes, choose Tools → Business Workflow → Development → Utilities → Events → Create event.

1. Describe the event to be triggered with object type and event ID.
2. Enter an object key so that the event is generated for a unique object.
3. Decide how the receiver function module is to be called:
   - Trigger receiver FM with delay
     The system calls the receiver function module as a tRFC with a delay of 10 minutes. In the meantime, you can call the tRFC log under your name (Environment → RFC queue) in order to call the receiver function module in the debugger from there. The call is then made in a separate logical context.
   - Trigger receiver FM synchronously
     The system calls the receiver function module synchronously. You have the opportunity to trace the determination of the receiver (search in linkage table) and the call to the receiver function module in the debugger, and to branch to the receiver called.

   Activate the debugging mode via the OK code /H before triggering the event.

   The receiver is called in the event manager context.

4. Select Create event.

If the expected reaction to the event created does not occur, check whether the event is entered as a triggering event of a task and whether the linkage is activated.
Event Trace

Use
All events created correctly are logged in the event trace irrespective of whether potential receivers exist. If a receiver is entered in the event trace, it does not necessarily mean that this receiver was called successfully.

Prerequisites
The event trace is only written if logging has been activated. To trace any events which may not have been created, you must first activate the event trace and then create the event again.

Features
The following data is logged in the event trace:

Event data
- Triggering object type, triggering object
- Event ID
- Triggering program
- Trigger date and time

Receiver or linkage data
- Receiver function module
- Receiver type
- Receiver instance
- Linkage status

Activating/deactivating the event trace
As well as for activating or deactivating the event trace, you can also use this function for specifying selection criteria so that only certain events are logged.

Displaying the event trace
The event trace display can also be restricted using a selection screen. In addition to the criteria for event data, receiver data and event receiver linkage data, the display can also be restricted to linkages with errors.

Deleting the event trace
The deletion of events can also be controlled using selection criteria. The function does not therefore always delete the entire event trace automatically. If you mark the field Only display list, the events you select are not deleted, but displayed in a list.

Activities
- To activate/deactivate the event trace, choose:
Event Trace

Tools → Business Workflow → Development → Administration → Event manager →
Event trace → Switch event trace on/off

- Select the function Selective tracing on the dialog box that follows to choose selection criteria.

- To display the event trace, choose:
  Tools → Business Workflow → Development → Administration → Event manager →
  Event trace → Display event trace

- To delete the event trace, choose:
  Tools → Business Workflow → Development → Administration → Event manager →
  Event trace → Delete event trace
Role Resolution

Use
Role resolution enables you to restrict the number of possible agents for a work item. Role resolution determines which responsible agents have a property described by a role [Seite 1277]. This improves the ability of SAP Business Workflow [Seite 1711] to get the right task to the right person at the right time.

Integration
The tools used to define roles [Seite 1281] are part of the Organizational Management component. At runtime, the SAP Business Workflow component uses the roles defined here in conjunction with the organizational plan for role resolution to determine agents for work items. (Decision as to which tasks must be assigned to which agents)

Prerequisites
To facilitate role resolution, roles must be defined [Seite 1281].

Features
The system performs role resolution. It is first performed at runtime depending on and using information from the process currently running.

The example refers to the role "orders administrator for customer <customer> as of order amount <order amount>".

If customer "Miller Ltd." and order amount "$34,569.34" are determined for a specific order, the agents are determined at workflow runtime who are the "orders administrator for customer Miller Ltd. as of order amount $34,569.34".

The principle of role resolution is always the same:

- The contents of the role container are read.
- The "regulations" or "rules" resulting from the role type are applied to this data.
- The agents so determined are returned as the role resolution result to an internal table with the structure SWHACTOR. This table contains the agents as Organizational Management objects (user, person, position, job, organizational unit) in the required "mix".

The type of role determines how role resolution is performed exactly. You can determine the type of role when defining a role.
Role

Definition
Object used by the *SAP Business Workflow* component to determine possible agents for a work item.

Use
You use roles to specify an agent (or agents) for a task if the set of possible agents is too large, or not specific enough. By assigning work items to organizationally suitable employees, responsibilities and authorizations are managed efficiently, and bottlenecks are avoided.

You want to forward Mr. Smith's notification of absence to his head of department. All heads of department at your enterprise are possible agents for a notification of absence. However, you do not want every head of department to receive Mr. Smith's notification of absence. At runtime, the *role used to determine a manager* [Seite 1311] enables you to evaluate assignments (relationships) within an organizational plan. The system uses relationships to determine that Ms. Miller is Mr. Smith's head of department. The task is forwarded to Ms. Miller.

Further examples:
- [Role to Determine Design Office](Seite 1304)
- [Role to Determine Organizational Unit of a User](Seite 1316)

The agent for a role does not have to be a user. All of the objects in the *Organizational Management* component can be agents for a role.

The workflow uses the values in the role container to select a subset of possible agents. Role resolution, which is performed at runtime to determine the agent for a workflow step, is therefore an intelligent, efficient, and flexible tool.

Roles as Default Roles for Defining Single-Step Tasks
When defining *single-step tasks* [Seite 1175], you can specify particular agents or recipients by their role in the following instances:

- Agent for task
- Recipient for completion
- Recipient for missed latest end
- Recipient for missed start
- Recipient for missed end

In this context, reference is made to default roles for a task. Specifying default roles for a single-step task is always optional. If default roles are specified, you may need to define binding from the task container to the role container.

Resolution is performed for default roles before the single-step task is executed. (If the single-step task is used as a step in a workflow definition, resolution is only performed for default roles if the workflow definition contains no other information with regard to responsibilities or recipients.)

As a general rule, a single-step task can only be executed by its possible agents (or a subset thereof) when it is processed.
Specifying a role restricts these agents to those you have selected. This method cannot be used to authorize new agents to execute a customer task/standard task.

**Roles in Workflow Definition**

When the following steps of a Workflow Definition are defined, the agents responsible and the recipients can be specified by their role:

- Activities
- Wait steps
- User decisions

These specifications only have local validity for the respective workflow definition, and they are optional. (Specifying a role is just one of several methods that can be used to specify the agents responsible and the recipients. It is also possible to specify responsibility by using a suitable organizational object (job, position, organizational unit) or by using an expression with reference to the workflow container.)

**Structure**

There are various ways of defining roles. You can use the following:

- **Function modules**
  
  You use a function module to define standard roles if the agent for a task must be found according to extremely complex selection criteria. If you define roles using function modules, the system finds agents by executing the function. How data is obtained varies from function to function. You can use predefined functions, or create your own functions.

- **Organizational data**
  
  You use organizational data to define standard roles if your business processes are managed on the basis of your organization model. If you define roles using organizational data, role resolution traces the possible agents for a task by using the relationships between the task, the objects in Organizational Management, and the SAP organizational objects.

- **Responsibilities**
  
  You use responsibilities to define standard roles if you need more precise selection criteria to find agents, but do not want to use function modules. You can also use the organization model to find possible agents using jobs, positions, etc.

All three methods offer certain advantages. However, it is preferable to use responsibilities because you do not require ABAP coding, and can easily display and change agent assignments. You can also use the organization model to find possible agents using jobs, positions, etc.

Each role has a role container that includes the values on which role resolution is based. Roles are always defined across clients, and they are always connected to the transport system as cross-client transport objects. At this time, the definition of client-specific roles is not supported.

When saved, each role is assigned an 8-digit number by the system that is preceded by AC, which is used for identification purposes.
Role Container

Use
The role container contains a role’s parameters. Each role has just one role container. At runtime, the role parameters contain the current, context-specific information that forms the basis of role resolution. Therefore, the role parameters constitute “input” for role resolution.
The role parameters are provided with values from the workflow container via binding.

Features
Depending on the role resolution procedure, the role container includes the following information:

<table>
<thead>
<tr>
<th>Role resolution procedure</th>
<th>Details</th>
</tr>
</thead>
<tbody>
<tr>
<td>Based on responsibilities</td>
<td>The role container includes object references or field values (with names as required) that must be provided with values from the calling component via binding.</td>
</tr>
<tr>
<td>Including organizational data</td>
<td>The role container only includes the Org_Object_ID element. This element is defined with a reference to the SAP organizational object whose assigned agents must be determined. The role container is created automatically, and is not visible during role definition.</td>
</tr>
<tr>
<td>By executing a function module</td>
<td>The role container includes object references or field values as required (with names as required) that are read by the function module for role resolution and processed accordingly. Prior to role resolution, the container elements are provided with values from the task container via binding.</td>
</tr>
</tbody>
</table>
| Based on evaluation paths | The role container includes the following elements:  
  - OType, data type reference OBJEC-OTYPE  
    Type of object in Organizational Management according to the first step in the evaluation path  
  - ObjID, data type reference OBJEC-REALO  
    Identification of organizational object.  
  - Org_Agent, data type reference WFSYST-AGENT  
    C14 field as combination of organizational object type and organizational object.  
When defining the role, create either the first two elements, or the last element, as role parameters in the container. |

Activities
You maintain the role container on the Container tab page when maintaining the roles.
Role Definition

Use
You use this function if you want to define more roles in addition to the roles delivered by SAP.

Prerequisites
Before you start defining roles, you must start the workflow process.

1. Decide what you want the workflow to achieve, and break it down to the smallest meaningful units of work. These units of work become single-step tasks in your workflow.
2. Select the objects you require. To do so, create a business object or use one of the existing objects. You should be able to find an object that you can use in the Business Object Repository.
3. Each task accesses an object method. Make sure that the methods you require already exist.
4. Create the tasks, or use the existing tasks. Define the possible agents for the task in the task definition.
5. Group the tasks together in the workflow.
6. Assign agents to the work items in the workflow definition. They must be a subset of the possible agents for the task. If this is not the case, the work item is not executed.

You can do this manually, as a 1:1 assignment, or by using role resolution. For more information, see SAP Business Workflow [Seite 1711].

Features
When defining a role, you determine

- Which information must be available so that role resolution can be performed when the workflow is executed.
  This information constitutes the role parameters. They are defined as elements of the role container.
- The rules or regulations in role resolution that are used to determine the appropriate employees.
  The role resolution procedure is determined by the role type.

The Responsibilities Role Type
When role resolution is performed, an assignment table is evaluated in which Organizational Management objects (jobs, positions, users, organizational units) are assigned to the various characteristics of the role parameters. This assignment table was explicitly created during role definition. For more information, see Define Roles Using Responsibilities [Seite 1283].

The Organizational Data Role Type
When role resolution is performed, the system evaluates SAP organizational objects, such as materials controller, planner group, shipping point, or sales office, which are maintained in the master data of an application object. This type of role resolution requires the use of a separate maintenance transaction independent of role definition to create assignments between the SAP organizational objects and the organizational objects in Organizational Management (jobs, positions, users, organizational units) with which they are related.
For more information, see Define Roles Using Organizational Data [Seite 1289].

**The Function to be Executed Role Type**

When role resolution is performed, a function module is accessed that then facilitates evaluations as required. A table that is maintained in Customizing is evaluated by the function module. The function module must adhere to a given interface, and is specified during role definition.

For more information, see Define Roles Using Function to be Executed [Seite 1305].

**Role Resolution Based on Evaluation Paths**

When role resolution is performed, the system uses information that is available in Organizational Management on the basis of relationships between individual objects that are maintained in an organizational plan.

This information can be used, for example, if you need to determine the head of an organizational unit, or the remaining members of the organizational unit, starting from the initiator of the workflow.

From a technical perspective, this role resolution is very similar to role resolution by executing a function module. The RH_GET_STRUCTURE function module must be used; an evaluation path is also specified.

For more information, see Role Resolution Using Evaluation Paths [Seite 1312].

**Activities**

If you want to use role resolution to restrict the number of possible agents for a work item in a workflow, you must:

1. Choose SAP menu → Tools → Business Workflow → Development → Definition tools → Standard roles to select the tools used to define roles
2. Decide how you want to find agents in the system, that is, using function modules, organizational data, or responsibilities
3. Create a container definition (not for the organizational data role type)
4. Binding is automatically suggested for the workflow. Confirm that the fields in the role container are compatible with the fields in the workflow container.
5. Start the workflow.

At runtime, role resolution provides you with a table containing a set of possible agents. These agents are determined at runtime using the values assigned to the role container elements.
Define Roles Using Responsibilities

Use
A responsibility is an organizational object in which you group criteria together that are required by the workflow at runtime to assign work items to possible agents. This type of criteria definition does not require Customizing settings or ABAP coding. If you use responsibilities for role definition, you enjoy numerous advantages. You can
- Use criteria that you select as required
- Work with value ranges or individual values
- Display (and change) user assignments
- Group several criteria together in a responsibility

Prerequisites
Before defining roles, you must define workflow steps using the procedure described in SAP Business Workflow [Seite 1711]. You then define the tasks that must be executed. Finally, you use roles to assign and find a list of possible agents for the task at runtime.

Procedure
Responsibilities can only be created for existing standard roles.
To define roles using responsibilities, proceed as follows:

1. **Create a container definition [Seite 1285]**
   A container is a generic structure that transfers data at runtime. You must create the container definition first because the responsibility depends on the data that you select in the container. You must define a container element for each criterion that you select for the evaluation at runtime.

2. **Create a responsibility [Seite 1287]**, and define criteria (individual values or value ranges) in the responsibility editor for the container elements.
   At this point, you can also process the values and display the container elements.

3. **Assign users or organizational objects to the responsibility [Seite 1288]**
   If the values in the role container are compatible at runtime with the criteria defined for a responsibility, the organizational objects assigned to this responsibility are identified as possible agents for the work item.

Result
If the values in the role container are compatible with the criteria defined for the responsibilities, SAP Business Workflow identifies the possible agents for a work item. As a result, the right person receives the right work item at the right time. This ensures that business processes at your enterprise run efficiently and on schedule.
Creating a Container Definition

Prerequisites
A container definition informs the system about the data type processed by the workflow. You must decide whether you want to create a container definition that references table fields or object types. When the container definition references object types, you use values in the key fields of objects to restrict the list of possible agents. When the container definition references Data Dictionary fields, you can use specific data such as the amount of an order or the customer name on an invoice to restrict the list of possible agents.

To call the transaction for processing roles, access the SAP menu and choose Tools → Business Workflow → Development → Definition tools → Standard roles → Create.

The Maintain Standard Role screen appears.

Creating a Container Definition with Table Fields

1. Choose Create.
   The Standard Role: Create screen appears.
2. Enter a name for your standard role in the Basic Data box.
3. Select the role type Responsibilities.
4. Choose Container Definition in the Role Definition box.
   The Standard Role: Process Container screen appears.
5. Choose Create.
   The Create Element dialog box appears.
6. If you want to use Data Dictionary fields, choose Yes.
   The Create with Data Dictionary Field Defaults dialog box appears.
7. Specify the table from which you want to select fields and Choose Continue.
   The table appears.
8. Select the fields that you want to appear in the container definition and confirm your entries by choosing Continue.
   The system prompts you to create the texts for the container elements in the Create dialog box.
   The Element <.....> dialog box appears. If you select the required indicator, an error will occur in the workflow if no binding has been defined for the element. If you select the multiple lines indicator, you can specify multiple values for one container element. You can select both indicators. Choose Continue.
10. Choose Back.
    The Standard Role: Create screen appears.

Creating a Container Definition with Object Types

Carry out steps 1 to 5, and then proceed as follows:
6. If you want to use object types, choose No.
   The Standard Role: Process Container screen appears.

7. Enter a name for the element. If you select the required indicator, an error will occur in the workflow if no binding has been defined for the element. If you select the multiple lines indicator, you can specify multiple values for one container element. You can select both indicators.

8. Select the object type you want to reference, and choose Continue.
   The Standard Role: Process Container screen appears.

   The Standard Role: Create screen appears.

Result
You have created a container definition with either Data Dictionary fields or object types as elements. You use these elements to define the criteria for role resolution.
Creating a Responsibility

Prerequisites
Before you create responsibilities, you must Create a Container Definition [Seite 1285]. The values in the role container must either agree with the criteria defined for the responsibility or be in their area. The reason for this is that role resolution compares the two values in order to return a list of possible agents. If you have created a container definition, the Responsibilities button appears on the Add Standard Role screen in the Role definition group box. If you are no longer on the Standard Role Definition screen, then in the SAP Standard Menu, choose Tools → Business Workflow → Development → Definition Tools → Standard Roles → Change and enter the required role. Choose Change.

Creating a Responsibility
On the Standard Role: Create screen:
1. Choose Responsibilities from the Role definition group box.
   The Responsibilities: Change screen appears.
2. Select the superior object (the role you are creating) and choose Create.
   The Create Responsibility screen appears.
3. Enter the name and the validity period of the responsibility and confirm your entries.
   The Change responsibilities for standard role screen appears. This is the responsibility editor.
4. Define values (either single values or value areas) and choose Save.
5. If you do not want to check all criteria for a particular responsibility, leave this line blank.
   The LED display turns yellow, to make you aware that some of the criteria of the standard role will not be checked for this responsibility.

Responsibility Editor
On the Change responsibilities screen, select the responsibility that you wish to edit and choose Change. The Change responsibilities for standard role screen appears. You can:
- Edit the values for each element
- Create a description for a responsibility
- Display a container definition
- Display details on each element

Result
You have created a responsibility with container elements.
Assigning Agents

Prerequisites
You must first create a container definition [Seite 1285] and then create a responsibility [Seite 1287], in which you define criteria for the container values. You then assign an agent to each responsibility.

If you are no longer on the Standard Role Definition screen, then in the SAP Standard Menu, choose Tools → Business Workflow → Development → Definition Tools → Standard Roles → Change and enter the required role. Choose Change and on the Standard Role: Change screen, choose Goto → Responsibilities.

Procedure
On the Responsibilities: Change screen:

1. Select the responsibility you want to assign to an agent and choose Edit → Agent assignment → Create.

   The Selection dialog box appears.

2. Select the organizational object type you want to assign as an agent and choose Continue.

   The system asks you to enter a search term and to create the relationship between the responsibility and the selected object. Confirm your entries.

   ![Arrow]

   In the Overall View, you can also specify a validity period for the relationship.

Result
At runtime, the workflow evaluates the possible agents and ensures that the right tasks are routed to the right agents at the right time.
Defining Roles Using Organizational Data

Use
A Business Object is often related with an organizational entity (such as MRP controller, laboratory, sales group, purchasing organization, or planner group) by virtue of its master data. From a technical perspective, organizational entities are represented by object types in the Business Object Repository. The indicator organizational type in their basic data defines such object types as SAP Organizational Objects [Seite 1291]. The attributes of an application object type can be defined with a data type reference to a SAP organizational object type.

These can be evaluated for role resolution purposes to locate the agent of a step. For this to function correctly, you must assign specific agents in the form of positions or organizational units from Organizational Management to the abstract SAP organizational objects. At runtime, these assignments are evaluated in a role resolution if the required input data is available.

The role resolution finds a valid organizational object from Organizational Management as output data.

There are some changes required to be made to material master data.

This task should be routed to employees in the Laboratory/Design Office stored in their material master data. SAP delivers the sample role LABOR from the SAP Organizational Objects for agent determination purposes. For more information, see Role to Determine a Design Office [Seite 1304].

Someone should contact the supplier of a certain material.

This task should be routed to employees in the purchasing group stored in the material master data.

Procedure

Relationship Between Organizational Plan and SAP Organizational Objects
You have to set up a relationship between SAP Organizational Objects and the corresponding organizational units or positions in the organizational plan. This step must always be performed because the organizational plan is set up by each customer specific to the enterprise.
For more information, see Assign SAP Organizational Objects [Seite 1293].

Definition of a Role for Evaluating this Relationship
You define a role according to a predefined schema that can evaluate the above relationship between the organizational plan and SAP Organizational Objects. This step is only required if you cannot use any of the roles delivered in the standard system.
For more information, see Defining Roles [Seite 1300].

Entering the Role for Specifying Responsibility
You enter the role as agent of an activity or a single-step task and maintain the role container binding.
For more information, see Defining Binding [Seite 1301].

The single step task Material Master: Maintain Design Data should always be processed by the MRP controller responsible for the material.
Definition time: The role Determine laboratory/design office is specified as the role of the agent of the step Material Master: Maintain Design Data. The binding definition is &Material.Labor& (Workflow Container) -> ORG_OBJECT_ID (Role container).

Run time: The material H4 Lamp, for example, is processed by the task. When the material is known, the agent determined for the work item could be the user that has the role of MRP controller for this material.

Result
At runtime, the workflow determines what organizational objects have the actual values of the key fields of the SAP organizational objects assigned to them. These organizational objects become the agents for the work items.
For more information, see Agent as Attribute of SAP Organizational Object [Seite 1302].
SAP Organizational Object

Definition
Instance of an *SAP Organizational object type* defined in the Business Object Repository.

Use
You can route tasks to the appropriate users by creating relationships between SAP Organizational Objects – which reside in the Business Object Repository (BOR) – and the *Organizational Management* objects. The system finds an agent by tracing the relationships between the task, the SAP Organizational Object and the *Organizational Management* object.

Before you can edit object assignments, you must choose the objects with which you want to work. You can then create relationships between the two kinds of objects. This allows you to use roles to identify potential agents for tasks in *SAP Business Workflow*. Once you have created these object assignments, you can edit, delete, delimit, and view them.

*SAP Organizational object types* represent organizational units on the object type level in the Business Object Repository. These units are used to form and describe employee groups.

Examples of organizational units and corresponding SAP organizational object types are:

<table>
<thead>
<tr>
<th>Object Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>BUS0005</td>
<td>Shipping point</td>
</tr>
<tr>
<td>BUS0007</td>
<td>Purchasing organization</td>
</tr>
<tr>
<td>SAP _40132</td>
<td>Work scheduler group</td>
</tr>
<tr>
<td>T024</td>
<td>Purchasing groups</td>
</tr>
<tr>
<td>T024D</td>
<td>MRP controller</td>
</tr>
<tr>
<td>T024L</td>
<td>Laboratory/office for material</td>
</tr>
<tr>
<td>TVKGR</td>
<td>Sales group</td>
</tr>
</tbody>
</table>

What does SAP deliver?
The attribute relationships between Business object types and *SAP organizational object types* mentioned above are, as a rule, already defined in the Business Object Repository.

Structure
A relationship between an *SAP organizational object* and a business object type is defined in such a way that the SAP organizational object is available as an attribute of an application object.

For the object type **BUS1001 (material)**, the attribute **Laboratory** is defined by a data type preference from object type **T024L (laboratory/design office for material)**.

Integration
SAP Organizational Objects must be entered in table T7791, in order for these assignments to *Organizational Management* objects to be possible. This table is preset in the appropriate format, and you can add new entries.
Assigning SAP Organizational Objects

Use
So that you can define roles using organizational data, you must create and edit assignments between SAP organizational objects and Organizational Management organizational objects.

For example, you want to restrict the task of buying certain materials to certain individuals within a specific organizational unit. This is achieved by creating an assignment between a purchasing group (purchasing groups are SAP Organizational Objects) and an organizational unit.

You can create assignments between any object classified as an SAP Organizational Object, and the Organizational Management objects – organizational units, positions, jobs, and work centers. You apply a validity period to these assignments, so that changes in responsibility can be shown.

Procedure
1. From the SAP menu, choose Tools → Business Workflow → Development → Definition tools → Organizational Management → SAP Org. Objects → Create Assignments.

   The Assignment to SAP Organizational Objects: Initial screen appears.

2. In the Organizational unit and Selection period fields, identify the Organizational Management objects you want to edit.

   Entries in these fields allow you to confine the objects you edit to:
   - A specific area of the organizational plan
   - Objects in the organizational plan that are valid during a specified time frame

   Enter a superior organizational unit if you want to create a relationship between an SAP organizational object and a position. You will subsequently be able to navigate from the selected organizational unit to the object.

3. In the View dialog box, select the SAP Organizational Objects you want to edit. You can choose either:
   - Organizational object type
     You confine your work to specific SAP Organizational Object Types. This reduces the number of steps you perform later, if creating assignments.
   - All organizational object types
     You can work with all types of SAP Organizational Objects. Select an object.

4. You can also create, delimit or delete object assignments.

Assignments between SAP organizational objects and organizational objects from Organizational Management can also be created in Infotype 1208 [Extern].
Creating Object Assignments

Prerequisites

You want to create a relationship between an organizational unit or a position and an *SAP organizational object*, in order to create the link between positions and MRP controllers or organizational units and design offices.

Prerequisites

You are in the Assigning *SAP Organizational Objects: Initial screen* and have selected the *organizational unit* and *SAP organizational object*, which you want to edit. (See Assigning SAP Organizational Objects [Seite 1293].)

Procedure

1. Choose Assignment → Change. Another screen appears, displaying the Organizational Management objects you selected.

   The tree structure can display additional information, including existing assignments with *SAP Organizational Objects*, and other Organizational Management objects in the organizational plan. To adjust the data displayed so that it meets your requirements, choose View.

2. Select the Organizational Management object you want to assign.

3. Choose Assignment → Create.

4. The procedure now varies, depending on the selections made in step 2:

5. If you are working with a specific *SAP Organizational Object* type, the appropriate dialog box appears, in which you can identify a specific object (for example, a specific purchasing group).

   – Make the appropriate selections from the dialog boxes.

   If you are working with all *SAP Organizational Object* types, a series of dialog boxes appears, allowing you to identify the type of *SAP Organizational Object* type you want to work with, and then a specific object.

   – Make the appropriate selections from the dialog boxes.

   You want to create a relationship between the organizational unit 50001285 and the *SAP organizational object Laboratory*, which is described by the key field 002.

   To do so, create a new relationship between the organizational unit and the SAP organizational object T024L (Laboratory). A dialog box appears in which you can specify the key field for this *SAP organizational object*.

Result

The system saves the assignment and displays it in the tree structure.
Function Module
The following function module enables you to set up a relationship between an SAP organizational object and a position or organizational unit:

**RH_SAP_ORG_OBJC_RELATE**
Assigns an SAP organizational object to an object in Organizational Management

**Interface**
**Export Parameters**
- **ACT_OBJTYPE**, reference field P1208-OBJTYP
- **ACT_OBJKEY**, reference field P1208-OBJKEY

**Exceptions**
- **SAP_OBJECT_KEY_NOT_VALID**

When the function module is accessed, you specify the SAP organizational object type (name is taken from the Business Object Repository) and the object type-specific key that is used to uniquely identify an object of this type. You assign an organizational unit to this SAP organizational object in the input field. This relationship is transferred to Organizational Management.
Delimiting Object Assignments

Use
You can delimit object assignments to change the validity period applied to the relationship between an Organizational Management object and an SAP Organizational Object, so that the end date occurs sooner than stated.
It can be necessary to delimit the object assignment, for example, if you plan to redirect responsibility for a task at a specific time in the future.

Prerequisites
You are in the Assigning SAP Organizational Objects: Initial screen and have selected the organizational unit and SAP organizational object, which you want to edit. (See Assigning SAP Organizational Objects [Seite 1293])

Procedure
1. Choose Assignment → Change.
   Another screen appears, displaying the Organizational Management objects you selected.
   
     The tree structure can display additional information, including existing assignments with SAP Organizational Objects, and other Organizational Management objects in the organizational plan. To adjust the data displayed so that it meets your requirements, choose View.

   2. Select the assignment that should be delimited:
      a) Locate the two objects in the assignment in the tree structure
      b) Choose the object that is at the lower level of the tree structure

   3. Choose Assignment → Delimit.
      A screen appears, displaying the assignment information.

   4. In the second Validity field, enter the appropriate end date for the validity period.

   5. Choose SAP OrgObjects → Delimit.
      A message appears confirming the system has delimited the assignment.
Deleting Object Assignments

Use
You should only delete assignments between *Organizational Management* objects and *SAP Organizational Objects* only if you want to erase all record of an assignment from the database.

⚠️
Deletions should only be necessary if positions have been created incorrectly or by accident. If you want to indicate that responsibilities have changed, use the delimit feature instead.

Prerequisites
You are in the *Assigning SAP Organizational Objects: Initial screen* and have selected the *organizational unit* and *SAP organizational object*, which you want to edit. (See *Assigning SAP Organizational Objects [Seite 1293]*)

Procedure
1. Choose *Assignment → Change*.
   
   Another screen appears, displaying the *Organizational Management* objects you selected.

   ![Tree structure](image)
   The tree structure can display additional information, including existing assignments with *SAP Organizational Objects*, and other *Organizational Management* objects in the organizational plan. To adjust the data displayed so that it meets your requirements, choose *View*.

2. Select the assignment, which you want to delete:
   a) Locate the two objects in the assignment in the tree structure
   b) Choose the object that is at the *lower* level of the tree structure

3. Choose *Assignment → Delete*.
   A message appears asking you to confirm that you want to delete.

4. Choose *Yes*.
   The system deletes the assignment.
Define Role

Prerequisites
You want to define a role whose resolution refers to an SAP organizational object type.

Procedure
1. Create a new role. To call the transaction for processing roles, access the SAP menu and choose Tools → Business Workflow → Development → Definition tools → Standard roles → Create.
2. Select the Organizational data checkbox.
3. Specify an SAP organizational object type.

Result
The role container is created automatically. It includes just one element, Org_Object_ID, in which the object reference to the SAP organizational object is stored. In this instance, you do not need to specify a function module or define a role container when defining the role.
Define Binding

Prerequisites
You have used your role container to create a role whose resolution is based on the evaluation of organizational data. For more information, see Define Role [Seite 1300].
To "provide" the role container with the object reference to the SAP organizational object, you now define binding for the role container:
- If you use the role for a single-step task, you define binding from the task container.
- If you use the role for a step, you define binding from the workflow container.

Procedure
Assign the following to the role container element: an expression that references the SAP organizational object as an attribute of the application object to be processed.

\[ \text{Org\_Object\_ID} \leq \&<\text{object reference}>.\langle\text{object reference}\rangle& \]

At runtime, the object reference to the processed object of the type material is included in the workflow container under the name Material. You assign the laboratory attribute of this object to the role container in the binding definition. By doing so, you take advantage of the fact that an attribute has been created for the material object type under the name Laboratory that includes an object reference to the design office. You define binding as follows:

<table>
<thead>
<tr>
<th>Org_Object_ID</th>
<th>&lt;=</th>
<th>&amp;Material.Labor&amp;</th>
</tr>
</thead>
<tbody>
<tr>
<td>(Role container)</td>
<td></td>
<td>(Workflow container)</td>
</tr>
</tbody>
</table>
Agent as Attribute of SAP Organizational Object

Use

Organizational entities, which are specified in the master data of a business object, are usually supported by positions or organizational units, which may need to be addressed as the recipient of a work item. The following description illustrates the procedure used to determine these agents on the basis of organizational entities, and how they are indirectly available as attributes of the business object.

SAP Organizational Object Types

Prerequisites

Business Object Type and SAP Organizational Object Type

From a technical perspective, organizational entities are represented by SAP organizational objects [Seite 1291] in the Business Object Repository.

The attributes of an application object type can be defined with references to these SAP organizational object types.

SAP Organizational Object Type and the IFSTROBJCT Interface

Each SAP organizational object type (example: design office) should support the IFSTROBJCT interface. As a result, this object type inherits the agent attribute. This attribute returns the position or organizational unit that is related to the SAP organizational object.

Many SAP organizational object types supplied by SAP already support the IFSTROBJCT interface.

You only need to enhance a subtype of the corresponding object type with the IFSTROBJCT interface if this is not the case. Programming is not required.
Agent as Attribute of SAP Organizational Object

Activities

Relationship Between Organizational Plan and SAP Organizational Objects

Each SAP organizational object must be related to the corresponding position or organizational unit. This step must always be performed because the organizational plan is initially determined for specific customers. For more information, see Assign SAP Organizational Objects [Seite 1293].

Responsibility for a Step

When responsibility is determined for a step as part of a workflow definition, the agent can be derived from the workflow container and specified using a multi-level expression of the following type: &<business object type>.<SAP organizational object type>.agents&.

On the entry screen for Responsibility, select Container and enter &Material.Labor.Agents&.

This procedure means there is no need to define and use a corresponding role.

Error Handling

When this step is performed, the following error situations can arise after the expression has been evaluated:

- The agent that is determined does not belong to the possible agents for the underlying single-step task.
- The relationship between the organizational plan and organizational object (see above) is not maintained, or not maintained in full.

If one of the above error situations arises, the step is instantiated as a work item and addressed to all of the possible agents for the single-step task.

Reference to Related Topics

An alternative concept exists that also makes use of the link between organizational entities and business objects and requires the definition of a role. This concept, which appears initially to be less easy to use, is required, for example, if you need to enter a role as a default role for a single-step task, or if a different method must be used to solve errors.
Role to Determine Design Office

Definition
A role delivered by SAP for determining the employees in a particular design office.

Use
This role enables you to use material to be processed to address the controller responsible for the material. As a result, the object reference to the material to be processed (object type BUS1001) is usually included in the WI_Object_ID element of the task container, and/or in a Material element (or similar) of the workflow container.
This BUS1001 object type has the Laboratory attribute, so that an expression of type & WI_Object_ID.Laboratory& and/or &Material.Laboratory& must be specified as the source of binding for the Org_Object_ID element of the role container.

Structure

<table>
<thead>
<tr>
<th>Role:</th>
<th>30100012</th>
</tr>
</thead>
<tbody>
<tr>
<td>Abbreviation:</td>
<td>Laboratory</td>
</tr>
<tr>
<td>Name:</td>
<td>Determine laboratory/design office</td>
</tr>
</tbody>
</table>

Integration
Individual positions from the organizational plan must be related with the corresponding SAP organizational objects of the T024L Laboratory/design office for material type. These relationships are evaluated for role resolution.
Defining Roles Using Function to Be Executed

Use
You use function modules for role definition when very complex selection criteria are required for agent determination, in other words, when it is not possible to use responsibilities to model agent selection. Function modules provide you with a very powerful tool for determining the agent of a task in Workflow.

Prerequisites
1. You must specify what job you expect the workflow to perform. Depending on this, you decide whether you want to use a standard function module delivered by SAP or your own function module that suits your requirements exactly. To avoid having to display the coding of the function module during the procedure, you should be absolutely sure about what container elements are required by the function module beforehand. If you do need to check the coding of a function module, on the Standard Role: Display screen, choose Goto → Function module.

2. You have created a role. To call the transaction for processing roles, access the SAP menu and choose Tools → Business Workflow → Development → Definition tools → Standard roles → Create.

Procedure
On the Standard Role: Create screen:
1. Select Function to be executed.
2. Enter the name of the function module. For example, if you want to use a function module for reporting on the organizational structure, you can enter RH_GET_STRUCTURE.
3. Save the role.
   Depending on what function module you used, the field Evaluation path appears.
4. Enter the relevant evaluation path and choose Save.
5. Choose Container Definition.
   The Standard Role: Process Container screen appears.


7. If you want to use Data Dictionary fields, choose Yes.
   The Create with Data Dictionary Field Defaults dialog box appears.

8. Specify the table from which you want to select fields and Choose Continue.
   The table appears.

9. Select the fields that you want to appear in the container definition. Bear in mind that these are the elements required by the function module.
   The system prompts you to confirm the texts for the container elements.

10. Choose Back.
    The Standard Role:Create screen appears.

**Result**
Workflow executes the function module and, depending on the container data, returns a list of possible agents.
Function Module for Role Resolution

The following excerpts from a fictitious function module for role resolution, which determines the agent responsible on the basis of a release code and object to be released, can be used as an example. The ReleaseCode and ReleaseObject elements are defined in the role container as role parameters.

```
FUNCTION GET_REL_RESPONSIBLE.

*"----------------------------------------------------------
*"  Lokale Schnittstelle:
* "    TABLES
* "      ACTOR_TAB STRUCTURE SWHACTOR
* "      AC_CONTAINER STRUCTURE SWCONT
* "      EXCEPTIONS
* "    NOBODY_FOUND
*"----------------------------------------------------------

INCLUDE <CNTN01>.

* define variables stored in container
DATA: RELEASE_OBJECT TYPE SWC_OBJECT.
DATA: RELEASE_CODE LIKE RM06B-FRGAB.

* local data
DATA: BEGIN OF RELOBJECTKEY,
    NUMBER LIKE EBAN-BANFN,
    POSITION LIKE EBAN-BNFPO,
  END OF RELOBJECTKEY.

REFRESH ACTOR_TAB.
CLEAR ACTOR_TAB.

* convert persistent container to runtime container
  SWC_CONTAINER_TO_RUNTIME AC_CONTAINER.

* read elements out of container
  SWC_GET_ELEMENT AC_CONTAINER 'ReleaseCode' RELEASE_CODE.
  SWC_GET_ELEMENT AC_CONTAINER 'ReleaseObject' RELEASE_OBJECT.

* separate object key
  SWC_GET_OBJECT_KEY RELEASE_OBJECT RELOBJECTKEY.

* loop and select table <TABLE> with
  * RELEASE_CODE
  * RELOBJECTKEY-NUMBER and RELOBJECTKEY-POSITION
    ....

* end of selection
* exception and parameter handling
```
IF SY-SUBRC NE 0.
   RAISE NOBODY_FOUND.
ELSE.
   ACTOR_TAB-OTYPE = 'TABLE'-ACTOR_TYPE.
   APPEND ACTOR_TAB.
ENDIF.

ENDFUNCTION.
Interface of Function Module for Role Resolution

The interface of a function module for role resolution is described by the following parameters:

**Table Parameters**

**AC_CONTAINER, Reference Structure SWCONT**
Role container with role-specific parameters that must be available as input values for role resolution.

**ACTOR_TAB, Reference Structure SWHACTOR**
Table with results of role resolution as return values.

The **SWHACTOR** structure has the following logical appearance:

<table>
<thead>
<tr>
<th>Field name</th>
<th>Type</th>
<th>Length</th>
<th>Short text</th>
</tr>
</thead>
<tbody>
<tr>
<td>Otype</td>
<td>CHAR</td>
<td>2</td>
<td>Object type in Organizational Management</td>
</tr>
<tr>
<td>ObjID</td>
<td>CHAR</td>
<td>12</td>
<td>ID of object in Organizational Management</td>
</tr>
</tbody>
</table>

The structure consists of two fields:

- **OType** includes a 2-character character field that contains the identification of the object type in Organizational Management. At this time, the following object types are permitted as the result of role resolution:
  
  - O Organizational unit
  - S Position
  - C Job
  - A Work center
  - US User name
  - P Person (PD master data)

  The entries for **OType** are checked against table T779O [Seite 1530].

- **ObjID** includes a 12-character character field that contains the identifying name of a user and/or the unique ID (8-digit number) of an object in Organizational Management.

**Exceptions**

**NOBODY_FOUND**
If the function module for role resolution is exited via the **NOBODY_FOUND** exception, the status of the **Cancellation for role resolution without result for further procedure** indicator is decisive.

- The indicator is set:
  
  The work item and/or workflow from which role resolution was requested is assigned the **incorrect** status.

- The indicator is not set:
  
  The work item and/or workflow from which role resolution was requested continues. To determine the agent, the possible agents are evaluated.

  The cause of the error, which may have been output as a message when the exception was triggered, is logged in the history of the work item and/or in the workflow step log. The message type is not relevant to error handling.
Error Handling for an Empty Table

If the function module for role resolution is not exited via its `NOBODY_FOUND` exception, and if the `ACTOR_TAB` table with the agents is still returned empty, the above information applies accordingly.
Role to Determine Manager

Definition
A role delivered by SAP for determining the manager of an agent, position, or organizational unit.

Use
Oftentimes, you will use this role to find the manager of the initiator of a workflow, or the manager of the current agent of a step. The _WF_Initiator element of the workflow container and _WI_Actual_Agent element of the task container are used to store the user names in a 14-character character field in accordance with the ROBJECTS-OBJECT reference. Binding, therefore, must be defined for the Org_Object element of the role container.
As an example, role 00000168 is also used in the example demo for processing a notification of absence. For more information, see Example Demo: Process Notification of Absence [Seite 52].

Structure

<table>
<thead>
<tr>
<th>Role:</th>
<th>00000168</th>
</tr>
</thead>
<tbody>
<tr>
<td>Abbreviation:</td>
<td>Manager</td>
</tr>
<tr>
<td>Name:</td>
<td>Manager of...</td>
</tr>
</tbody>
</table>

The SWX_GET_MANAGER function module is used to define this role.

Integration
The organizational plan of the enterprise is used for role resolution.
The link between an employee and his or her manager can be depicted in the organizational plan by two different relationships:

- Indication of the chief position for an organizational unit (position "manages" organizational unit, relationship A/B012).

- Direct reporting structure between positions (position "reports to" position, relationship A/B002).
Role Resolution Using Evaluation Paths

Use

The organizational situation of employees within an enterprise is depicted in the organizational plan. Using this existing information, you can perform role resolution starting from one particular employee to determine one other employee, or several other employees, along specific relationships in the organizational plan.

The following role evaluates a function module to perform resolution. Please note that this function module is provided by SAP as a default.

Starting from a particular employee, you want to determine his or her organizational unit so that you can address a work item to all of the employees in this organizational unit.

The role that includes this functionality is available in your system and can be used as an explanation of this documentation. For more information, see the role used to determine the organizational unit of a user [Seite 1316].

Integration

From a technical perspective, this role resolution is very similar to Role Resolution Using a Function to be Executed [Seite 1305]. The RH_GET_STRUCTURE function module must be used; an evaluation path is also specified.

Activities

1. Maintain an organizational plan with the appropriate relationships.
2. An evaluation path describes how the relationships between organizational objects are processed in a particular logical order, as required for the role resolution described above.
3. Define a role that evaluates these relationships. This role uses the RH_GET_STRUCTURE function module to perform resolution.

   This step is only required if you cannot use any of the roles delivered in the standard system. For more information, see Define Role [Seite 1300].

4. If you enter the role as an agent for an activity or as a default role for a single-step task, maintain binding for the role container. For more information, see Define Binding [Seite 1314].
Define Role to Evaluate Evaluation Paths

Use
You want to define a role whose resolution is based on the evaluation of evaluation paths.

Procedure
4. Create a new role. To call the transaction for processing roles, access the SAP menu and choose Tools → Business Workflow → Development → Definition tools → Standard roles → Create.
2. Select the Function to be executed checkbox.
3. Enter RH_GET_STRUCTURE as the function module for role resolution.
   The named function module is provided by SAP for this purpose. Further programming is not required.
   If this function module is entered as a function module for role resolution, you can specify the evaluation path in an additional dialog box.
4. Define the role container.
   The role container for a role based on the RH_GET_STRUCTURE function module must only contain the following elements:
   Element OType, data type reference OBJEC-OTYPE: type of object in Organizational Management according to the first step in the evaluation path.
   Element Org_Agent, data type reference WFSYST-AGENT: C14 field as combination of object type in Organizational Management and object in Organizational Management.
   Create the first two elements, or the last element, as role parameters.
   The object type in Organizational Management (example: US) and the object in Organizational Management (example: SCHMIDT) can be transferred either to two separate fields or to one field (example: USSCHMIDT). Data specified in one field is evaluated first.
   You define the manager_of role (manager of a user). To do so, you refer to the RH_GET_STRUCTURE function module and specify US_CHEF as the evaluation path.
   You create a new element with the name Org_Agent in the role parameter container. You define this container element as an obligatory element with a data type reference to dictionary table field WFSYST-AGENTS.
Define Binding

Prerequisites
You have created a role based on the RH_GET_STRUCTURE function module using your role container. To "provide" the role container with the agent that represents the starting point of role resolution, you now define binding for the role container:

- If you use the role as a default role for a single-step task, you define binding from the task container.
- If you use the role as a role for a step, you define binding from the workflow container.

The workflow container and task container include system fields that are always available. In the standard system, they are filled by the workflow system. These container elements often include the information required as role parameters and can, therefore, be used as a source for binding. The following container elements include information on users in a C14 field in the <USName> structure.

- The _WF_Initiator element of the workflow container
- The _WI_Actual_Agent element of the task container

Procedure
The procedure is explained using the example of an "approve leave" single-step task. This task must always be completed by the manager of the person who submits the request. Therefore, you want to enter the manager_of role as the default role, and then integrate this customer task as an activity in a workflow definition.

1. Create an element in the task container called applicant with reference to the WFSYST-AGENTS ABAP Dictionary field.
2. Declare the manager_of role as the default role of this customer task.
3. Define the following binding from the task container to the role container:

<table>
<thead>
<tr>
<th>Org_Agent</th>
<th>&amp;Applicant&amp;</th>
</tr>
</thead>
<tbody>
<tr>
<td>(Role container)</td>
<td>(Task container)</td>
</tr>
</tbody>
</table>

4. In the description of the appropriate activity within the workflow definition, define the following binding from the workflow container to the task container:

<table>
<thead>
<tr>
<th>APPLICANT</th>
<th>&amp;_WF_Initiator&amp;</th>
</tr>
</thead>
<tbody>
<tr>
<td>(Task container)</td>
<td>(System variable in workflow container)</td>
</tr>
</tbody>
</table>
Binding Definition for Role Resolution
Role for Determining Organizational Unit of a User

Definition
A role delivered by SAP for determining the organizational unit to which a particular employee belongs.

Use
This role enables you to determine an employee’s organizational unit on the basis of the employee. A distinction is made between the following scenarios:

- The role is used in conjunction with a single-step task, which is classified as a general task [Extern].
  
  The generated work item can be viewed by all of the employees from the organizational unit that has been determined (and can be processed by a user from this organizational unit).

- The role is used in conjunction with a single-step task whose possible agents [Extern] are specified by one or more positions.
  
  The work item can only be viewed by employees with a position that belongs to the organizational unit that has been determined and is part of the possible agents.

- The role is used in conjunction with a single-step task whose possible agents [Extern] are specified by a job.
  
  The work item can only be viewed by employees with a position that belongs to the organizational unit that has been determined and is described by the job.

  The possible agents of a single-step task result from the relationship with the job of secretary. If the described role is used, a work item can always be directed to the secretary of the organizational unit in question.

Structure

<table>
<thead>
<tr>
<th>Role</th>
<th>30000011</th>
</tr>
</thead>
<tbody>
<tr>
<td>Abbreviation</td>
<td>FindOwnOrgUn</td>
</tr>
<tr>
<td>Name</td>
<td>Organizational unit of a user</td>
</tr>
</tbody>
</table>

This role uses the RH_GET_STRUCTURE function module in conjunction with the WF_ORGUN evaluation path (organizational unit of a user/person).

Integration
Each user whose organizational unit must be found must, of course, belong to an organizational unit via his or her position.
The Cancellation for Role Resolution Without Result Indicator

Definition
Indicator that determines how the system reacts if role resolution fails to find a valid agent.

Use
It is possible for role resolution to fail to find a valid agent. This is the case:
- If role resolution runs with errors and does not provide any results at all
  (From a technical perspective: the function module for role resolution is exited via its NOBODY_FOUND exception, and/or returns an empty agent table.)
- If role resolution provides agents that do not belong to the possible agents [Extern] of the single-step task.
  (Of course, if recipients or persons responsible for workflow are expressed by specifying a role, the latter cannot occur.)

<table>
<thead>
<tr>
<th>If this indicator is...</th>
<th>Then...</th>
</tr>
</thead>
<tbody>
<tr>
<td>SET</td>
<td>Low is cancelled if role resolution fails to find an agent. This protects the data.</td>
</tr>
<tr>
<td>NOT SET</td>
<td>All of the possible agents in the system become actual agents for the work item if role resolution fails to find an agent. The task is completed.</td>
</tr>
</tbody>
</table>

The error situation is recorded in the step log.

Role resolution is used to distribute invoices to various buyers at your enterprise. Only Mr Miller can execute this work item if the invoice amount exceeds $5,000. If you set the Cancellation for role resolution without result indicator, and if role resolution fails to find an agent for an invoice that exceeds $5,000, the workflow is cancelled. Only the workflow administrator can restart it. If you do not set the Cancellation for role resolution without result indicator, and if role resolution fails to find an agent for an invoice that exceeds $5,000, all buyers can process the invoice.
Selecting Objects from the Organizational Plan

Use
As well as defining roles, it can be necessary to describe certain relationships between employees and areas of responsibility or the borders between areas of responsibility in system tables. This information is then available for role resolution.

Create a table by assigning certain companies and invoice amounts to administrators via their positions.

<table>
<thead>
<tr>
<th>Position</th>
<th>Company</th>
<th>Amount ($)</th>
</tr>
</thead>
<tbody>
<tr>
<td>50001234</td>
<td>A - H</td>
<td>0 - 50.000,-</td>
</tr>
<tr>
<td>50001235</td>
<td>A - H</td>
<td>&gt; 50.000,-</td>
</tr>
<tr>
<td>50001236</td>
<td>I - Z</td>
<td>0 - 60.000,-</td>
</tr>
<tr>
<td>50001237</td>
<td>I - Z</td>
<td>&gt; 60.000,-</td>
</tr>
</tbody>
</table>

If the name of the company and the invoice amount are known, the details in this table are used for role resolution, in order to find an administrator.

The following function modules are available, to ensure that you have access to objects from Organizational Management which you need to fill the table and that you can program possible entries.

Features

**RH_DETERMINE_ORG_OBJECT**
Determines the ID of any Organizational Management object.

**Interface**

**Import Parameter**
- ORG_OBJECT, Reference structure SWHACTOR

**Exceptions**
- NO_ACTIVE_PLVAR
- NO_OBJECT_ID_SELECTED
- NO_OBJECT_TYPE_SELECTED

This function module can be called without transfer parameters.
When you access this function module, as dialog box is displayed from which you can select an object type from Organizational Management.
Once you have selected an object type from Organizational Management, one of the Organizational Management objects stored for this type can be selected and displayed using the search function.
The object type and ID of this Organizational Management object is returned as an export parameter in the SWHACTOR structure and can be used in your application.

**RH_DETERMINE_ORG_OBJID**
Determines the ID of an Organizational Management object from a predefined Organizational Management object type.

**Interface**

**Export parameter**
- ORG_OBJECT_TYPE, Reference field OBJEC-OTYPE
Import Parameter
- ORG_OBJECT_OBJID, Reference field SWHACTOR-OBJID

Exceptions
- NO_ACTIVE_PLVAR
- NO_OBJECT_ID_SELECTED

This function module forms only the “second half” of the function module described above. When you access it, you transfer the Organizational Management object type whose possible values you want to display using the possible entries function. You transfer this Organizational Management object type as a 1 or 2 character ID. You can select and display one of the entries, which is stored for the Organizational Management object type using the search function.

The object type and ID of this Organizational Management object is returned as an export parameter in the variable ORG_OBJECT_OBJID and can be used in your application.
Specify Agent, Recipient, or Administrator by His or Her Role

Use
You want to specify an agent, the workflow administrator, or a recipient by his or her role. To do so, you can use the roles you defined yourself or the standard roles provided by SAP.

Procedure
The procedure for entering a role is always the same:
1. To enable you to specify the agent/administrator/recipient as a role, select Role.
2. Enter the unique, 8-digit number of the role in the appropriate input field.
   
   If you do not know what this number is, use the input help function. In the standard system, the input help function displays the abbreviation and description of the role. By choosing F17, you can display the plan version and number of the role instead of the description.
3. Define binding for assigning values to the role container:
   
   When a role is specified in the task definition or workflow definition, the role container must be filled with values from the appropriate container (task container or workflow container) via a binding definition. The binding definition editor is available for this purpose. It enables you to define the appropriate assignments to the elements of the role container.
Business Workplace: Workflow Functions

Use
You use this part of the Business Workplace [Extern] if you want to use the functions of SAP Business Workflow. The Business Workplace is the main interface between an end user and the workflow system. All dialog and missed deadline work items to which the user is assigned as a recipient are displayed in the user’s workflow inbox.

As a head of department, you are responsible for approving leave requests. The relevant approval process is implemented using a workflow in your enterprise.

The requests (in the form of work items) appear in your worklist (workflow inbox) and must be rejected or approved there.

The rejected or approved requests (executed work items) are not only returned to the applicants after processing, but are also put into your workflow outbox (under Work items executed by me). You can therefore check the requests you have processed.

Features

Workflow settings
You can configure the workflow functions in the Business Workplace using the personal workflow settings [Seite 1394].

Business Workplace screen areas
The Business Workplace has three screen areas, which are used in the following manner by SAP Business Workflow:
Overview tree

The following workflow functions are available under the Inbox node, which is under the initial node Workplace:

- Workflow
  - Grouped according to task
  - Grouped according to content
  - Grouped according to content type
  - Grouped according to sort key
- Overdue entries
- Deadline messages
- Incorrect entries

For information on these functions, refer to Workflow Inbox [Seite 1408].

The following functions are available under the Outbox node, which is under the initial node Workplace:

- Started workflows
- Work items executed by me
- Forwarded work items

For information on these functions, refer to Workflow Outbox [Seite 1441].

The Resubmissions node is located under the initial node Workplace and contains the:

- Workflow resubmissions [Seite 1443]
Worklist
The worklist [Extern] is displayed in the upper right corner of the Business Workplace screen. Depending on whether you are in the workflow inbox, the workflow outbox or the workflow resubmissions, you have various functions available to you, which are described at the respective locations.

Work item preview
In the lower right corner of the Business Workplace screen, a work item selected in the worklist is displayed in a preview [Seite 1445]. Not all the functions of the work item display or the workflow log are available.
A user exit [Extern] can be used to configure the work item preview to suit your individual requirements.

Support for context menus
All workflow functions can be called using the relevant context menu.

Workflow Toolbox
SAP Business Workflow's Workflow Toolbox [Seite 1446] enables the user to access workflow functions even during a workflow-driven application transaction.

E-mail notification for new work items
The report RSWUFML can be used to inform an employee by mail that there is a new work item in their Business Workplace inbox.
This function is therefore beneficial to all employees who do not work with their Business Workplace on a daily basis.

Activities
To access the Business Workplace from the SAP Easy Access screen, choose one of the following options:

- Menu → Business Workplace
Work Item

Definition

Object that represents a task or action in the workflow system at runtime.

Use

Work items are subdivided into a specific work item type according to their assignments. The internal processing procedures are controlled via this work item type. The work item type determines which statuses and transitions are valid.

Depending on the work item type, some of these work items are displayed in a user's work list. Other work items, on the other hand, are only used and processed internally.

Structure

Work item types displayed in the Business Workplace

<table>
<thead>
<tr>
<th>Work item type</th>
<th>Short text</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>W</td>
<td>Work item with dialog</td>
<td>Runtime representation of single-step tasks that require interaction with the user.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Runtime representation of user decisions</td>
</tr>
<tr>
<td>D</td>
<td>Missed deadline</td>
<td>Work item for notification of missed deadline</td>
</tr>
<tr>
<td>A</td>
<td>Work Queue</td>
<td>A work queue is a list of objects to be processed once and together in a limited time frame.</td>
</tr>
</tbody>
</table>

Other work item types

<table>
<thead>
<tr>
<th>Work item type</th>
<th>Short text</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>F</td>
<td>Workflow</td>
<td>Runtime representation of a multistep task</td>
</tr>
<tr>
<td>B</td>
<td>Batch item</td>
<td>Runtime presentation of a single-step task that runs in the background</td>
</tr>
<tr>
<td>E</td>
<td>Wait step work item</td>
<td>Runtime representation of a wait step in the workflow definition</td>
</tr>
<tr>
<td>C</td>
<td>Container anchor</td>
<td>This type of work item is required as a special development in the EDI environment. It does not normally appear in the workflow environment. Work items of this type should be regularly deleted or archived.</td>
</tr>
</tbody>
</table>
Dialog Work Items (Type W)

Definition
Work item that represents a task at runtime that requires interaction with the user.

Since the user decision is also represented internally by a task, a dialog work item can also represent a user decision.

When a dialog work item is executed, the underlying object method of the task is called. The deadlines for executing dialog work items are monitored.

Use
A dialog work item is displayed with ready status in the workflow inbox of the Business Workplace. It is removed from the integrated inboxes of the other agents when the recipient reserves, executes, or processes this work item with other functions.

The database oriented approach used in SAP Business Workflow allows a work item to be seen by several recipients equally authorized in organizational terms in their inboxes and executed from there. However, only one recipient can actually reserve this work item for processing and execute it. The work item is then no longer available to any other recipients.

Integration
A task represented by a dialog work item can be

- a step in a workflow definition:
  
  In the workflow definition, reference is made to tasks in the activity and user decision steps.

- started as single steps via an event or in dialog:
  
  Tasks can be started as elementary activities directly in dialog or via a triggering event. These tasks are then also represented by a dialog work item in the workflow inbox.
## Status of a Dialog Work Item

The valid statuses for dialog work items (type W) are listed in the following table.

<table>
<thead>
<tr>
<th>Work item status</th>
<th>Description</th>
</tr>
</thead>
</table>
| **waiting**      | The work item has been scheduled for its *requested start*.  
                    A work item has this status  
                    - if it already exists but the *requested start* specified in the workflow definition has not been reached yet.  
                    - if it has been set to resubmission  
                    Work items in the *waiting* status are not displayed in the workflow inbox. |
| **ready**        | The work item has been released for execution and appears in the workflow inbox of all recipients. |
| **reserved**     | The work item has been received by one of its recipients with the result that its status has changed from *ready* to *reserved*.  
                    A work item in the *reserved* status is then displayed to this recipient only. It is no longer displayed in the workflow inboxes of the other recipients. |
| **In process**   | The work item is currently being processed by a recipient or in a different mode.  
                    A work item also has this status  
                    - if the work item is waiting for its terminating event.  
                    - if the user cancelled the method.  
                    - if the method was ended with a temporary exception for which no subsequent steps have been modeled.  
                    The point at which processing is completed cannot be detected by the workflow system in this status. As long as the status of the work item is set to *in process*, database changes have not been made. |
| **Executed**     | The work item is awaiting explicit confirmation of its completion.  
                    The work item only has this status if it is necessary to confirm that it has been completed. A work item with *executed* status can be executed or forwarded several times until it is set to the status *done* in the Business Workplace.  
                    In this way, groupware components are realized in SAP Business Workflow. |
| **completed**    | The execution of the work item is completed.  
                    The result of the task represented by the work item is correct, i.e. the result modeled in the workflow definition.  
                    Work items in the *completed* status are not displayed in the workflow inbox of the Business Workplace. |
| **Logically deleted** | Execution of the work item is no longer meaningful or required by the workflow logic.  
                    A work item changes to the *logically deleted* status in the following way:  
                    - Termination in parallel processing branches  
                      When the required number of processing paths has been executed in a fork, the work items in the other paths that have not yet reached the *completed* status are automatically set to the *logically deleted* status.  
                    - Intervention by an administrator  
                      An administrator can only set a work item to the *logically deleted* status if it has not yet reached the *completed* status and is not part of a higher-level workflow. |
<table>
<thead>
<tr>
<th><strong>Status of a Dialog Work Item</strong></th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Work item status</strong></td>
</tr>
<tr>
<td>Work items in the <em>logically deleted</em> status are not displayed in the workflow inbox of the Business Workplace.</td>
</tr>
<tr>
<td>A work item with the <em>logically deleted</em> status may have caused database changes or other actions (generate event, send notification). These changes are not canceled automatically (compensated).</td>
</tr>
<tr>
<td><strong>Error</strong></td>
</tr>
<tr>
<td>Execution of the work item was terminated with an error.</td>
</tr>
</tbody>
</table>

In addition to the statuses given above, a work item can be **locked against execution**. This is possible in any status and involves an administrative function which is selected for a work item via the *Change* option.
Status Transitions of a Dialog Work Item

The diagram below shows the possible status transitions that a dialog work item (type W) can undergo:

The arrows are labeled with the functions used by a user to trigger the respective status transition.

**Comments**

For further information, refer to Status of a Dialog Work Item [Seite 1373].

**Transition from status waiting**

The transition from the status waiting to the status ready is performed automatically by the work item manager when the requested start is reached. A workflow system administrator can set a work item to the status ready manually.

**Transition from status ready**

From this status, the work item passes either to the status reserved or via the status in process to the status completed.

**Transition from status reserved**

A work item with the status reserved can be reset to the status ready.

**Transition to and from status in process**

A work item with the status in process can be reset to the status ready. A workflow system administrator can reset a work item manually. This function is available when changing the work item.

**Transition from status executed**

After confirmation of end of processing, the work item assumes the status completed.
Transition from status completed

Work items with the status completed can no longer be set to another status even if a workflow system administrator intervenes.

Transition from status error

A workflow system administrator can intervene and set work items with errors to the status in process or the status logically deleted (possibly after eliminating the error).

Transition from status logically deleted

Work items with the status logically deleted can no longer be set to another status even if a workflow system administrator intervenes.
Missed Deadline Work Item (Type D)

Definition
Notification of a deadline recipient if the runtime system detects that the deadline for a certain work item has been exceeded.

Use
This work item informs its recipients that a deadline (start or end deadline) of a monitored work item has been exceeded. The recipients are informed by means of a missed deadline work item (type D) in the workflow inbox of the Business Workplace.
When it is executed, this work item displays information on the monitored (and now late) work item. The text for notifying the recipient is set by default.
When a deadline is monitored for an activity [Seite 1024] or user decision [Seite 1074], it is also possible to enter an individual text in the respective task definition.
## Status of a Missed Deadline Work Item

The valid statuses for missed deadline work items (type D) are listed in the following table:

<table>
<thead>
<tr>
<th>Work item status</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Ready</td>
<td>The work item appears in the Business Workplace of the recipients.</td>
</tr>
<tr>
<td>In process</td>
<td>One of the recipients has executed the work item.</td>
</tr>
<tr>
<td>Completed</td>
<td>The execution of the work item is completed. Work items in the <code>completed</code> status are not displayed in the workflow inbox of the Business Workplace.</td>
</tr>
</tbody>
</table>
Status Transitions of a Missed Deadline Work Item

The following status transitions are possible for missed deadline work items (type D):

**Transition to and from status *in process***

The status of a missed deadline work item changes from *ready* to *in process* when it is executed by a user.
Execution of a missed deadline work item displays the most important information on the monitored work item.
The missed deadline work item remains in the *in process* status until end of processing is confirmed explicitly. Until then, the work item can be executed or forwarded several times.

**Transition to *completed* status**

The status of the missed deadline work item changes to the *completed* status when completion of processing has been confirmed explicitly.
Work Queue Work Item (Type A)

Definition
Work item that represents a work queue. A work queue comprises a list of objects that are to be processed once together within a specified period. The work queue serves as a framework for the individual entries to be processed and manages the list of objects to be processed including their statuses and the tasks to be performed on them. Work queue work items are displayed in the Business Workplace. The work item status indicates the overall processing status of the work queue.

Use
Once you have created the work queue work item and you know its work item ID, you have the following options:

- Process the work queue within a workflow.
- Control the release, processing and status evaluation of the work queue with function modules.
- Control the release and processing of the work queue by processing the work queue work item directly.

Integration
To create the work queue work item from the list, you call the function module SWZ_AI_CREATE.
Status of a Work Queue Work Item

A work queue work item can have the following statuses:

<table>
<thead>
<tr>
<th>Work item status</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>waiting</td>
<td>The work queue has been created but cannot be released yet.</td>
</tr>
<tr>
<td>ready</td>
<td>The work queue work item (type A) is displayed in the Business Workplace of the recipients passed in the table AGENTS of the function module SWZ_AI_CREATE. If the work queue is to be processed via a workflow and therefore not appear as a separate work item in any Business Workplace, a non-existent user must be specified in the table when the work queue is created. The work queue must be reserved for further processing.</td>
</tr>
<tr>
<td>reserved</td>
<td>A person with release authorization has reserved the work queue.</td>
</tr>
<tr>
<td>in process</td>
<td>Work queue processing has begun.</td>
</tr>
<tr>
<td></td>
<td>– Dialog work items (type W) have been created for entries with dialog.</td>
</tr>
<tr>
<td></td>
<td>– For entries without dialog marked accordingly, the methods have been executed directly.</td>
</tr>
<tr>
<td>completed</td>
<td>All lines in the work queue have the status COMPLETED or CANCELLED and have reported back accordingly to the work queue. A work queue work item (type A) that assumes this status automatically creates the event created.</td>
</tr>
<tr>
<td>logically deleted</td>
<td>Further processing of the work queue is invalid (and therefore no longer possible).</td>
</tr>
<tr>
<td>error</td>
<td>At least one line of the work queue has the status error.</td>
</tr>
</tbody>
</table>
Workflow Work Item (Type F)

Definition
Work item that represents a multistep task at runtime.

Use
For every multistep task started there is one type F work item. The workflow log and the workflow container can be accessed:
- For error diagnosis and error correction if no work items are displayed
- For information on steps of a workflow (including their current agents, notes, and ad hoc objects) already processed
- For modifying an ongoing workflow by changing the workflow container
- For reporting on completed processes
Type F work items are not displayed in the Business Workplace but can be found using the work item selection [Seite 1490].

Structure
A workflow consists of a sequence of work items that are executed by agents or the system. The work items represent the steps in the workflow definition that refer to a particular task. These are steps of the types activity [Seite 1024] or user decision [Seite 1074].

Integration

The Work Item Manager manages the processing of work items and monitors deadlines. To automate workflow processes, activities in the workflow can also refer to object methods which run in the
background. If this is the case, the work item manager initiates the calling of the background processes. The work items whose execution requires dialog can be accessed by the selected agents (determined from the organizational model and role resolution) from their worklists in order to select them for processing. This worklist is displayed and managed in the workflow inbox of the Business Workplace [Seite 1368].
Status of a Workflow Work Item

The valid statuses for workflow work items (type F) are listed in the following table:

<table>
<thead>
<tr>
<th>Work item status</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Waiting</strong></td>
<td>The (sub) workflow is scheduled but its requested start has not yet been reached.</td>
</tr>
<tr>
<td><strong>Ready</strong></td>
<td>Temporary interim status</td>
</tr>
<tr>
<td><strong>In process</strong></td>
<td>Execution of the (sub) workflow has begun.</td>
</tr>
<tr>
<td><strong>Completed</strong></td>
<td>The end of the (sub) workflow has been reached.</td>
</tr>
</tbody>
</table>
| **Logically deleted** | Execution of the (sub) workflow is no longer required or meaningful. A status of a workflow changes to *logically deleted* in the following way:  
  - Intervention by an administrator. 
    An administrator can only set a workflow item to the *logically deleted* status if it has not yet reached the *completed* status.  
  - Termination in parallel processing branches 
    When the required number of processing paths has been executed in a fork, the work items in the other paths that have not yet reached the *completed* status are set to the *logically deleted* status. 
    A workflow set to the *logically deleted* status is recursively scanned for dialog and workflow work items (type W or F) that do not yet have the status "completed". These work items are then also set to the status *logically deleted*. 
    A work item changed from the *executed* status to the *logically deleted* status may already have caused database changes or other actions (generate event, send notification). These changes are *not* canceled automatically (compensated), but are recorded in the log. |
| **Errors**       | An error occurred in workflow control. |

Workflows and their statuses are **not** displayed in the Business Workplace. Information about the workflow can be accessed via the subordinate dialog work items.
Status Transitions of a Workflow Work Item

The diagram shows the possible status transitions.

**Status and Status Transitions of Work Items (F)**

The statuses of a workflow work item and the permissible status transitions always concern an entire workflow. A step (activity) in the workflow definition can reference both a task and another workflow. A workflow can therefore also contain subordinate subworkflow items.

**Comments**

**Transition from status waiting**
The work item manager automatically carries out the transition from the waiting status to the ready status when the requested start date/time of the workflow has been reached. A workflow system administrator can set a work item to the status ready manually.

**Transition from status ready**
The ready status is a temporary interim status because it only exists until the first work item of the workflow has been created.

**Transition to and from status in process**
The status of the workflow changes to in process as soon as the first work item of this workflow has been created. The workflow remains in this status until the entire workflow definition has been processed.

**Transition to incorrect status**
An error occurs during workflow control or coordination.

Role resolution for determining an agent does not return a result which can be used.
A workflow system administrator can intervene and set workflows with errors to the status *in process* or the status *logically deleted* (possibly after eliminating the error). If a workflow is incorrect, the responsible workflow system administrator specified either globally in Customizing or in the basic data of each workflow definition is notified by mail.

This status does *not* mean that a dialog work item of this workflow has the *incorrect* status.

**Transition to completed status**

A workflow is set to the status *completed* when the last step of the relevant workflow has been completed.
Background Work Item (Type B)

Definition
Work item that represents a single-step task at runtime whose execution does not require a dialog and, therefore, can be controlled automatically by the system.

Integration
Type B work items are not displayed in the Business Workplace.
## Status of a Background Work Item

The valid statuses for background work items (type B) are listed in the following table.

<table>
<thead>
<tr>
<th>Work item status</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Waiting</strong></td>
<td>The work item has been scheduled for its requested start. A work item has this status after it has been created until the requested start specified in the workflow definition is reached.</td>
</tr>
<tr>
<td><strong>Ready</strong></td>
<td>Temporary intermediate status of a background work item. The system calls the associated object method as soon as a background work item can be processed. The status of the background work item then changes to in process immediately.</td>
</tr>
<tr>
<td><strong>In process</strong></td>
<td>The work item is currently being processed. A work item also has this status if the method was left with a temporary exception. In this case, special Error Handling for Background Work Items with Temporary Errors [Seite 1388] is carried out. If the work item is waiting for its terminating event. The point at which processing is completed cannot be detected by the workflow system in this status.</td>
</tr>
<tr>
<td><strong>Completed</strong></td>
<td>The execution of the work item is completed. The result of the task represented by the work item is correct, i.e. the result modeled in the workflow definition.</td>
</tr>
<tr>
<td><strong>Logically deleted</strong></td>
<td>Execution of the work item with regard to the process logic is no longer meaningful or necessary for the process to continue. A work item changes to the logically deleted status in the following way: Termination in parallel processing branches When the required number of processing paths has been executed in a fork, the work items in the other paths that have not yet reached the completed status are automatically set to the logically deleted status. Intervention by an administrator The administrator can only set a work item to the logically deleted status if it has not yet reached the completed status and is not part of a higher-level workflow. A work item with the logically deleted status may have caused database changes or other actions (generate event, send notification). These changes are not canceled automatically (compensated).</td>
</tr>
<tr>
<td><strong>error</strong></td>
<td>Execution of the work item was terminated with an error.</td>
</tr>
</tbody>
</table>

In addition to the statuses given above, a work item can be **locked against execution**. This is possible in any status and involves an administrative function which is selected for a work item via the Change option.
Error Handling for Background Work Items

Use
Error handling for background work items must be carried out by the workflow system:

- because background work items (with errors) are not displayed in the Business Workplace and are therefore detected too late, if at all.
- because background work items that are temporarily incorrect, for which another execution attempt can (theoretically) be successful, cannot be restarted by a user.

(Temporarily incorrect work items are work items whose method was terminated with a temporary exception for which no subsequent step has been defined.)

Features
In Customizing for the workflow system, you can define how often the workflow system attempts to restart a temporarily incorrect work item. You can also define the interval between two repetition attempts and activate the monitoring function.

Monitoring is also activated as part of automatic Customizing [Extern] (Tools → Business Workflow → Development → Utilities → Customizing).

You can also specify the maximum number of repetition attempts (“repetition counter”) separately for each individual background step in the respective workflow definition. This setting overrides the value in Customizing if it is greater than zero.

If an error (method exception) occurs when background work items are executed by the workflow system, the system responds as follows depending on the type of exception and error modeling settings:

<table>
<thead>
<tr>
<th>In the workflow definition...</th>
<th>The exception is defined for the object method as a...</th>
<th>System or application error</th>
</tr>
</thead>
<tbody>
<tr>
<td>a subsequent step is modeled for the exception.</td>
<td>temporary error</td>
<td>completed and the modeled subsequent step is executed.</td>
</tr>
<tr>
<td></td>
<td>system or application error</td>
<td>completed and the modeled subsequent step is executed.</td>
</tr>
</tbody>
</table>
Error Handling for Background Work Items

| no subsequent step is modeled for the exception | The step is not yet completed. The respective work item retains the status *in process*. Background work items are restarted by the system. The number of repetition attempts is determined either by the repetition counter in the step definition or - if this is equal to zero - by the repetition counter set in Customizing. If all of the attempts are unsuccessful, the work item status changes to incorrect. | Workflow and work item assume the error status. |

Processing Incorrect Work Items

The workflow system sends a mail to the relevant system administrator for every background work item with the status *incorrect*.

Processing Work Items That Have Been Started

The workflow system determines all of the background work items that have been *in process* for longer than 30 minutes. An error message is then sent to the workflow system administrator for all of these background work items, since the system assumes that processing has been cancelled. However, this does not necessarily mean that an error has occurred.
Wait Step Work Item (Type E)

Definition
Work item that represents a wait step or a workflow at runtime, which is waiting for an event to occur.

Use
Type E work items are not displayed in the Business Workplace.
Work Item with Express Notification

Use
When the system creates a *work item* with priority 1, each of its *recipients* receives an express notification (dialog box with appropriate text) on the screen. The recipient can call the Business Workplace directly from the express message. Excluded agents do not receive a message.

Constraints
The system does not create an express message if
- the user processes the work item immediately due to *advance with immediate dialog*.
- the work item represents a *general task* that is not restricted to certain agents.
- the work item was forwarded.

The system only creates express notifications when a work item is created. An express notification is not created for the new recipients of priority 1 work items that are forwarded.

If one of the selected agents processes the work item, the other agents still receive an express message.

Features
Express messages are only sent for dialog steps as soon as the system has created them with the *ready* status. If the work item is created first with the *waiting* status because its requested start has not been reached yet, the express message is not sent until the status changes from *waiting* to *ready*.

Activities

How is the priority set?
The priority of a work item can be determined for steps that require a dialog with the user. It is defined in the tab page *Miscellaneous* of the step definition.
Workflow Settings

Use
You use the workflow settings to maintain the special workflow functions in the Business Workplace.

Features
The following functions are available:

- **Personal settings [Seite 1395]**
- **Display organizational assignment [Seite 1397]**
- **Refresh organizational environment [Seite 1397]**
- **Adopt substitution [Seite 1400]**
- **End substitution [Seite 1400]**
- **Maintain substitute [Seite 1400]**
- **Activate substitute [Seite 1400]**
- **Adopt view [Seite 1404]**
- **Exit view [Seite 1404]**

Activities
You can access the workflow settings within the [Business Workplace [Seite 1368]] via **Settings** → **Workflow settings**.
Personal Settings

Use
The personal settings for workflow enable you to adapt the runtime system to suit your requirements.

Features

Work item display

<table>
<thead>
<tr>
<th>Setting</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>User view with ActiveX (32-bit)</td>
<td>Both of these settings define the work item display as the default. Note that the ActiveX variant is only available on 32-bit platforms.</td>
</tr>
<tr>
<td>User view without ActiveX</td>
<td>This setting defines the technical work item display as the default.</td>
</tr>
<tr>
<td>Technical view</td>
<td></td>
</tr>
</tbody>
</table>

Workflow log

<table>
<thead>
<tr>
<th>Setting</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>User view with ActiveX (32-bit)</td>
<td>User view of the workflow log without any technical aspects. This view uses ActiveX controls and is therefore only available on 32-bit platforms.</td>
</tr>
<tr>
<td>User view without ActiveX</td>
<td>User view of the workflow log without any technical aspects. This view uses the hierarchical list display of the ABAP List Viewer.</td>
</tr>
<tr>
<td>Technical view</td>
<td>In addition to the main semantic information, the technical view of the workflow log also displays technical numbers and texts that may not be available in the logon language of the user. The technical view is intended primarily for system administrators.</td>
</tr>
</tbody>
</table>

Further settings

<table>
<thead>
<tr>
<th>Setting</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Display work item texts in logon language</td>
<td>All work item texts in the Business Workplace are always displayed in the user’s logon language. If the user’s logon language is different to the language in which the work item was created, additional database accesses are required that can delay display of the Business Workplace.</td>
</tr>
<tr>
<td>Enable forwarding of work items to several users</td>
<td>The work item can be forwarded to several agents. The term “agent” is used in a broad sense in this context. You specify the agent as an organizational object (organizational unit, job, position, user) when forwarding the work item. This organizational object can consist of several persons. For further information, refer to Forwarding.</td>
</tr>
<tr>
<td>(Double)-clicking on an object displays the object in the same window.</td>
<td>You can choose an object in the work item display, which is then displayed in the current</td>
</tr>
</tbody>
</table>
Personal Settings

<table>
<thead>
<tr>
<th>Setting</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Session window and replaces the work item display.</td>
<td>When you exit the object display, you return to the work item display. If you do not set the indicator, a new session is created to display the object.</td>
</tr>
<tr>
<td>This setting does not apply if you work with the SAP GUI for HTML.</td>
<td></td>
</tr>
<tr>
<td>No tips &amp; tricks in workplace</td>
<td>The Business Workplace [Seite 1368] includes tips and tricks for working with work items and workflows in the work item preview. If you set this indicator, the tips and tricks are not displayed.</td>
</tr>
<tr>
<td>No HTML in execution of decision tasks.</td>
<td>You use this indicator to decide whether the execution of decision tasks is displayed in HTML or not. Deactivating the HTML display may make sense, for example, if you have problems displaying HTML documents generally because of the settings in your local HTML browser.</td>
</tr>
</tbody>
</table>

Activities

To change the personal settings for workflow, you choose Settings → Workflow settings → Personal settings in the Business Workplace.

The settings are saved as user parameters and take effect the next time the work item display or workflow log is called.

You can change the standard values for these personal workflow settings in Customizing for SAP Business Workflow. These standard values are always used if users have not maintained settings of their own.
Organizational Assignment of a User

Use
The worklist displayed is formatted specifically for the user who is logged on. You can query this user's organizational assignment.

Prerequisites
The functionality described depends on a setting in the PD system table [Seite 1533].

Features
The organizational assignment of a user is buffered and read at each new logon or once a day (but not each time the Business Workplace is called). If the organizational assignment of a user changes while the user is logged on and must be updated, you choose Settings → Workflow settings → Refresh organizational environment in the Business Workplace.

Activities
To view current information on the assignment of the user logged on in the organizational plan of the enterprise, choose Settings → Workflow settings → Display organizational assignment in the Business Workplace.

The following information is displayed for the user logged on:
- The name maintained in the user address
- The organizational unit
- The position the user occupies
- The job describing the position

By double-clicking on an entry, you can display the staff assignments or the job description.
Substitutes for Workflows

Use

*SAP Business Workflow* implements a substitution arrangement to deal with a user's duties in their absence.

Prerequisites

The automatic appearance of work items in the substitute's worklist is dependent on a setting in the *PD* system table. It is possible to define which objects can be entered as position-related substitutes in Administration of Organizational Management.

Features

Who is involved?

Substitution always involves two users:

- One user specifies a substitute: User B
- The other user (the substitute) 'adopts' the substitution: User A

The substitute can process “foreign” work items for the duration of the substitution. It is irrelevant whether they are assigned to the underlying single-step tasks as a possible agent.

How does substitution work?

Substitution works in two ways:

3. User A adopts substitution for user B and for the duration of the substitution sees only the work items seen previously by B in their workflow inbox.

   User B must have entered user A as their substitute for a particular period without activating the substitution. During this period, A can adopt substitution for B at any time without having to confer with B.

   Instead of their own worklist, user A is displayed user B's worklist. B's name is in the column *Substitution for* in the worklist, as long as this column is included in the current configuration of B's Business Workplace.

4. User A automatically sees not only the work items previously seen by employee B, but also their own work items in their Business Workplace. This also applies to all work items generated for B in the future ("automatic forwarding").

   User B must enter user A as substitute and activate the substitution. No further action on the part of user A is required for this kind of substitution.

   A can continue working as usual. They see a Business Workplace to which B's work items are added dynamically. A can recognize these work items by the fact that B's name is in the column *Substitution for*, as long as this column is included in the current configuration.

In both cases, user A can perform operations on these "foreign" work items within the scope of the *substitute profile* assigned to them by B. It is irrelevant whether A is assigned to the underlying single-step tasks as a possible agent.

User B can continue working without any restrictions in both cases.
Activities

You can access substitute maintenance in the Business Workplace [Seite 1368] by choosing Settings → Workflow settings. For information on the individual functions, refer to Maintaining and Activating Substitutes [Seite 1400]
Maintaining and Activating Substitutes

Maintaining substitutes

2. In the Business Workplace [Seite 1368], choose Settings → Workflow settings → Maintain substitute.

The Maintain substitute dialog box appears.

To facilitate maintenance of the substitution, the Personal substitutes entry and the positions you occupy are displayed in the hierarchy on the Maintain substitutes screen.

You specify your substitute either as a personal substitute or as a position-related substitute.

- **Personal substitute**
  
  A personal substitute can see and execute all your work items, including those assigned to you via a personal agent assignment.
  
  You can only specify one other user as a personal substitute.

- **Position-related substitute**
  
  A position-related substitute can only see and execute the work items you have received on the basis of agent assignment at the level of position, job or organizational unit.
  
  You can enter another position or a user as a position-related substitute.

7. Position the cursor either on the entry Personal substitutes or on the relevant position, and select Create substitute.

You can maintain existing entries by double-clicking on the relevant entry.

If you want to maintain a position-related substitution, decide whether you want another position or another user as a substitute. Select either position or user as the substitute type and specify the number or the user name of the substitute.

8. Specify the validity period for the substitution on the detail screen for substitution.

Only within this period can the substitute adopt the substitution.

9. Specify a substitute profile in the dialog box Detail screen substitution.

Irrespective of whether you have created your substitute as a personal or position-related substitute, you can limit the scope of the work items displayed to your substitute by specifying a substitute profile.

10. Select the field Substitution active if applicable.

You must activate the substitution if you want work items to be visible for the substitute automatically from now on. If you do not activate the substitution, the substitute must adopt the substitution explicitly to see your work items.

11. Save your entries and exit substitute maintenance.
Activating substitutes
3. In the Business Workplace, choose Settings → Workflow settings → Activate substitute.
   The Activate substitute dialog box appears.
4. Select the substitutes that you want to activate, and choose the function Activate.

Deactivating substitutes
3. In the Business Workplace, choose Settings → Workflow settings → Activate substitute.
   The Activate substitute dialog box appears.
4. Select the substitutes that you want to deactivate, and choose the function Deactivate.

Adopting substitution
4. In the Business Workplace, choose Settings → Workflow settings → Adopt substitution.
   The Choose substitution dialog box appears.
5. Select the user(s) for which you want to adopt substitution.
6. Exit the dialog box.

Ending substitution
2. In the Business Workplace, choose Settings → Workflow settings → End substitution.
   The substitution is ended.
Views

Use
The Business Workplace [Seite 1368] provides various views on the work items displayed in the workflow inbox.

Using an appropriate view, a superior can "see" and process the work items of their employees.

Choosing a particular view also gives you the opportunity to see work items of other users in your workflow inbox and process them with full functionality, although the underlying tasks are not organizationally assigned to you.

Prerequisites
Views only require action by an employee: This employee chooses a view from a catalog of defined views. The employee must have the authorization required to choose a particular view.

Authorizations
To choose a particular view, you require a corresponding authorization based on authorization object S_WF_LVIEW.

To maintain a particular view, you must have the relevant authorization. This is an authorization based on the authorization object S_TABU.DIS to maintain tables for authorization group SWES.

Features
Views are always based on an evaluation path [Seite 1534] starting from the employee who wishes to adopt a view and leading to the employees whose inboxes could be viewed. The employee adopts a view by selecting another employee from the result list of the evaluation.

Activities
The activities associated with this function include:

- Adopting and Exiting Views [Seite 1404]
- Maintaining Views [Seite 1403]
Maintaining Views

Prerequisites
A view is always based on an evaluation path. This evaluation path describes which relationships are traced from the user who wants to adopt the view to the users whose Business Workplaces can be viewed.

You can use one of the evaluation paths available in the system. If there are no suitable evaluation paths, you can define an evaluation path of your own.

Different views can only be maintained as a customer setting.

Procedure

8. To call table maintenance for views, choose Tools → Business Workflow → Development → Definition tools → Worklist client → Maintain views.

This displays the screen Change View "View for Maintaining Views": Overview.

Maintaining views is an activity that is described in the Implementation Guide and can be performed in Customizing.

9. Create a new view. To do this, choose the function New entries.

10. Assign a unique name to the view.

11. Specify an evaluation path.

12. Describe the view with a long text.

13. Specify a start evaluation path.

This selection is optional. The start evaluation path is used to get an initial selection of objects, which is then evaluated further via the first evaluation path.

14. Select Save.
Adopting and Exiting Views

Procedure

Adopting views
4. To adopt a view as the standard view (= view on your own work items), choose Settings → Workflow settings → Adopt view in the Business Workplace.

You are now on the dialog box View: Choose Agent. Only the views for which you have an authorization are available.

5. Choose a view.

On the basis of the evaluation path defined for the view, the system selects the positions, organizational units or users connected to you.

6. From the result list of this selection, choose the object whose workflow inbox you want to view.

Exiting views
A view is only active while the Business Workplace is displayed. The next time you call the Business Workplace, you are asked if you wish to adopt the view previously set.

To return to the standard view when working in the workflow inbox of the Business Workplace, choose Settings → Workflow settings → Exit view.
Dynamic Columns for the Business Workplace

Use
Up to 6 columns in the workflow inbox of the Business Workplace [Seite 1368] can be filled on a task-specific basis with contents that are determined dynamically at runtime. The standard functions for filtering, sorting, and grouping are available for these columns. Please compare with Selectable Columns for the Business Workplace [Seite 1406].

Features
If you want to include one of the “dynamic columns” into the workflow inbox display, you must specify an element from the task container for each task, from which the content of the column is established at runtime.

Work items that belong to different tasks are then also displayed with different information. Work items that belong to tasks for which this functionality is not used are displayed with a blank entry.

Activities
The column contents are maintained via Tools → Business Workflow → Development → Definition tools → Worklist client → Dynamic columns for worklist.

You can also define the column headings. These headings are displayed if all of the work items displayed in the Business Workplace refer to the same task.
Selectable Columns for the Business Workplace: Workflow

The columns displayed essentially determine the appearance and information content of the workflow inbox of the Business Workplace. Detailed knowledge of the columns is also required to make full use of the filter and grouping criteria.

You can determine the selection of columns via display variants. The following columns are available:

<table>
<thead>
<tr>
<th>Column</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>Work item is executable</td>
<td>Indicator denoting that the work item is executable.</td>
</tr>
<tr>
<td>Work item title</td>
<td>Title of work item.</td>
</tr>
<tr>
<td>Status indicator</td>
<td>Status of work item.</td>
</tr>
<tr>
<td>Creation date</td>
<td>Date when the work item was created with the status ready or waiting for the first time. A work item is only created with status waiting if a requested start was declared for the work item and the work item is created before the requested start.</td>
</tr>
<tr>
<td>Creation time</td>
<td>Creation time of a work item.</td>
</tr>
<tr>
<td>Priority</td>
<td>Priority of the work item.</td>
</tr>
<tr>
<td>Attachments exist</td>
<td>Attachments are shown with the symbol.</td>
</tr>
<tr>
<td>End of processing must be confirmed</td>
<td>Indicator denoting that the end of processing must be confirmed explicitly.</td>
</tr>
<tr>
<td>Work item overdue</td>
<td>Indicator denoting that a deadline has been missed for the work item.</td>
</tr>
<tr>
<td>ID</td>
<td>Unique number of a work item, which is assigned internally by the system.</td>
</tr>
<tr>
<td>Type</td>
<td>Type of work item.</td>
</tr>
<tr>
<td>Task</td>
<td>Identification for the task represented by the work item (for example TS00008323).</td>
</tr>
<tr>
<td>Technical status</td>
<td>Status of work item.</td>
</tr>
<tr>
<td>Work item type</td>
<td>Type of work item.</td>
</tr>
<tr>
<td>Task name</td>
<td>Name</td>
</tr>
<tr>
<td>Work item status text</td>
<td>Status of work item.</td>
</tr>
<tr>
<td>Deadline status</td>
<td>The deadline status specifies whether one of the deadlines has been missed. The possible values in this column are therefore:</td>
</tr>
<tr>
<td></td>
<td>• None</td>
</tr>
<tr>
<td></td>
<td>• Latest start</td>
</tr>
<tr>
<td></td>
<td>• Requested end</td>
</tr>
<tr>
<td></td>
<td>• Latest end</td>
</tr>
<tr>
<td></td>
<td>• Other</td>
</tr>
<tr>
<td>Current agent</td>
<td>Name of the user who last reserved or processed the work item.</td>
</tr>
</tbody>
</table>
## Selectable Columns for the Business Workplace: Workflow

<table>
<thead>
<tr>
<th>Column</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Latest end date</strong></td>
<td>Latest end of work item. The end is reached when the work item assumes the status <em>completed</em>.</td>
</tr>
<tr>
<td><strong>Latest end time</strong></td>
<td>Latest end time of a work item.</td>
</tr>
<tr>
<td><strong>Forwarder</strong></td>
<td>Name of the party who forwarded the work item.</td>
</tr>
<tr>
<td><strong>Substitution for</strong></td>
<td>Name of the substituted party whose work item is displayed</td>
</tr>
<tr>
<td><strong>Work item content</strong></td>
<td>Column in which the default attribute of the object referenced in the container element _WI_Object_ID is displayed.</td>
</tr>
<tr>
<td><strong>Group object</strong></td>
<td>Column in which the default attribute of the object referenced in the container element _WI_Group_ID is displayed.</td>
</tr>
<tr>
<td><strong>Execution can be rejected</strong></td>
<td>Indicator denoting whether execution of the work item can be rejected (X).</td>
</tr>
<tr>
<td><strong>Dynamic columns</strong></td>
<td>Refer to [Dynamic Columns for the Business Workplace](Seite 1405).</td>
</tr>
</tbody>
</table>
Workflow Inbox

Use
The worklist of the user currently logged on to the Business Workplace is displayed in the workflow inbox.

Integration
As is the case for the workflow resubmissions [Seite 1443] and the workflow outbox [Seite 1441], the workflow inbox is an integral part of the Business Workplace.

Features

Views in the workflow inbox
A user's worklist can be displayed as an overview or according to the following grouping criteria:

<table>
<thead>
<tr>
<th>Grouping</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Grouped according to task</td>
<td>The work items are grouped according to the tasks to which they belong.</td>
</tr>
<tr>
<td>Grouped according to content</td>
<td>The work items are grouped according to the object instances to which they belong.</td>
</tr>
<tr>
<td>Grouped according to content type</td>
<td>The work items are grouped according to the object types to which they belong.</td>
</tr>
<tr>
<td>Grouped according to sort key</td>
<td>The work items are grouped according to sort keys. Please refer to Grouping According to Sort Keys [Seite 1432].</td>
</tr>
</tbody>
</table>

You can also choose from the following views:
- Overdue entries
- Deadline messages
- Incorrect entries

Workflow Functions
The screen area in which the worklist is displayed has a toolbar containing the workflow functions and functions for configuring the list. For information about these functions, refer to the documentation on the ABAP List Viewer [Extern].

The workflow functions can be accessed in the Business Workplace via the toolbar or the relevant context menu (right-hand mouse button). The following functions can be executed on work items:

- Update
  This function updates the worklist [Extern] of the user currently logged on to the Business Workplace.

- Execute
  To be executed, a work item must have either the status ready or the status in process.
  When a dialog work item [Extern] is executed, the object method to which the single-step task for this work item refers is carried out.
  When a missed deadline work item [Extern] is executed, the most important information on the monitored work item is displayed.

- Display work item
  This function goes directly to the work item display [Seite 1411]
• **Reserve** (dialog work items only)

  This reserves a work item for execution by the end user in question. The work item must have the status *ready*. This work item is then no longer visible to the other recipients who could previously see it in status *ready*. The status of the work item changes from *ready* to *reserved*.

• **Replace** (dialog work items only)

  Replacing releases the work item from its reserved status and makes it available again to all of the original recipients. The status of the work item changes from *reserved* back to *ready*. The status of the work item must be *reserved* before it can be replaced.

• **Forward**

  Forwarding [Seite 1435] passes a work item on to another employee for execution.

• **Resubmit**

  If a user chooses this function, the selected work item is placed in workflow resubmissions [Seite 1443].

• **Display workflow log**

  This function displays the workflow log [Seite 1420].

• **Manage attachments** … (Functions for attachment management [Seite 1429], dialog work items only)

  - Display attachments
  - Create attachments
  - Change attachments
  - Delete attachments

• **More functions**

  - Set to ‘Done’ [Seite 1440]
  - Reject execution [Seite 1430]
  - Execute together [Seite 1431]
  - Change priority [Seite 1433]
  - Send mail [Seite 1425]
  - Change work item [Seite 1438]

• **Environment**

  - Display objects in workflow (dialog work items only)
  - Display workflow relationships [Seite 1436]
  - Start Workflow [Seite 1450]
  - Create link [Seite 1434]
Activities

You can access the workflow inbox in the tree on the left in the Business Workplace via Workplace → Inbox.
Work Item Display: Standard View

Use
The objective of the work item display is to display all the information and activities that are relevant to an end user in the environment of the displayed work item in a clear and concise manner. The work item display supports the agent not only in the processing of the current work item but also in the compilation of an activity list, which then functions as the agent's personal worklist.

Integration
A technical work item display [Seite 1418] is available for work items of all other types. You can make this technical work item display standard for dialog work items as well. The standard view of the work item display can be shown with or without ActiveX. You make this setting for the work item display variant in your personal workflow settings [Seite 1395].

Features
The work item display is primarily designed for displaying dialog work items [Extern]. It contains details about deadlines, statuses, agents, attachments and linked objects for a work item. The work item display also enables an end user - providing they have the relevant authorization - to compile an activity list as their personal worklist.

The work item display has three tab pages (Basic data, Activities, and Available objects).
- Tab page Basic data [Seite 1414]
- Tab page Activities [Seite 1415]
- Tab page Available objects [Seite 1416]
- Customer-defined tab page for work item display [Extern]

You can define another tab page, which is then displayed as the first tab page when the work item display is called.

Application toolbar functions
- Execute
  To be executed, a work item must have either the status ready or the status in process.

  When a dialog work item is executed, the object method to which the single-step task for this work item refers is carried out.

  When missed deadline work items [Extern] are executed, the most important information on the monitored work item is displayed.

- Display last message
  The return code that was returned to the workflow system after the object method was executed can be retrieved for processed work items using the Messages function.

- Forward
  Forwarding [Seite 1435] passes a work item on to another employee for execution.

- Resubmit
If a user chooses this function, the selected work item is placed in workflow resubmissions [Seite 1443].

- **Change priority**
  Refer to Changing Priorities [Seite 1433].

- **Change deadlines**
  Refer to Changing Deadlines [Seite 1428].

- **Display/create/change attachments**
  For attachments, refer to Attachment Management [Seite 1429].

- **Reserve** (dialog work items only)
  This reserves a work item for execution by the end user in question. The work item must have the status **ready**. This work item is then no longer visible to the other recipients who could previously see it in status **ready**. The status of the work item changes from **ready** to **reserved**.

- **Replace** (dialog work items only)
  Replacing releases the work item from its reserved status and makes it available again to all of the original recipients. The status of the work item changes from **reserved** back to **ready**. The status of the work item must be **reserved** before it can be replaced.

- **Mail**
  Refer to Send Mail [Seite 1425].

- **Display workflow log**
  This function displays the workflow log [Seite 1420].

- **Display graphical workflow log**
  This function displays the graphical workflow log [Seite 1424].

**Additional functions in the menus**

**Work Item**

- **End resubmission**
  The work item is put back into the workflow inbox. It has the status **reserved**.

- **Create link** [Seite 1434]
- **Reject execution** [Seite 1430]
- **Set to ‘Done’** [Seite 1440]

**Goto**

- **Methods**
  You can use this function to execute the defined secondary methods [Extern] of the work item displayed from the work item display.
  A prerequisite for this is that the work item represents an activity that is part of a workflow. At least one secondary method must be defined for this activity.
• **Workflow description**
  
  The description text of the superordinate multistep task, the "process description", is displayed.
  The work item must be part of a workflow and a description text must be maintained for the workflow.

• **Technical work item display [Seite 1418]**

**Extras**

• **Displaying tasks**
  
  This function can be used to display the definition of the task represented by the work item. Refer to Definition of a Single-Step Task [Seite 1175] and Definition of a Multistep Task [Seite 1194].

• **Technical data**
  
  Technical data about the work item, for example its ID, its texts or the ID of the superordinate work items are displayed.

• **Organizational Assignment [Seite 1397]**

• **Displaying Agents [Seite 1427]**

**Activities**

You can access the work item display by:

• Selecting a work item in the Business Workplace and choosing 📌.

• Double-clicking one of the entries for a step in the workflow log.

• Selecting an entry that does not represent a workflow in the work item selection [Seite 1409] hit list displayed. (If you choose a workflow, the workflow log is displayed.)
Tab Page Basic Data

Use
The information on this tab page of the work item display [Seite 1411] is mostly self-explanatory. Except for the priority, you cannot make any changes here.

Features

Deadlines
These are the deadlines monitored by the runtime system. (Deadlines that are not set are displayed without a date.)
- Start by (latest start)
- End by (latest end)
Depending on whether the work item represents a step in the workflow or a task, the deadlines were either specified when this step was defined in the workflow definition or when the task was started online.
To display all of the deadlines of the work item, choose Work item → Deadlines.

Further Information
- Forwarded by
  If the work item was forwarded to you, the name of the forwarder is entered here.
- Priority
  The priority of a work item is derived from the definition of this step in the workflow definition. The priority is used as a sort criterion for positioning the work item in the Business Workplace.
  The priority can be changed here.
- Status
  The current processing status is expressed by the work item status [Seite 1373].
- Creation date (created on) and processing date (processed from)
  These are the actual dates and times (when the work item was created and when processing was started).
Using the Messages function, you can display the return codes for processed work items, which were returned to the workflow system after execution of the object method.

Work Item Description
A description of the work item to be executed is provided at the bottom left of this tab page.
The task description is entered in the task definition. It is used for information purposes and generally contains instructions and recommendations on processing the work item displayed.

Attachments
The titles of all the attachments added to this work item or, if the work item is part of a workflow, to the preceding work items are shown in the lower right of this tab page.
Tab Page Activities

Use
The Activities (not yet processed) list contains all the activities that are relevant for processing this work item. The tab page Activities is part of the work item display [Seite 1411].

Features
The work item text of the task represented by the work item is generally at the start of the list ("main activity"). Once this activity has been processed (and completion of processing has been confirmed, if necessary), the status of the work item changes to completed. No other actions can then be carried out in the work item display.

Activities
You can extend the activity list and in this way create a worklist. The activities added represent your "personal worklist" as end user (agent).

<table>
<thead>
<tr>
<th>Function</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Done</strong></td>
<td>An activity selected from the list is reported to be done. This functionality cannot be executed on the leading activity (first line) of this work item. The agent is responsible for reporting that an activity is done. The actual processing is not checked.</td>
</tr>
<tr>
<td><strong>Execute</strong></td>
<td>An activity selected from the list is executed. The main activity (first line) of the work item can also be executed via the menu path Work item → Execute.</td>
</tr>
</tbody>
</table>
| **Create** | Another activity is added to the list. This activity can entail:
  - Executing a method on an existing object (created invoice, created material, etc.)
  - Creating a new object (text, etc.)
When you create an activity, you are given step-by-step support in dialog by a "wizard". |
| **Delete** | An activity is deleted from the list. |

Scrolling in the activity list: The activity list contains extensive information on each activity, which you can view by scrolling to the right. This information includes the following:
- Description
- Creator of the activity with date and time
- Actual agent of the activity with date and time
- Method and object type
Tab Page Available Objects

Use
All objects that are related to the work item are displayed in the list of available objects. These objects are stored in container elements of the task container as object references. You can display these objects or add new objects to the container element.

There are:
- Ad hoc objects
  - Objects added to a work item in this or one of the preceding steps of the workflow (container element _Adhoc_Objects of the task container)
- Attachment objects
  - Documents added to a work item in this or one of the preceding steps of the workflow (container element _Attach_Objects of the task container).
- Process objects
  - The object currently being processed (container element _WI_Object_ID of the task container)
  - The object added for grouping purposes (container element _WI_Group_ID of the task container)
  - Objects that are referenced in other elements of the task container

The tab page Available objects is part of the work item display [Seite 1411].

Features

Displaying objects with their default attributes
Each object referenced in the work item container is displayed with its default attribute [Extern]. The default method [Extern] of each object can be executed upon request. If no default attribute was defined for the object type, the key fields of the object are displayed.

Adding objects
You can extend and process the list of objects. The main purpose of this is to make the relevant information available to the agents of the subsequent steps in the workflow as well. Only object types [Extern] that support the IFFIND interface can be selected. You identify an actual object [Extern] of this type by specifying its key fields [Extern].

Activities
To execute the functions displayed, proceed as follows:

<table>
<thead>
<tr>
<th>Function</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Create</td>
<td>An object is added to the list. When you add an object, you are given step-by-step support in dialog by a &quot;wizard&quot;.</td>
</tr>
<tr>
<td>Display</td>
<td>The default method of an object selected from the list is executed.</td>
</tr>
<tr>
<td>Remove</td>
<td>An object is deleted from the list.</td>
</tr>
</tbody>
</table>
Work Item Display: Technical View

Use
The technical work item display shows all information from the environment of the work item displayed. The technical work item display is intended in particular for workflow system administrators.

The data is always formatted and displayed in a similar way irrespective of the work item type. You should note the **work item type** shown on the screen.

Features

**General**

**Work item information**
- Information derived from the definition of the relevant task: **Work item type**, **work item text**, *way in which processing is completed*.
- **Status** of the work item as current processing information
- **Actual agent** (available after dialog work items have been executed).

In the case of processed work items, the **Messages** function can be used to display the return codes returned to the workflow system after execution of the object method.

**Deadline data**
Here you can find the **current dates/times** (creation date/time of work item and date/time processing started and ended) as well as the deadlines that are monitored by the runtime system (requested and latest start and end deadlines).

A monitored deadline shown with red is in the past. If the symbol is displayed as well, a defined action was triggered.

**Description**
This description displayed here is taken from the task definition. It is used to inform the selected agents and contains instructions and recommendations on processing the work item displayed.

If you have selected a work item using the **work item selection** or the workflow outbox rather than your workflow inbox, you can execute it provided you are one of the possible agents.

You do not need to be one of the recipients in this case.

**Additional functions**
As well as the functions available in the standard view of the **work item display**, the following **additional** functions are also available:

- **Display/create/delete object**

  Each object referenced in the work item container is displayed with its **default attribute**. The **default method** of each object can be executed upon request. If no default attribute is defined for the object type, the key fields of the object are displayed instead.
You can extend and process the list of objects. The main purpose of this is to make the relevant information available to the agents of the subsequent steps in the workflow as well.

Only object types [Extern] that support the IFFIND interface can be selected. You identify an actual object [Extern] of this type by specifying its key fields [Extern].

- **Type-specific data**
  
  Type-specific data only applies to work queue and wait step work items.

  For work queue work items (type A), the objects and tasks contained in the work queue are listed.

  For wait step work items (type E), the number of events expected (information taken from the workflow definition) and the number of events that have already occurred are specified.

- **Execute for testing**
  
  In order to make it possible to check the runtime system's program execution (work item manager and workflow manager) in debugging mode after execution of an object method, internal communication within the workflow system cannot take place asynchronously. To achieve this, execute the work item for test purposes. Enter /h in the command field first to go to the debugging mode.

- **Monitored work item**
  
  The work item (type F or W) whose deadlines or completion are monitored by the workflow system is displayed.

  This function is only possible and active from missed deadline work items [Extern]. This function displays the work item. Full display functionality, including all navigation and change options, is also available here for this work item.

- **Instance linkage**
  
  You go to the relevant line of the instance linkage table, in which the expected event (identified by object type and event ID) and the object (identified by object reference) are specified.

  This function is only possible and active for those work items that wait for an event that completes them. This may apply to dialog work items (type W) and background work items (type B). This does apply to wait step work items (type E).

- **Container**
  
  This function displays the content of the task container.

- **Change work item [Seite 1438]**

**Activities**

You can go to the technical view of the work item display by choosing Goto → Technical work item display in the work item display or by having this display variant as a presetting in your personal workflow settings [Seite 1395].
Workflow Log: Standard View

Use
The workflow log [Extern] enables you to format all the information that is generated or collected during the execution of one active business process in one central location. The standard view described here is intended, in particular, for agents and process controllers who want to get an overview of the steps processed so far.

Prerequisites
To use the view described here, you must have set a view with ActiveX in your personal workflow settings [Seite 1395].

Integration
Other options for displaying the workflow log are:
- Workflow Log: Technical View [Seite 1422]
- Graphical Workflow Log [Seite 1424]

Features

Application toolbar functions
The following functions are available on the application toolbar:
- Update
- List with technical details
  This function takes you to the technical view of the workflow log.
- Graphical workflow log
- Personal workflow settings
  You can use this function to maintain your personal workflow settings.

Tab pages
The system shows the various views on the workflow log on different tab pages.

Tab page Workflow Chronicle ( , what was processed when?)
The tab page Workflow Chronicle shows a hierarchical display of all steps in the workflow, which have been processed so far or are due to be processed. If the workflow has a subworkflow structure, the subworkflows are also displayed.
The Details function ( symbol) lists the following information about each step in the lower part of the screen:
- Who carried out what detailed actions for these work items and with what results.
- When these actions were carried out.
- Which objects were involved.
The Agents function ( symbol) displays the selected and possible agents of a step.
The Graphic function ( symbol) displays the graphical workflow log.
Tab page Workflow Agents (📜, who processed what?)
The tab page Workflow Agents shows the employees involved in this workflow up to now. The following is displayed for each employee:
- What action was carried out in what step.
- When this action was carried out.
- Which objects were involved.
This view shows how an employee was involved in the execution of a workflow.

Tab page Workflow Objects (🌈, what was processed?)
The tab page Workflow Objects lists the objects related to the workflow or addressed up to now in the execution of the workflow. These objects include:
- The “leading” object of the workflow.
- Any attachments and objects added in the individual steps of the workflow.
The following is displayed for each object:
- Who carried out what detailed action for what task.
- When this action was carried out.
This view shows what information was generated and processed, and how.

Information at the click of a mouse
You can view all the information provided in the workflow log using the mouse.
You can also go to the [work item display][1411] for each dialog step. You can display address data for agents as well as the contents of work item attachments or the result of actions that have been executed.

Activities
You can access the workflow log from the work item display or the [Business Workplace][1368] via the 📌 icon.
Workflow Log: Technical View

Use
The workflow log [Extern] enables you to format all the information that is generated or collected during the execution of one active business process in one central location. The view described here, in particular the technical view described below, is intended for workflow system administrators.

Integration
Other options for displaying the workflow log are:
- Workflow Log: Standard View [Seite 1420]
- Graphical Workflow Log [Seite 1424]

Prerequisites
To use the view described here, you must have set the view without ActiveX in your personal workflow settings [Seite 1395]. If you have chosen technical view in your settings, the standard view with technical details is displayed (see below). Otherwise, the two workflow log views are identical.

Features
The system uses a two-level, hierarchical list from the ABAP List Viewer [Extern] to display the various workflow log views. You can adapt the appearance of the list to suit your requirements using display variants. By single-clicking on an entry or a symbol, you can then branch to the workflow container or the work item display [Seite 1411], for example.

The following views are available:
- **Workflow chronicle**
  The first level of the Workflow chronicle view shows all the workflow steps that have already been processed or are currently pending. If the workflow has a subworkflow structure, the subworkflows are also displayed.
  The second level (detail view) shows the following for each step:
  - Who carried out what detailed actions for these work items and with what results.
  - When these actions were carried out.
  - Which objects were involved.
  This view is used to determine what activities were carried out in a workflow and in what order.

- **Workflow agents**
  The first level of the Workflow agents view shows the employees involved in this workflow up to now.
  The second level (detail view) shows the following for each employee:
  - What action was carried out in what step.
  - When this action was carried out.
• Which objects were involved.
  This view shows how an employee was involved in the execution of a workflow.

• **Workflow objects**
  The first level of the *Workflow objects* view lists the objects that are related to the workflow or that have been addressed up to now during execution of the workflow. These objects include:
  - The “leading” object of the workflow.
  - Attachments and objects added in the individual workflow steps.

The second level shows the following for each object:
  - Who carried out what detailed action for what task.
  - When this action was carried out.

This view shows what information was generated and processed, and how.

• **... With technical details (technical view)**
  The *technical view* shows technical control information for execution of a workflow, as required by workflow administrators, for instance.

  Based on the workflow chronicle, the *technical view* shows technical nodes and control structures, and makes additional data available, such as container elements ( ), employee data ( ), and workflow data ( ).

  The status of the work item is also displayed.

• **... With subworkflow structure**
  Here you can choose whether or not to display any subworkflows and their structure.

• **... With error indicators**
  If you activate the function *View with error indicators*, errors are marked in the log with the symbol. The standard indicator is .

**Activities**
You can access the workflow log from the workflow inbox of the *Business Workplace [Seite 1368]* via the symbol, or using the *context menu* (click the right-hand mouse button).
You can maintain the individual views (such as the technical view) within the workflow log via *Views* or *Views → Other views*. 
**Graphical Workflow Log**

**Use**
The workflow log enables you to format all the information that is generated or collected during the execution of one active business process in one central location.

**Integration**
Other options for displaying the workflow log are:
- Workflow Log: Technical View
- Workflow Log: Standard View

**Features**
The graphical workflow log adds to the textual information. The workflow steps already processed are shown with in a graphical representation of the workflow definition. You can see at a glance which “route” a workflow instance has taken and which activities are processed in parallel to your own within a business process. Unlike the text version of the workflow log, the graphical workflow log also shows the subsequent flow of a workflow instance.
The screen of the graphical workflow log is split into the workflow area (left) and the overview area (right).
The following functions are available:
- Refresh
  Refreshes the display.
- Display node
  If you select an executed step and choose this function, the system goes to the technical work item display.
  If the step has not yet been executed, the relevant activity is displayed.
- Align...
  The workflow is centered within the workflow area.
- Zoom in
  The size of the workflow is increased within the workflow area.
- Zoom out
  The size of the workflow is decreased within the workflow area.
- Complete view
  The whole of the workflow is shown within the workflow area.

For other functions, refer to the documentation on the Workflow Builder.

**Activities**
You can call the graphical workflow log from within the workflow log via.
Send Mail

Purpose
You want to send a mail to another user in connection with a work item that requires processing. The work item being processed is therefore also to be made accessible to this user for information purposes. The Send mail function enables you to send mails to any recipients with a text which can be entered freely. These mails are automatically linked to your work item in such a way that when the mail is executed by the recipient the work item is displayed.

Process Flow

Procedure as sender of mail
4. In the Business Workplace's workflow inbox, position the cursor on a work item and choose Other functions → Send mail.
   If you are already in the work item display, choose Work item → Send mail.
5. Enter the text of the mail.
6. Send the mail to any recipients who are available as mail addressees
   The mail text is added to the work item as an attachment. The work item remains in your Business Workplace with the same status.
You can still complete the work item despite the fact that you have sent a mail. Since this makes the mail which has been sent superfluous, the recipient of the mail receives a second mail informing him/her that the (first) mail no longer needs to be dealt with.

Reply ends resubmission
Optional: If you do not want to process the work item until the mail has been replied to, define a resubmission for the work item. To do this, position the cursor on the entry, choose Resubmit and enter a date in the future.
The work item is resubmitted irrespective of this date and appears in your workflow inbox when the reply to the mail is received.

Procedure as recipient of a mail
You receive a mail that can be executed.
5. Read the mail.
6. Execute the mail, if applicable. The work item referred to by the sender in their mail is displayed. For further information on the object to be processed, choose Goto → Object to display the default attribute or execute the default method (generally Display) for the processed object.
   You can execute this work item if you are one of the possible agents of this task.
7. Reply to the mail. To do this, choose Document → Reply or Document → Reply w/reference when the mail is displayed. Then enter your reply and save your entries.
8. Choose Document → Send. On the send screen, the work item to which the mail referred is entered as the "recipient". Do not change this entry. Send the reply to the proposed recipient.
Result
The reply sent ends a resubmission which may have been defined for the recipient work item. This work item is appended as an (additional) attachment and can be read as such by the original user and taken into account.
Display Agents

Use
This function displays information about the agents of the work item.

Features
The system displays the following agents for the work item:
- Recipient [Extern]
- Possible agents [Extern]
- Excluded agents [Extern]
You can choose a compressed display - containing only the user names (users only function) - or a complete display with additional information on the relationships used by the system to determine the agents (overall view function). You can also display the organizational assignment of each user.
Note that this information is only available for work items of types for which an agent is logical and necessary.

Activities
You can display the agents from the work item display [Seite 1411] by choosing Goto → Agents → ...
Changing Deadlines

Use
You use this function if you want to change the deadlines for a work item at workflow runtime. This dialog box displays all the deadlines of work item processing. Please also refer to Current Dates/ Times of a Work Item [Extern].

Features
Depending on whether the work item represents a step in the workflow or a single-step task, the deadlines were either specified when this step was defined in the workflow definition or when the single-step task was started in dialog. The dialog box is split into the following sections:

Deadlines
- Start by (latest start)
- End by (latest end)

Planned deadlines
- Start by (requested start)
- End by (requested end)

A monitored deadline shown in blue is in the past. If is also displayed, the appropriate action has been initiated. This generally involves informing the deadline recipient.

Actual dates/times
- Created on: Creation date/time of the work item
  (Technically: The work item is created with the status ready or - if the requested start date has not yet been reached - with the status waiting).
- Processed from: Start of processing
  (Technically: Transition of the work item to the status in process).
- Completed on: End of processing or date when set to 'done').
  (Technically: Transition of the work item to the status completed.)

Activities
To execute this function, choose in the work item display [Seite 1411].
Attachment Management

Use

One or more attachments can be assigned to each work item that appears in the Business Workplace's workflow inbox. Attachments are documents written either with a SAPscript editor (document classes RAW, SCR) or with a PC application (document classes DOC, URL, PPT, XLS, PDF, ...) and then imported. You can enter new documents as attachments or create attachments from existing files.

Features

General

The attachment is automatically

- Added to the work item container
- Added to the container of the superordinate workflow
- Added to the containers of the subsequent work items in the workflow

You can define default documents for the individual document classes. For further information, refer to Default Documents [Extern].

Attachments can be displayed by the recipients of the subsequent steps. But they cannot be changed and, therefore, have a document character.

A superior who is to make a decision on releasing a budget can enter an attachment justifying their decision. The selected agents of the subsequent steps can display this attachment.

If a work item has attachments, this is indicated by a symbol in the Attachments column in the Business Workplace. You can also execute the function for processing an attachment by double-clicking in this column (column header AT). If an attachment already exists, it is displayed.

Activities

You can access attachment maintenance in the Business Workplace by choosing or the relevant context menu (right-hand mouse button).
Reject Execution

Use
You can use this function if you need to reject execution of a work item for business or technical reasons. This function is only available for work items of type W.

The table entry to be processed already exists or the material whose master data is to be changed is no longer used.

Prerequisites
This function is only available if the property processing rejectable has been selected for the related activity in the workflow definition.

Features
Processing of the work item is terminated with the reject execution function. The subsequent steps defined in the workflow definition are executed. Do not use this function if you do not want to or cannot process the work item for personal reasons (not responsible, not competent). In this case, replace the reserved work item or forward it.

Activities
You can access this function in the Business Workplace [Seite 1368] by choosing Other functions or the work item display (Work item → Other functions → Reject processing).
Execute Together

Use
You can select various work items and then execute them together. This function is only available for work items of type W.

The work items do not necessarily have to belong to the same workflow.

Prerequisites
The work items must refer to the same single-step task.
For information about other prerequisites for this function, in particular with regard to the underlying method, refer to Creating an "Execute Together" Method [Extern].

Features
Only the first of the selected work items is proposed for execution. The entries you make here also apply to the other work items to be executed together.

Activities
Select the work items that you want to execute together. Then choose Execute together via Other functions... in the Business Workplace [Seite 1368].
Grouping According to Sort Key

Use
Each work item carries the two container elements _WI_Object_ID and _WI_Group_ID in its container. Both elements have been defined to hold an object reference.

- The container element _WI_Object_ID automatically contains the reference to the object to be processed in the work item.
- The container element _WI_Group_ID can contain an object reference, which must be assigned to this container element in a binding or via initial value assignment.

The object reference assigned via _WI_Group_ID is generally not identical to _WI_Object_ID, but is derived from the work item execution environment. It is used to group work items that refer to different objects or object types but are nevertheless connected.

Features

Display default attributes in workflow inbox
The default attributes [Extern] of the objects referenced in _WI_Object_ID and _WI_Group_ID are available in the Business Workplace's workflow inbox [Seite 1408] under the column headers Object or Group for grouping, sorting and filtering purposes.

Material master data is processed in several instances of a workflow. The container element _WI_Object_ID in the containers of the individual work items therefore always contains the reference to the object of type BUS1001 (material) to be processed.

The container element _WI_Group_ID is assigned the expression &Material.Labor& (laboratory/drawing office for material as object reference) in the relevant steps of the workflow definition. The default attribute of the laboratory is therefore available as a group.

Default methods
The default method [Extern] of the objects referenced in _WI_Object_ID and _WI_Group_ID is executed by double-clicking in the Object or Group column.
Change Priority

Use
The priority of a work item is a measure of its urgency.
It can be used as a sort criterion for organizing the workflow inbox. This function is only available for
dialog work items.

Features
End users can display and change the priority of a work item. The priority is between 1 (highest) and 9
(lowest).
The change in priority can also be passed on automatically to the superordinate workflow and then to all
work items created subsequently. The Pass on priority to subsequent steps indicator must be selected for
this to take effect.

If a higher priority (lower number) is defined in the workflow definition for one of the
subsequent steps, it is not changed.

Activities
You can access the Change priority function in the Business Workplace by choosing Other
functions...
Create Link

Use
By storing links to work items in folders [Extern], you can organize your work effectively using a personal folder hierarchy.

Features
The work item is added to a personal or shared folder as a link. This work item can be displayed from this folder and executed by its recipients. Work items can have an unlimited number of links.

Activities
You can access this function in the Business Workplace [Seite 1368] by choosing Environment...
Forward Work Item

Use
Forwarding passes a work item on to another user for execution.

Integration
The range of users to whom a work item can be forwarded is determined from the task definition as follows:

<table>
<thead>
<tr>
<th>Task definition:</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Task is a general task [Extern]</td>
<td>Work item may be forwarded to all users.</td>
</tr>
<tr>
<td>The indicator General forwarding allowed is set for the task</td>
<td>Work item may be forwarded to all users.</td>
</tr>
<tr>
<td>The indicator General forwarding not allowed is set for the task</td>
<td>Work item may only be forwarded to the recipients [Extern] of the task.</td>
</tr>
</tbody>
</table>

Features
The user has the following search options to select the new recipient:

- The F4 input help can be used to determine the new recipient by performing a generic search for name components.
- The structure search can be used to determine the new recipient using the graphical display of the organizational plan.

The new recipient does not have to be an actual user. (The work item can also be forwarded to an organizational unit or a job.)

Activities
You can access this function in the Business Workplace [Seite 1368] by choosing ✂️.
Workflow Relationships

Use
You can use this function to establish the work items in which a particular object is being processed. The following work item types are taken into account:
- **Dialog work items** (type W)
- **Background work items** (type B)
- **Workflow work items** (type F)

The list displayed is purely runtime information.

Prerequisites

Authorization
You require appropriate authorization for the functions shown. This is an authorization based on the authorization object S_PROGRAM for executing a program for authorization group 'SWI_OA'.

Features
You can call the function for displaying linked workflows from several starting points.

Calling via SAP Easy Access
Two functions are available via SAP Easy Access:
- **Workflows for object**
  - This function displays all workflows linked to a particular object (for example a specific notification of absence). You must specify this object beforehand.
- **Workflows for object type**
  - This function displays all workflows linked to a particular object type (for example FORMABSENC). You must specify this object type beforehand.

You can only specify objects whose object type definition supports the interface IFFIND.

Calling from the Business Workplace
In this case, you do not specify the object. The workflow relationships are displayed for the object processed in the selected work item.

Calling as a generic object service
In this case, you do not specify the object. The workflow relationships are displayed for the object being processed.

What is displayed?
The display of the workflow relationships has two parts. The upper part displays the workflows identified for the object or object type. The lower part displays data on the currently selected work item in the form of a simplified workflow log:
- **Steps so far**
The workflow steps processed so far are listed under *Step name*. The steps are linked to the work item preview of the relevant work item. The current agents of the work item are listed under *Agents*. Click once to go into detailed display of the user data.

- **Information objects addressed so far**

  All the objects and attachments belonging to the workflow are displayed here. Click once to display the information objects.

### Activities

<table>
<thead>
<tr>
<th>Calling...</th>
<th>Path</th>
</tr>
</thead>
<tbody>
<tr>
<td>... Via SAP Easy Access</td>
<td>Tools → Business Workflow → Development → Runtime Tools → Workflows For Object</td>
</tr>
<tr>
<td>... From the <em>Business Workplace [Seite 1368]</em></td>
<td><img src="image" alt="Display workflow relationships" /></td>
</tr>
<tr>
<td>... As a generic object service</td>
<td>System → Workflow → Workflow overview</td>
</tr>
</tbody>
</table>
Change Work Item

Use
The function enables you to change information and data associated with the selected work item.

Features

Changes in the initial screen
In general, the changes that are allowed depend on the type and status of the work item. The following changes are possible:

- **Work item text**
- **Priority**
- **Deadline data**
  You can set and change deadline data for the start and end deadlines of the work item. You can only make specifications for deadlines that are still being monitored and are in the future. For example, dialog work items with the *ready* status can no longer be assigned a requested start.

Manual intervention into the processing of work items
If you are an administrator with the appropriate authorization, you can intervene manually into the processing and therefore into the process flow of a workflow. This is particularly helpful for dealing with errors. The following changes are possible:

- **Set to ‘ready’ manually**
  The work item status is changed from *waiting* to *ready*.
  The work item therefore appears in the workflow inboxes of the selected agents.

- **Complete manually**
  The work item status is changed to *completed*.
  In the case of dialog work items, this status change (together with the current work item container) is passed on to the workflow system for evaluation. If the object method to be executed is a synchronous method with a result, the possible result values are displayed for selection.

- **Replace manually**
  The status of the work item is reset to *ready*.
  The work item is therefore displayed again in the workflow inboxes of all selected agents.

- **Restart after error**
  Work items and, in particular, workflows with errors can be restarted after the errors have been corrected.
  The information written by the workflow system in a log when the error occurred is now taken into account.

- **Delete logically**
  The status of the work item is changed to *logically deleted*.
In the case of dialog work items, this status change is passed on to the workflow system.

**Locking/unlocking execution**
An administrator can lock or unlock the current work item for execution. To do this, you select the appropriate function from the *Edit* menu.

**Deleting deadlines**
To delete a deadline you have changed or entered and reset it to its initial values, choose *Edit → Delete deadline*.

**Changing work item containers**
Choose *Edit → Change container* to go to the editor for changing the work item container.
The contents of the container for the relevant work item are displayed. You can change the current, runtime-specific data for the particular work item.
The container may still have elements that do not currently have a value. These elements are hidden as standard. The presence of elements that are not displayed is shown by an indicator. To display these elements, choose *Edit → Show elements*.
To add more lines to container elements defined as multiline, choose *Edit → Additional line*.

**Activities**
You can access this function in the [Business Workplace][Seite 1368] by choosing [Other functions], the relevant context menu or the [technical work item display][Seite 1418].
Set Work Item to Done

Use
This function is used by the recipient of a work item to confirm explicitly that processing of this work item has been completed. As long as explicit confirmation has not been provided, the work item has executed status and can be executed again or forwarded.

> The status of the work item changes from executed → completed.

Prerequisites
This function is only possible on a work item if a setting was made in the definition of the associated single-step task stating that the end of processing must be confirmed explicitly.

Activities
You use this Business Workplace function if the status of the work item has been changed to executed by a terminating event.
In general, however, the work item is not set to done via the Business Workplace. This function is usually provided as a dialog box directly after a work item has been processed.
**Missed deadline work items [Extern]** must always be confirmed explicitly. After execution, they remain in the status in process until they are set to set to done.
You can access this function in the Business Workplace [Seite 1368] by choosing Other functions.
Workflow Outbox

Use
The workflow outbox lists the workflows started and the work items forwarded and executed by the current user.

Integration
As is the case for the workflow resubmissions [Seite 1443] and the workflow inbox [Seite 1408], the workflow outbox is an integral part of the Business Workplace.

Features

Views in the workflow outbox

Started workflows
This view shows work items for the tasks started by you as a user in dialog or by a triggering event whose event container contains your user name as _Evt_Creator.

Work items executed by me
The work items executed by you are displayed in this view.

Forwarded work items
The work items forwarded by you are displayed in this view.

Workflow Functions
The screen area in which the worklist is displayed has a toolbar containing the workflow functions and functions for configuring the list. For information about these functions, refer to the documentation on the ABAP List Viewer [Extern].

The following functions are available for the different workflow outbox views:

- **Update**
  You use this function to update the workflow outbox.

- **Display work item** (not in Started workflows view)
  This function goes directly to the work item display [Seite 1411]

- **Change agent**
  You use this function to perform an ad-hoc agent assignment [Seite 1097], [Seite 1411]

- **Display workflow log**
  This function displays the workflow log [Seite 1420].

- **Display attachments** (refer to attachment management [Seite 1429])

- **Environment**
  - Display objects in workflow (dialog work items only)
  - Display workflow relationships [Seite 1436]
  - Start Workflow [Seite 1450]
• [Create link [Seite 1434]]

• [Change selection period]
  • Today only
  • Last 7 days
  • Last 30 days
  • User-defined ...

The functions described above can be called using the relevant context menu.

**Activities**

You can access the workflow outbox in the overview tree in the Business Workplace [Seite 1368] via Workplace \(\rightarrow\) Outbox.
Workflow Resubmissions

Use
The workflow resubmissions [Extern] function can be used to resubmit work items for processing at a later point in time or periodically and to display these work items in a list. Technically, the following applies to work items which appear in your workflow resubmission folder: the work item status is set to waiting, your name is entered as the actual agent, and the requested start date is set to the resubmission date.

Integration
As is the case for the workflow inbox [Seite 1408] and the workflow outbox [Seite 1441], the workflow resubmissions is an integral part of the Business Workplace.

Features

Workflow Functions
The screen area in which the worklist is displayed has a toolbar containing the workflow functions and functions for configuring the list. For information about these functions, refer to the documentation on the ABAP List Viewer [Extern].

The following functions are available in the workflow resubmissions:

- **Update**
  You use this function to update the workflow resubmissions.

- **Display work item**
  This function goes directly to the work item display [Seite 1411].

- **Display workflow log**
  This function displays the workflow log [Seite 1420].

- **Display attachments**
  This function is only available if the work item has attachments. Refer to attachment management [Seite 1429].

- **End resubmission**
  The work item is put back into the workflow inbox. It has the status reserved.

- **Environment**
  - Display objects in workflow (dialog work items only)
  - Display workflow relationships [Seite 1436]
  - Start Workflow [Seite 1450]
  - Create link [Seite 1434]

The functions described above can be called using the relevant context menu.

Activities
You can access the workflow resubmissions in the overview tree in the Business Workplace [Seite 1368] via Workplace → Resubmissions.
You are asked for a date for resubmission of the work item. The work item then disappears automatically from the workflow inbox and is not displayed there again until the date specified.
Work Item Preview

Use
The work item preview is in the bottom right corner of the Business Workplace screen and provides a preview of the properties of a work item. All the information on a work item is displayed if you choose the function Display work item in the worklist of the workflow inbox [Seite 1408] or the workflow resubmissions [Seite 1443]. This function goes directly to the work item display [Seite 1411].
A concise view of the workflow log [Seite 1420] is offered in the workflow outbox [Seite 1441] when the started workflows are displayed.

Integration
The work item preview is part of the Business Workplace [Seite 1368].

Features
Workflow functions
The description of the work item is displayed in the work item preview. It is also possible to access objects and attachments of the work item directly from the preview.
The concise view of the workflow log enables you to access processed steps (individual work items), their agents and information objects addressed so far.
For more information about the concise view of the workflow log, please refer to Workflow Relationships [Seite 1436].

Tips & tricks
"Tips & tricks" are displayed when you position the mouse pointer on the title of a "tips & tricks" entry. Whenever the work item preview is updated, a new title is offered.
You can activate/deactivate "tips & tricks" in your personal workflow settings [Seite 1395] (Settings → Workflow settings → Personal settings).

User exit
The work item preview can be adapted to customer-specific requirements using a user exit [Extern].

Activities
The work item preview is activated/updated whenever you select a work item in your worklist.
Workflow System Administrator

Definition
Person within the system of workflow roles [Seite 94] who is responsible for the technical maintenance of the development environment and the runtime behavior after auto-Customizing.

Use
This part of the documentation is to be used in conjunction with the user menu for the workflow system administrator [Extern]. This user can access this user menu on the screen SAP Easy Access via .

Assign the following roles to the user:
- SAP_BC_ENDUSER
- SAP_BC_SRV_USER
- SAP_BC_BMT_WFM_ADMIN

Structure
The documentation for this role consists of:
- Administration of the Workflow System [Seite 684]

Integration
This role is part of the role-based procedure model containing the following other roles:
Administration of the Workflow System

Purpose
Administration of the workflow system consists of the constant monitoring and maintenance of the development environment and runtime environment of the workflow system.

Process Flow

The administration tools include:

- Reports enabling operations to be performed on work items (runtime)
- Event manager settings (linkage tables, event trace, event queue)
- Maintenance of the "global" workflow system administrator [Extern]
- SAPforms [Seite 1718] settings

For further information, refer to Administration Tools [Seite 1500].
Administration Tools

Use

The administration of the runtime system consists of several reports that can be split up into the following categories:

- Workflow runtime
- Event manager
- Workflow definition time
- WebFlow
- SAPforms [Seite 1718]

Prerequisites

Customizing

The following assumes that you have already performed automatic Customizing [Seite 1504]. Remember task-specific Customizing [Seite 1710], if you want to use SAP workflows or SAP tasks. Refer to the documentation Workflow Scenarios in Applications [Seite 1703].

Features

If you, as workflow system administrator, need an alternative way of accessing SAP Business Workflow functions, you can use the relevant transaction code. For an overview of the most important transaction codes, refer to Important Transaction Codes [Extern].

Workflow runtime

Work item deadline monitoring

Tasks with deadlines have also have deadline monitoring based on a background job. You can change the configuration of the background job, schedule it, delete it or display it. For further information about the background job, select (Display background job).

Work item error monitoring

If work items have the status ERROR for a long time, they are restarted automatically. This task is performed by a report that is controlled by a background job. You can change the configuration of the background job, schedule it, delete it or display it. You can also start the report manually via Execute work item error monitoring.

For further information about the background job, select (Display background job).

Reorganization

These functions include reports for archiving and deleting work items. Refer to:

- Archive work item [Extern]

  During archiving, data that is no longer required in the system is checked using application-specific criteria and put in an archive file. The data is not removed from the database. Once the files which are to be archived have been copied to the archive file completely, they can be removed from the database in a separate program run.

  All interactions in conjunction with archiving are performed via archive management (Tools → Administration → Administration → Archiving).
All actions or programs are processed in the background. The background jobs required can be scheduled in archive management.

For more details about archiving, refer to the documentation Archiving Application Data [Extern].

- **Display workflows from archive**
  
  You can use this report to display a workflow for an application object. After the workflow work item determined via the selection criteria is read from the archive, the system displays the workflow log. The functions of the workflow log are not fully available however.

  For further information, select [ ].

- **Delete work item [Seite 1510]**

  The report Delete work item history (RSWWHIDE) deletes the work item history (all workflow log entries relevant to work items) without archiving. If you do not select the indicator Delete immediately, the report is only executed on a test basis.

**Work items without agents**

This report displays all work items without agents for a particular selection period. You can go to work item display [Seite 1411] from the list displayed by double-clicking.

**Work items with monitored deadlines**

This report displays all work items with monitored deadlines for a particular selection period. You can go to the work item display from the list displayed by double-clicking.

**Diagnosis of workflows with errors**

You can use this function to investigate and, if necessary, restart workflows with errors [Seite 1507].

**Unlock workflows**

This report can be used to select and unlock work items that are locked against execution.

  Locking tasks against execution is a function in task maintenance [Seite 1192].

**Execute work item without agent check**

This function enables a user to execute work items without checking agents [Seite 1508].

**Workflow restart after error**

This report can be used to restart workflows with errors [Seite 1509].

**Synchronize runtime buffer**

You have the opportunity to initialize the main memory buffer used by the workflow development environment. This rereads the current values from the database so that the current data is used.

  You should always refresh the runtime buffer when you have changed something in the task definition.

After execution of this report, certain workflow functions initially have lower performance. This applies until the main memory contains all the necessary data.
Continue workflow after system crash
You can use this report to select and continue workflows that have had the status started for longer than a day. This means that workflows that have come to a halt after system errors can be continued.

Configure workflow RFC destination
The workflow runtime system executes its RFC and tRFC (transactional RFC) calls on a logical destination. You can set up or change a logical destination. No logical destination is configured for the workflow system tRFC calls as standard.

Upper limits workflow runtime
You can make the following settings here:
- Maximum number of nested subworkflows permitted at runtime
- Maximum number of nodes that can be processed at runtime within a workflow definition
You can also activate/deactivate the display of agent names in the workflow log.

Event manager
Type linkages and instance linkages
Tables containing the assignment of events to the event receivers interested in the event. For further information, refer to Maintenance of Linkage Tables [Seite 1353].

Event trace
The event trace can be used to log events.

Event queue
The event queue can be used to store events temporarily.

Workflow definition time
Workflow Builder
Here you can maintain:
- The workflow system administrator with "global" responsibility, that is, responsible for all workflow definitions
- The task executed in a user decision. If you create a step of the type user decision in a workflow definition, the task entered here is transferred into the workflow definition.

Please refer also to the Customizing activity Maintaining Administration Data for the Workflow Builder.

WebFlow
Customizing Web server
You use this function to specify the Web server that is configured for your SAP System. For further information, refer to Defining the Web Server [Seite 1513].

SAPforms
Mail-enabling
This report can be used to schedule the sending of R3F messages. This is necessary if you want to execute a work item as a form in an external mail system.
The report is started immediately and then scheduled as a background job in accordance with the period entered.

**Diagnosis**
You have the opportunity to call reports with which you can search for errors that have occurred.

**Trace on/off**
The SAPforms trace logs all SAPforms-related actions.
You have the opportunity to activate/deactivate this trace.
Please also refer to [SAPforms Administration][1].

**Activities**
You can access all the workflow system administration reports described by choosing **Tools → Business Workflow → Development → Administration.**
Diagnosis of Workflows with Errors

Use
This function establishes all workflows with errors and groups them according to error cause. The evaluation results displayed are runtime information. If the tables are already very full, the report will have a longer runtime (a few minutes).

To be precise, the system establishes top-level work items with errors. Top-level work items are either workflows or work items of all other types that are not part of a workflow.

Features
The ABAP List Viewer [Extern] is used to display the work items with errors.
The possible error causes used by the system for grouping purposes are:
- Agents
- Deadlines
- Binding from workflow to work item
- Other
The following additional functions are available on the application toolbar:
- **Workflow log**
  If you select a work item and choose this function, the relevant workflow log [Seite 1420] is displayed.
- **Error analysis**
  If you select a work item and choose this function, a dialog box appears listing errors, warnings and notes about the selected work item.
- **Change work item**
  If you select a work item and choose this function, you go the screen Change work item [Seite 1438].
- **Restart after error**
  Choose this function if you have processed a work item with errors and you want to restart it.

Activities
You can access the processing of workflows with errors by choosing Tools → Business Workflow → Development → Administration → Workflow runtime → Diagnosis of workflows with errors.
Executing Work Items Without Checking Agents

Use
This function enables a user to execute work items without checking agents. The system does not check whether the user is one of the possible agents [Extern] of the work item to be executed.

Features
You use a selection screen to generate a list of work items. Selection is carried out as in work item selection [Seite 1490]. The following functions are available in the list:

- 📊 **Execute without check**: The work item is executed without checking agents.
- 📊 **Display work item**: This function branches to the work item display [Seite 1411].
  
  As an alternative, you can double-click on the work item.

- 📊 **Display workflow log**: This function branches to the workflow log [Seite 1420].

- 📊 **Display work item container**: This function branches to the work item container [Seite 1511].

- 📊 **Display task**: This function displays the underlying task of the work item.

- 📩 **Send...**: The work item is sent as an attachment to a mail.

For information on other functions, refer to the ABAP List Viewer [Extern] documentation.

Activities
You can access this function by choosing Tools → Business Workflow → Development → Administration → Workflow runtime → Execute work items without agent check.
**Workflow Restart After Error**

**Use**
This report can be used to display a list of workflows with errors for a particular selection period, and then restart them. The indicator *Restart immediately* can be used to restart a workflow immediately.

**Features**
You use a selection screen to generate a list of work items. The following functions are available in the list:

- **Restart workflow**: The selected workflow is restarted.
- **Display work item**: This function takes you to the work item display [Seite 1411].
  
  As an alternative, you can double-click on the work item.

- **Display workflow log**: This function takes you to the workflow log [Seite 1420].

- **Display work item container**: This function takes you to the work item container [Seite 1511].

- **Display task**: This function displays the underlying task of the work item.

For information on other functions, refer to the ABAP List Viewer [Extern] documentation.

**Activities**
You can access this function by choosing **Tools → Business Workflow → Development → Administration → Workflow runtime → Workflow restart after error.**
Deleting Work Items

Use
This report (RSW0WIDE) deletes work items from tables without archiving.

⚠️
Therefore, this report should not be used in a production system.
In a production system you must use archive management [Extern] to archive and delete work items in order to ensure data consistency.

Features
On the selection screen, you can set the following indicators in Technical settings.

- **Delete immediately**
  
  If you set this indicator, the work items selected are deleted immediately.
  
  Otherwise, a list is displayed in which you can delete work items using the function. You can also go to the workflow log by double-clicking.

- **Delete log data as well**
  
  If you set this indicator, the log data for the selected work items is deleted from the workflow log.

- **Number of work items per LUW**
  
  Here you can enter the number of work items to be deleted per database transaction. The setting is database-specific. Refer to the SAPNet - R/3 Frontend note with the number 107410.

Activities
You can access this function by choosing Tools → Business Workflow → Development → Administration → Workflow runtime → Reorganization → Delete work item.
Administration of Event Queue

Use
The event queue can be used to store events temporarily in a memory. If the event queue is switched on, certain events are not passed immediately to the event receivers, but are stored temporarily in the event queue instead. This means that the system load caused by a large number of events being created can be spread over a longer time period (which can be set by the workflow system administrator). This combats the threat of system overload.

This is achieved by the transactional RFCs for starting the receivers not being called immediately, but after a time delay and in small numbers.

Using the event queue therefore delays the calling of the receivers. The event queue can also be used to store events that have errors temporarily.

Integration
The event queue is one of the SAP Business Workflow administration tools [Seite 1498].

Prerequisites
The event queue can only be used for events if the relevant event linkages are appropriately classified. The indicator Enable event queue in type linkage table maintenance [Seite 1358] can be used to do this.

Features

Functions on the application toolbar

- **Refresh**
- **Set default values**
- **Undo change**
- **Restore**
- **Transport administration data**
  This function can be used to assign the administration data to a transport request.
- **Event trace**
  All events created correctly are logged in the event trace [Seite 1396] irrespective of whether potential receivers exist.
- **Browser**
  The event queue browser [Seite 1524] can be used to view the contents of the event queue.
- **Delete events**
  This report can be used to delete events from the event queue.
**Administration of Event Queue**

**Tab page Overview**
When you start event queue administration, the tab page *Overview* is activated. This tab page shows all the statuses of the settings that can be made on the other tab pages. It also shows the current content of the event queue and other statistics.

**Other tab pages**
The following tab pages are available for the actual administration:

<table>
<thead>
<tr>
<th>Tab page</th>
<th>Function</th>
</tr>
</thead>
<tbody>
<tr>
<td>Basic data [Seite 1517]</td>
<td>The administrator and the error feedback behavior are maintained here.</td>
</tr>
<tr>
<td>Activation [Seite 1519]</td>
<td>The event queue is activated here.</td>
</tr>
<tr>
<td>Background job [Seite 1521]</td>
<td>The event queue is read at regular intervals by a background job.</td>
</tr>
<tr>
<td>Event delivery [Seite 1523]</td>
<td>The event delivery specifies how the event receivers are to be started.</td>
</tr>
<tr>
<td>Linkages with errors [Seite 1525]</td>
<td>Those event linkages for which there are events stored temporarily because of errors, or which have an error status are displayed here.</td>
</tr>
</tbody>
</table>

**Activities**
You can access administration of the event queue by choosing *Tools → Business Workflow → Development → Administration → Event manager → Event queue.*
Maintenance of Tab Page Basic Data

Use
You maintain the basic settings for the event queue on this tab page. The administrator and the behavior if there are events with errors are defined here.

Integration
This tab page is maintained within administration of the event queue [Seite 1515].

Features

Administrator
An object of Organizational Management (organizational unit, job, position, work center, user) can be defined as an administrator. This organizational object is notified if no other organizational objects are defined in the relevant context (for example a notification if there are event receivers with errors).

Receiver error feedback
Various errors can occur within an event receiver (for example a workflow started by an event). If an error of this type occurs, feedback can be sent to the event manager.

The runtime behavior in the event of errors can be determined not only in administration of the event queue, but also in maintenance of type linkages [Seite 1358].

The entries on the tab page Basic data work as system presettings. They can be adopted or redefined for each type linkage.

Deactivation of linkage
The linkage for the relevant event receiver is deactivated. If the linkage is deactivated, this receiver (for example a workflow) is not started for all events that follow. The events are not stored temporarily. This setting is in accordance with the classic runtime behavior.

Mark linkage as having errors
The event linkage is not deactivated, but marked as Errors. All events for this linkage are stored temporarily and can be delivered manually. The event linkage must be defined manually as No errors after the error cause has been removed for the receiver to be started again automatically. This can be done either on the tab page Linkages with errors [Seite 1525] or in maintenance of event linkages [Seite 1358]. You can also deliver the events from the queue again on this tab page.

Do not change linkage
If there is error feedback, the linkage is not changed. The event is stored temporarily for redelivery. If more events follow for this linkage, an attempt is made to start the receiver each time. If an error is discovered, the relevant events are stored temporarily. In contrast to marking the linkage as Errors, an attempt is made to start the receiver for each event. If there is always an error, the system load is greater than with direct temporary storage of events. Errors may, however, depend on event parameters. Therefore, only those events for which the error has occurred are stored temporarily with this configuration. Receivers that have no errors based on the event parameters are started immediately.
Activities
You can access administration of the event queue by choosing Tools → Business Workflow → Development → Administration → Event manager → Event queue.
Maintenance of Tab Page Activation

Use
On this tab page you activate the event queue and define the linkages for which the event queue is to temporarily store events and receivers.

Integration
This tab page is maintained within administration of the event queue [Seite 1515].

Features

Settings

Switch on event queue
If the event queue is switched on, all the events created are written into the event queue before delivery, provided delivery via the event queue was specified for them. Delivery via the event queue must be defined separately for each event linkage (see below).
If the event queue is switched off, no events are written into the event queue.

Reading from the event queue does not depend on this setting. Therefore, events are delivered from the event queue even if it is switched off. The background job must be active for this.

Delete event after delivery
The event are stored persistently in the event queue.
You can set whether events are to be deleted after delivery. If the delivered events are not deleted, the indicator delivered is set for them. If delivered events are not deleted, you can still view them even after the delivery.

You should delete delivered events in a production system to accelerate the processing of undelivered events.

Event linkages
The event queue can only be used for events if the linkages are appropriately classified.
You must set the indicator Enable event queue in maintenance of type linkages [Seite 1358].

Active configuration
The information in this status frame answers the following questions:

- Is the event queue switched on?
- Is the background job active?
- Are events deleted automatically after delivery?
- How many event linkages are using the event queue?
Maintenance of Tab Page Activation

Activities

You can access administration of the event queue by choosing Tools → Business Workflow → Development → Administration → Event manager → Event queue.
Maintenance of Tab Page Background Job

Use
The event queue is read at regular intervals by a background job. The events found are delivered to the registered event receivers.

Integration
This tab page is maintained within administration of the event queue [Seite 1515].

Features
You can make the following settings:

- **Operation mode of background job**
  The operation mode can be used to control how often the background job is to be started.
  - **Periodic**
    The background job is scheduled periodically with the specified interval.
  - **Depending on load (dynamic)**
    The intervals at which the background job is to be started can be defined as dependent on the content of the event queue. You should, for example, schedule the background job for a shorter interval if there are still events in the event queue after a processing run. You should choose a longer interval if the event queue does not contain any more events.
    Adapting the interval dynamically can relieve the load on the system, since the background job is started less frequently.
    However, the processing of the events may also be delayed because of the longer interval.
    To switch operation mode, you must first remove the current background job from the schedule.

- **Number of events per read access**
  Here you specify the number of events to be read each time the event queue is read.

- **Time interval between two read accesses**
  Here you specify the time interval between two read accesses. The specification has the following effect:
    - **With periodic processing**
      Processing is carried out with the predefined interval. The interval can only be changed if the background job is removed from the schedule. It is considered when the job is rescheduled.
    - **With processing that depends on load**
Maintenance of Tab Page Background Job

The time interval is adhered to if the event queue contains more events after a processing run. The interval can even be changed when the background job is scheduled, since the job reschedules itself after every processing run (depending on the content of the queue).

- **Interval until event queue next checked**
  
  This interval is adhered to if there are no more events after the queue has been read.
  
  This setting can only be made with processing that depends on load.

- **Schedule background job**
- **Display background job**
- **Unschedule background job**

**Activities**

You can access administration of the event queue by choosing *Tools → Business Workflow → Development → Administration → Event manager → Event queue.*
Maintenance of Tab Page Event Delivery

Use
On this tab page you maintain how the event receivers are to be started.
A defined number of events is delivered by the background job in each cycle. These events can be processed sequentially or in parallel (with aRFC). The receivers can be started synchronously (with RFC) or asynchronously (with tRFC).

Only synchronous starting of receivers is currently supported for parallel processing.

Integration
This tab page is maintained within administration of the event queue [Seite 1515].

Features
Processing of events
Sequential
Events are delivered from the event queue in defined units (for example five events per read access). Sequential processing means that these events are processed in succession.

Parallel on server group
Events are delivered from the event queue in defined units (for example five events per read access). Parallel processing means that the events are processed at the same time in separate processes. These parallel processes can run using a defined server group.

A server group combines a number of application servers to form a logical group. If you specify a server group, the event delivery is executed using the application servers in this group.

If you do not specify a server group, the parallel execution is distributed across all application servers. You are recommended to use a server group with the appropriate resources.

Start of receivers
Synchronous
Various receivers (for example workflows) can be started for an event. Synchronous start of receivers means that they are started using a remote function call (RFC). The processing waits until the receiver call is complete.

Asynchronous
Various receivers (for example workflows) can be started for an event. Asynchronous start of receivers means that they are started using a transactional remote function call (tRFC). The processing does not wait until the receiver call is complete, but is continued immediately instead. The receivers are started asynchronously.
Sequential processing of events with synchronous start can be used for test purposes.

The delivery mode recommended in the standard version is sequential processing of events with asynchronous start of receivers.

Parallel processing (with synchronous start) can be used to distribute the system load on a predefined server group. Delivery then only uses the application servers assigned to this server group. If you do not specify a server group, the parallel execution is distributed across all application servers. You are recommended to use a server group with the appropriate resources.

**Activities**

You can access administration of the event queue by choosing Tools → Business Workflow → Development → Administration → Event manager → Event queue.
Maintenance of Tab Page Linkages With Errors

Use
If an error occurs within an event receiver, this error can cause the event data to be lost and prevent the receiver from starting. To enable the event to be delivered again after an error, the event data can be entered into the event queue.

Integration
This tab page is maintained within administration of the event queue [Seite 1515].

Features
Those event linkages for which there are events stored temporarily because of errors, or which have an error status are displayed in the overview.
Temporarily stored events can be initiated again by selecting a linkage and then starting the delivery (function Start delivery). You can specify whether you want to deliver one event or all events (indicator Only deliver one event).
The option of delivering a single event is intended to save resources in the event of another error. Irrespective of that, events can be redelivered immediately or by the event queue background job. The background job [Seite 1521] must be active for this.
The linkage between the event and the receiver can be set to status No errors by selecting the symbol for the linkage that has an error in the column Status of linkage.
Events can be deleted from the list Linkages with errors using Events. This always deletes all events.
The indicator Only deliver one event is not evaluated.
The function Delete events is available in the application toolbar to delete individual events.

Activities
You can access administration of the event queue by choosing Tools → Business Workflow → Development → Administration → Event manager → Event queue.
Event Queue Browser

Use
The event queue browser displays not only the current content but also a history of the content of the event queue. This is dependent on the selection criteria chosen.
Event container, event trace and workflow log can be displayed from the selected list. Events not yet delivered can also be delivered directly.

Integration
Which events are written into the event queue is defined in the administration environment [Seite 1515] of the event queue.

Features

Selection area
In the selection area you can select events for display in the browser.
If you restrict the search range, the symbol appears next to the pushbutton Standard criteria.
You can “fine-tune” the selection via the pushbutton Further criteria. The symbol appears here too if you enter selection criteria.

Selected list
The list is refreshed automatically every time the selection conditions are changed. An event can be displayed from the list by double-clicking. The general event data and the event container are displayed.

Functions on the application toolbar

- Refresh
Event Queue Browser

- **Event**
  The selected event is displayed.

- **Deliver**
  The selected event is delivered, that is, it is provided to the relevant receiver.

- **Event trace**
  All events created correctly are logged in the [event trace][Seite 1366] irrespective of whether potential receivers exist.

- **Work Item**
  This function takes you to the [work item display][Seite 1411] of the work item operating as receiver of the selected event.

  You can only go to the work item display if the event has been written into the event trace.

**Activities**

You can access the event queue browser by choosing Tools → Business Workflow → Development → Utilities → Events → Event queue browser.
Defining the Web Server

Use
You must define the Web server to enable the WebFlow to receive XML reply documents.

Prerequisites
The SAP Internet Transaction Server (ITS) must be installed as well to enable your SAP System to react to XML documents and XML reply documents appropriately.

Procedure
2. On the screen Customizing Web Server: Change, enter the address of your Web server and, if applicable, the port number.

If you want to test the WebFlow in SAP System A and the starting of a Business Workflow in SAP System B, you require the system architecture below:

To check the prerequisites for the test, carry out the following steps:
1. Check whether the release of the ITS installed on SAP Systems A and B is at least 4.6C.
2. Check whether the service [Extern] WF_HANDLER is on the ITS.
3. The user used for the service logon must exist in the SAP System. This user is defined in the global service directory for the ITS or in the service WF_HANDLER.
4. The user defined must be authorized to store documents in the Business Document Service (BDS).
5. The user defined in SAP System B must be one of the possible agents [Extern] of the workflow to be started.

**Result**

The URL required in a Web activity is created with the Web server defined here, as soon as the indicator *Wait for feedback* is set.
Displaying XML Documents

Use
You use this function to search for and display XML documents. When a workflow that was to be started by an inbound XML document is not started, this function is the only way to display the XML document.

Integration
XML documents are saved as objects of a work item and can be displayed using the work item display.

Activities
Select an appropriate search period to look for the XML document in question. You can display the XML document from the hit list. The display takes place in the Business Document Navigator [Extern].
Authorization Management

Use
The authorizations are maintained in authorization management. You can create roles there, which contain the individual authorizations. There are roles, authorization profiles and authorization objects specifically for SAP Business Workflow. These can be used for creating your own roles. For further information, refer to BC - Users and Roles.

Activities
You should use roles for allocating authorizations. The following roles are available:

Roles for Business Workflow

<table>
<thead>
<tr>
<th>Role</th>
<th>Technical name</th>
<th>Explanation</th>
</tr>
</thead>
<tbody>
<tr>
<td>Non-critical, basic authorizations</td>
<td>SAP_BC_ENDUSER</td>
<td>Basic authorizations for all workflow participants</td>
</tr>
<tr>
<td>for all users</td>
<td></td>
<td></td>
</tr>
<tr>
<td>User for communication, workflow,</td>
<td>SAP_BC_SRV_USER [Extern]</td>
<td>Authorizations for a workflow agent [Seite 805]</td>
</tr>
<tr>
<td>appointments, etc.</td>
<td></td>
<td>and basic authorizations for all workflow</td>
</tr>
<tr>
<td>users</td>
<td></td>
<td>participants</td>
</tr>
<tr>
<td>team</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

April 2001 709
Administration of Organizational Management

Use
There are settings for controlling the interaction between Organizational Management functions and SAP Business Workflow functions.

Features
You need only learn more about these settings if you are planning further administration or consulting activities.
They are:

Table settings
Workflow-relevant data for Organizational Management
For further information, refer to Workflow-Relevant Data for Organizational Management [Seite 1533].

Permitted relationships
Table T777E contains all the relationships permitted between organizational objects.
In connection with the substitution rules, relationship 210 (substituted with profile by or substitutes with profile) is used to determine which object types (user, person, position) may be specified as a substitute [Seite 1398] for a position.
This table is supplied with content and can be processed in Personnel Management Customizing (Personnel Management→Organizational Management→Basic settings→Maintain relationships) or using transaction OOVK.

SAP organizational object type assignment
Table T7791 contains information about which SAP organizational object types should be linked with positions, which with jobs and which with organizational units.
This table is supplied with content and can be processed in SAP Business Workflow Customizing (Basis Components→Business Management→SAP Business Workflow→Basic Settings (Organizational Management)→Maintain assignments for SAP organizational object types) or using transaction 0091.
For further information, refer to Agent as Attribute of SAP Organizational Object [Seite 1302].

Prefix numbers for standard object types
Table T78NR stores the 3-digit prefix numbers used for the first 3 digits of numbers of standard tasks, multistep tasks and roles, on a system-specific and client-specific basis. The remaining 5 digits of the 8-digit numbers are allocated by internal number assignment.
These values are set in SAP Business Workflow Customizing or in automatic Customizing.

⚠️
Number assignment for customer tasks is not carried out as described above, but with a separate number range object.

The table is supplied with content and can be processed in SAP Business Workflow Customizing (Basis Components→Business Management→SAP Business Workflow→Basic Settings (Organizational Management)→Maintain prefix numbers) or using transaction OOW4.

Permitted object types for role resolution
As a check table, table T7790 contains all the organizational object types [Extern] that can be established as the result of role resolution.
The table is maintained via transaction SM30.

**Reports for PA administration and evaluation**

**Report RHRELAT0 (permitted relationships for object types)**
All the permitted relationships for one organizational object are displayed. The field *Object type* must contain a permitted abbreviation for an object type. The field *Relationship type* can remain blank, in which case all relationships in both possible directions are displayed.

**Report RHSHOWOR (display organizational assignment)**
The report displays the organizational environment of a user. The following information is displayed:
- Organizational unit to which the user is assigned.
- Position the user occupies.
- Job that describes the position.
- Tasks that are assigned directly to the organizational unit, the position, the job or the user.

**Report RHSTRU00 (display/maintain structure)**
The report goes into the structure of the organizational plan from the object entered and displays all the objects linked via the evaluation path with the object concerned. The position of the object displayed in the hierarchy is illustrated using indents.
An organizational object is specified with *plan version*, *object type* and *object ID* (user name for object type US) on the selection screen for this report.
In addition, an evaluation path is entered in the relevant field. You can establish evaluation paths for the objects used using the F4 help.
It is highly recommended that you set the *Recursion check* indicator to avoid endless loops.

**Number range objects**

**Number range object HRSOBJECT (standard objects)**

Number range [Extern] for standard objects.
There is an eight-digit number for:
- Standard tasks (TS)
- Standard roles (AC)
- Workflow templates (WS)

  It is made up of two parts:
  i. Three-digit prefix number set per client and system in table T77NR.
  ii. Five-digit sequence number assigned internally according to the settings in number range object HRSOBJECT.

**Number range object HRWFOBJECT (PD workflow objects)**

Number range for PD workflow objects.
The eight-digit number for workflow tasks (WF) is made up of two parts:
1. Three-digit prefix number set per client and system in table T77NR.
2. Five-digit sequence number assigned internally according to the settings in number range object HRWFOBJECT.
Number ranges are maintained from the ABAP Workbench via Development → Other tools → Number ranges.

**Number range object RP_PLAN**

Number assignment for
- customer tasks (T)
- organizational units, positions, jobs and work centers

is carried out according to the settings in the number range object RP_PLAN. As standard, the interval from 50000000 to 99999999 is set for subgroup $$$$ (valid for all plan versions and all object types) with internal number assignment.

You can change this setting in Personnel Management Customizing (Personnel Management → Organizational Management → Basic Settings → Number Range Maintenance) or by using transaction OONR.

**Evaluation paths**

For further information, refer to Evaluation Paths [Seite 1534].
Workflow-Relevant Data for Organizational Management

Use

You can maintain workflow-relevant data for Organizational Management via table T77S0.

Features

Change active plan version

Table T77S0 contains the value of the active plan version. This value is set in Customizing. In SAP Business Workflow automatic Customizing, the value 01 is entered for the active plan version if the field had no value.

This setting can be processed via transaction OOAP.

Work item display for substitutes

Table T77S0 contains an indicator specifying whether the substitute should automatically see the additional work items when a substitution is activated. This indicator should be set.

This setting can be processed via transaction SM30. The indicator must be set for AUTOF.

Default value for task classification

Table T77S0 contains an entry for the default value for task classification. The default value is NO_CLASS.

This setting can be processed via transaction SM30. The entry must be made for DEFCL.

Buffering in the Organizational Management environment

Table T77S0 contains information about the buffering of tasks. You can refresh the buffer manually with the report RHWFINDEXRESET. The buffer is refreshed automatically once a day.

⚠️

The buffer must be refreshed manually for changes to a workflow definition to take effect immediately. Refreshing does, however, have a negative influence on performance.

Activities

You can access automatic Customizing by choosing Tools → Business Workflow → Development → Utilities → Customizing

Refer also to:

Automatic Customizing [Extern]
Evaluation Paths

Definition
An evaluation path describes a chain of relationships that exists between individual organizational objects in the organizational plan.

Use
Evaluation paths are used in connection with the definition of roles [Extern] and views.

The evaluation path O-S-P describes the relationship chain Organizational unit > Position > Employee.

Evaluation paths are used to select other objects from one particular organizational object. The system evaluates the organizational plan along the evaluation path.

Starting from an organizational unit, evaluation path O-S-P is used to establish all persons who belong to this organizational unit or subordinate organizational units via their positions.

Integration
Evaluation paths are defined in the standard system. The evaluation paths available include both elementary evaluation paths with just one relationship (for example A003 belongs to, B007 is described by), and complex evaluation paths (for example O-S-P internal persons per organizational unit, as described above).

You can use the evaluation paths available or define your own.
Evaluation paths can be tested and analyzed using the report RHSTRU00 [Seite 1530].

See also:
Views [Seite 1402]
Maintaining Evaluation Paths

1. To maintain evaluation paths, go to the relevant activity "Maintain evaluation paths" in Organizational Management Customizing. During operation use transaction OOAW.

   Here you will find the maintenance view Change evaluation paths: Overview.

2. Assign a name to the evaluation path (maximum of 8 alphanumeric characters).

3. Go to the dependent view Evaluation paths (individual maintenance).

4. Describe the relationship chain to be used for the evaluation.

   To establish the permitted relationships for an object type, use report RHRELAT0.

   When using an evaluation path in a view, you should consider the following:

   - Define the evaluation path in such a manner that the relationship chain always starts from a user (object type US in Organizational Management) and ends at an organizational unit, a position or a user.

   - When defining the evaluation path, use the Skip indicator (see below) in order not to overload the result of the evaluation.

   When using an evaluation path in a role, you should consider the following:

   - Define the evaluation path in such a manner that the relationship chain starts from the object type of Organizational Management, which you can pass in the role parameter container.

   - You can select the Skip field to determine that this part of the evaluation path should be taken into account but not output.

5. You must always enter * in the field Priority.

   Before creating new evaluation paths, check the evaluation paths available as standard.

   Starting from an organizational unit, evaluation path O-S-P establishes all persons who belong to this organizational unit or subordinate organizational units via their positions. Evaluation path O-S-P is defined as follows:

<table>
<thead>
<tr>
<th>No.</th>
<th>Object type</th>
<th>A/B</th>
<th>Relationship</th>
<th>Relationship name</th>
<th>Priority</th>
<th>Type</th>
<th>Skip</th>
</tr>
</thead>
<tbody>
<tr>
<td>10</td>
<td>O</td>
<td>B</td>
<td>003</td>
<td>incorporates</td>
<td>*</td>
<td>S</td>
<td></td>
</tr>
<tr>
<td>20</td>
<td>S</td>
<td>A</td>
<td>008</td>
<td>holder</td>
<td>*</td>
<td>P</td>
<td></td>
</tr>
<tr>
<td>30</td>
<td>O</td>
<td>B</td>
<td>002</td>
<td>is line supervisor of</td>
<td>*</td>
<td>O</td>
<td></td>
</tr>
</tbody>
</table>

   The relationship defined in the last line ensures that user assignments in subordinate organizational units are included in the evaluation path as well. Since the Skip indicator is not selected for any of the lines, evaluations will always display all the objects established.

   Starting from a person or a user, evaluation path WF_ORGUN first establishes the relevant position and then the organizational unit to which this position belongs. Evaluation path WF_ORGUN is defined as follows:

<table>
<thead>
<tr>
<th>No.</th>
<th>Object type</th>
<th>A/B</th>
<th>Relationship</th>
<th>Relationship name</th>
<th>Priority</th>
<th>Type</th>
<th>Skip</th>
</tr>
</thead>
<tbody>
<tr>
<td>10</td>
<td>US</td>
<td>A</td>
<td>208</td>
<td>is identical to</td>
<td>*</td>
<td>P</td>
<td>x</td>
</tr>
</tbody>
</table>
Maintaining Evaluation Paths

| 20 | * | B | 008 | holder | * | S | x |
| 30 | S | A | 003 | belongs to | * | O |

This evaluation path works both when the positions are occupied by users and when the positions are occupied by persons who are then assigned to users.

This evaluation path can be used with a user or a person as the starting point.

Since the *Skip* indicator is selected for the first two lines, evaluations only display the organizational units found most recently.
Work Item Selection

Use
This function can be used to select and display work items of all types according to various criteria. The option of finding work items using the selection report is intended primarily as a tool for administrators. In particular, you can use this report for "lost" work items that do not appear in any workflow inbox [Seite 1408] or not in the expected inbox.

If you want to program your own work item selections or evaluations, you must only use the view SWWVPUBLIC for selection. This view contains all the released header data of a work item.

Features

Selection conditions

- **ID**
  
  Sequence number of a work item, which is assigned automatically by the work item manager. If you know the ID of the work item, for example from an error report, enter it here. If you enter an ID as a selection criterion, the system will ignore all other selection criteria.

- **Work item type**
  
  For this and the two selection criteria below, refer to work item [Seite 1371].

- **Status of the work item**

- **Priority**

- **Task**
  
  ID of the task [Seite 1172] represented by the work item.

  The task ID must be appropriate to the work item type selected.

  Work item type W either represents a standard task (ID TSxxxxxxxx) or a customer-defined customer task (ID Txxxxxxxxx).

  Work item type F either represents a workflow task (ID WFxxxxxxxx) or a workflow template (ID WSxxxxxxxx).

- **Task group**

- **Creation date, creation time**

  When specifying periods, ensure that the second entry is later than the first. For example, do not use periods such as "from 13:05:00 to 00:00:00".
You select the *additional data* indicator if the hit list is also to contain the columns *number* of the workflow definition, *version* of the workflow definition, *workflow administrator* and *executed by*.

**Functions on hit list entries**
The selection report displays the work items found in a hit list. The [ABAP List Viewer](http://example.com) is used to display the hit list.

**Application toolbar functions**

- **Refresh**: This function refreshes the hit list.
- **Display work item**: This function takes you to the [work item display](Seite 1411).
- **Display workflow log**: This function takes you to the [workflow log](Seite 1420).
- **Display work item container**: This function takes you to the [work item container](Seite 1511).
- **Display task**: This function displays the underlying task of the work item.

**Functions accessible by double-clicking**

By double-clicking on an entry in the hit list, you can go:

- To the [work item display](Seite 1411) if the entry is not a workflow work item (type F).
- To the [workflow log](Seite 1420) of a workflow if the entry is a workflow work item (type F).

**Other functions under Edit → Work item →**

- **Restart after error**
  
  This function restarts errored *workflows* after the error has been corrected.

  The information written by the workflow system in an error log when the error occurred is now taken into account.

- **Complete manually**
  
  This function changes the status of the work item to *completed*.

  In the case of type W work items, this status change together with the current work item container is forwarded to the workflow system for evaluation. If the object method to be executed is a synchronous method with a result, the possible result values are displayed for selection.

- **Execute without check**
  
  You can use this function to execute the work item without any further checks.

- **Execute batch item**
  
  This function is for executing [background work items](Seite 1385).

- **Complete with event**
  
  This function enables you to [create an event](Seite 1365) that completes the selected work item.

- **Technical data**
  
  This function displays additional technical data on the selected work item.

- **Activate/deactivate workflow trace**
These functions activate/deactivate the workflow trace [Seite 1496] for an individual work item.

Use this type of call if a workflow has already been started in the background.

**Activities**

To execute work item selection, choose Tools → Business Workflow → Development → Utilities → Work Item Selection.

You start the selection by choosing Program → Execute.
Workflow Relationships

Use
You can use this function to establish the work items in which a particular object is being processed. The following work item types are taken into account:
- Dialog work items (type W)
- Background work items (type B)
- Workflow work items (type F)
The list displayed is purely runtime information.

Prerequisites

Authorization
You require appropriate authorization for the functions shown. This is an authorization based on the authorization object S_PROGRAM for executing a program for authorization group 'SWI_OA'.

Features
You can call the function for displaying linked workflows from several starting points.

Calling via SAP Easy Access
Two functions are available via SAP Easy Access:
- Workflows for object
  This function displays all workflows linked to a particular object (for example a specific notification of absence). You must specify this object beforehand.
- Workflows for object type
  This function displays all workflows linked to a particular object type (for example FORMABSENC). You must specify this object type beforehand.
  You can only specify objects whose object type definition supports the interface IFFIND.

Calling from the Business Workplace
In this case, you do not specify the object. The workflow relationships are displayed for the object processed in the selected work item.

Calling as a generic object service
In this case, you do not specify the object. The workflow relationships are displayed for the object being processed.

What is displayed?
The display of the workflow relationships has two parts. The upper part displays the workflows identified for the object or object type. The lower part displays data on the currently selected work item in the form of a simplified workflow log:
- Steps so far
Workflow Relationships

The workflow steps processed so far are listed under Step name. The steps are linked to the work item preview of the relevant work item. The current agents of the work item are listed under Agents. Click once to go into detailed display of the user data.

- **Information objects addressed so far**
  All the objects and attachments belonging to the workflow are displayed here. Click once to display the information objects.

### Activities

<table>
<thead>
<tr>
<th>Calling...</th>
<th>Path</th>
</tr>
</thead>
<tbody>
<tr>
<td>... Via SAP Easy Access</td>
<td>Tools → Business Workflow → Development → Runtime Tools → Workflows For Object</td>
</tr>
<tr>
<td>... From the Business Workplace [Seite 1368]</td>
<td>Display workflow relationships</td>
</tr>
<tr>
<td>... As a generic object service</td>
<td>System → Workflow → Workflow overview</td>
</tr>
</tbody>
</table>
RFC Monitor

Use
You can use this function to display the log file of the transactional RFC.

Prerequisites

Authorization
You require appropriate authorizations to carry out the actions described above. These are an authorization based on the authorization object S_PROGRAM allowing a program to be executed for the authorization group ‘SWU_DIAG’, and the system administrator authorization (authorization object S_ADMI_FCD) for the network administration.

Features
The log entries are displayed with the target system (log. destination) WORKFLOW_LOCAL_<client>.

- Choose this method of access when you want to follow up errors that have occurred in connection with the feedback from work items to the workflow runtime system after their execution. Depending on the general system load, it may take some time until this feedback is delivered.

- It is essential that you refer to the information on maintaining the logical destination WORKFLOW_LOCAL_<client> in Customizing [Extern].

Activities
To display the log file, choose Tools → Business Workflow → Development → Utilities → Workflow RFC monitor.
Event Trace

Use
All events [Seite 1321] created correctly are logged in the event trace irrespective of whether potential receivers exist.
If a receiver is entered in the event trace, it does not necessarily mean that this receiver was called successfully.

Prerequisites
The event trace is only written if logging has been activated.
To trace any events which may not have been created, you must first activate the event trace and then create the event again.

Features
The following data is logged in the event trace:

Event data
- Triggering object type, triggering object
- Event ID
- Triggering program
- Trigger date and time

Receiver or linkage data
- Receiver function module
- Receiver type
- Receiver instance
- Linkage status

Activating/deactivating the event trace
As well as for activating or deactivating the event trace, you can also use this function for specifying selection criteria so that only certain events are logged.

Displaying the event trace
The event trace display can also be restricted using a selection screen. In addition to the criteria for event data, receiver data and event receiver linkage data, the display can also be restricted to linkages with errors.

Deleting the event trace
The deletion of events can also be controlled using selection criteria. The function does not therefore always delete the entire event trace automatically.
If you mark the field Only display list, the events you select are not deleted, but displayed in a list.

Activities
- To activate/deactivate the event trace, choose:
Event Trace

Tools → Business Workflow → Development → Administration → Event manager → Event trace → Switch event trace on/off

- Select the function Selective tracing on the dialog box that follows to choose selection criteria.

- To display the event trace, choose:

  Tools → Business Workflow → Development → Administration → Event manager → Event trace → Display event trace

- To delete the event trace, choose:

  Tools → Business Workflow → Development → Administration → Event manager → Event trace → Delete event trace
Start Conditions for Workflows

Use
You use this function if you want to define additional conditions for the start of a workflow.

You can only define start conditions for workflows that are started by an event.

Integration
Refer also to the following for workflow start conditions:
- [Event Seite 1321]
- [Starting Tasks Seite 1448]

Features
This function enables the definition of start conditions for a workflow.
This means that the start of a workflow no longer depends only on whether a specific event is created, but also on the occurrence of additional conditions.

A workflow is to be started when a notification of absence has been created and the creator of the notification of absence belongs to a specific group of people.

Selection area
The following selection criteria are available for selecting a particular workflow:
Start Conditions for Workflows

- **Start condition**
  Here you can search for the name of a particular start condition defined for a workflow.

- **Business object**
  Here you can search for workflows with start conditions using a particular object type.

- **Event**
  All the events for the chosen object type, which are linked with workflows as triggering events are offered for selection here.

- **Workflow**
  Here you can search directly for workflows with start conditions.

  
  If a newly defined triggering event [Seite 1206] is not displayed, check that the event linkage was activated when the workflow was defined.

**Definition area**

In this area you can define additional start conditions for workflows or view and change existing start conditions.

The individual lines of a start condition provide the following information:

- **Name of the start condition**
- **Currency**
- **Triggering event of workflow**
- **Start condition**

To **change** the start condition, the name of the condition or the currency, single-click on the relevant area.

  
  The **condition editor** [Seite 1012] is used to determine conditions.

Select to **activate** the start condition (or to **deactivate** it). Select to **delete** the start condition.

**Personal settings**

You can display and edit personal settings by choosing Extras → Settings. In particular, you can set whether the maintenance of start conditions is to be cross-client or client-specific.

  
  In the cross-client view you cannot activate or deactivate start conditions if they are redefined in a client. They are marked with the symbol. Start conditions that are not redefined are **only activated or deactivated for the current client** in the cross-client view.

**Activities**

Choose the following for workflow start conditions: Tools → Business Workflow → Development → Definition Tools → Events → Event Linkages → Workflow Start Conditions.

To create a new start condition, select.
You can also create start conditions directly from the Workflow Builder. To do this, go to the tab page Start in the basic data of the workflow. Click on the cell **Start condition** in the line of the relevant triggering event. This displays the condition editor directly.
Business Workplace: Workflow Functions

Use
You use this part of the Business Workplace if you want to use the functions of SAP Business Workflow. The Business Workplace is the main interface between an end user and the workflow system. All dialog and missed deadline work items to which the user is assigned as a recipient are displayed in the user’s workflow inbox.

As a head of department, you are responsible for approving leave requests. The relevant approval process is implemented using a workflow in your enterprise.

The requests (in the form of work items) appear in your worklist (workflow inbox) and must be rejected or approved there.

The rejected or approved requests (executed work items) are not only returned to the applicants after processing, but are also put into your workflow outbox (under Work items executed by me). You can therefore check the requests you have processed.

Features

Workflow settings
You can configure the workflow functions in the Business Workplace using the personal workflow settings.

Business Workplace screen areas
The Business Workplace has three screen areas, which are used in the following manner by SAP Business Workflow:
Overview tree

The following workflow functions are available under the Inbox node, which is under the initial node Workplace:

- Workflow
  - Grouped according to task
  - Grouped according to content
  - Grouped according to content type
  - Grouped according to sort key
- Overdue entries
- Deadline messages
- Incorrect entries

For information on these functions, refer to Workflow Inbox [Seite 1408].

The following functions are available under the Outbox node, which is under the initial node Workplace:

- Started workflows
- Work items executed by me
- Forwarded work items

For information on these functions, refer to Workflow Outbox [Seite 1441].

The Resubmissions node is located under the initial node Workplace and contains the:

- Workflow resubmissions [Seite 1443]
Worklist
The worklist [Extern] is displayed in the upper right corner of the Business Workplace screen. Depending on whether you are in the workflow inbox, the workflow outbox or the workflow resubmissions, you have various functions available to you, which are described at the respective locations.

Work item preview
In the lower right corner of the Business Workplace screen, a work item selected in the worklist is displayed in a preview [Seite 1445]. Not all the functions of the work item display or the workflow log are available. A user exit [Extern] can be used to configure the work item preview to suit your individual requirements.

Support for context menus
All workflow functions can be called using the relevant context menu.

Workflow Toolbox
SAP Business Workflow's Workflow Toolbox [Seite 1446] enables the user to access workflow functions even during a workflow-driven application transaction.

E-mail notification for new work items
The report RSWUWFML can be used to inform an employee by mail that there is a new work item in their Business Workplace inbox. This function is therefore beneficial to all employees who do not work with their Business Workplace on a daily basis.

Activities
To access the Business Workplace from the SAP Easy Access screen, choose one of the following options:

- Menu → Business Workplace
Work Item

Definition
Object that represents a task or action in the workflow system at runtime.

Use
Work items are subdivided into a specific work item type according to their assignments. The internal processing procedures are controlled via this work item type. The work item type determines which statuses and transitions are valid. Depending on the work item type, some of these work items are displayed in a user's work list. Other work items, on the other hand, are only used and processed internally.

Structure

Work item types displayed in the Business Workplace

<table>
<thead>
<tr>
<th>Work item type</th>
<th>Short text</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>W</td>
<td>Work item with dialog</td>
<td>Runtime representation of single-step tasks that require interaction with the user.</td>
</tr>
<tr>
<td></td>
<td>[Seite 1372]</td>
<td>Runtime representation of user decisions.</td>
</tr>
<tr>
<td>D</td>
<td>Missed deadline [Seite 1377]</td>
<td>Work item for notification of missed deadline.</td>
</tr>
<tr>
<td>A</td>
<td>Work Queue [Seite 1390]</td>
<td>A work queue is a list of objects to be processed once and together in a limited time frame.</td>
</tr>
</tbody>
</table>

Other work item types

<table>
<thead>
<tr>
<th>Work item type</th>
<th>Short text</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>F</td>
<td>Workflow [Seite 1380]</td>
<td>Runtime representation of a multistep task.</td>
</tr>
<tr>
<td>B</td>
<td>Batch item [Seite 1385]</td>
<td>Runtime presentation of a single-step task that runs in the background.</td>
</tr>
<tr>
<td>E</td>
<td>Wait step work item [Seite 1392]</td>
<td>Runtime representation of a wait step in the workflow definition.</td>
</tr>
<tr>
<td>C</td>
<td>Container anchor</td>
<td>This type of work item is required as a special development in the EDI environment [Extern]. It does not normally appear in the workflow environment.</td>
</tr>
</tbody>
</table>

Work items of this type should be regularly deleted or archived.
**Dialog Work Items (Type W)**

**Definition**
Work item that represents a task at runtime that requires interaction with the user.

Since the **user decision** is also represented internally by a task, a dialog work item can also represent a user decision.

When a dialog work item is executed, the underlying object method of the task is called. The deadlines for executing dialog work items are monitored.

**Use**
A dialog work item is displayed with *ready* status in the workflow inbox of the Business Workplace. It is removed from the integrated inboxes of the other agents when the recipient reserves, executes, or processes this work item with other functions.

The database oriented approach used in SAP Business Workflow allows a work item to be **seen** by several recipients equally authorized in organizational terms in their inboxes and **executed** from there. However, only one recipient can actually **reserve** this work item for processing and **execute** it. The work item is then no longer available to any other recipients.

**Integration**
A task represented by a dialog work item can be

- a step in a workflow definition:
  
  In the workflow definition, reference is made to tasks in the **activity** and **user decision** steps.

- started as single steps via an event or in dialog:
  
  Tasks can be started as elementary activities directly in dialog or via a triggering event. These tasks are then also represented by a dialog work item in the workflow inbox.
Status of a Dialog Work Item

The valid statuses for dialog work items (type W) are listed in the following table.

<table>
<thead>
<tr>
<th>Work item status</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>waiting</td>
<td>The work item has been scheduled for its requested start. A work item has this status if it already exists but the requested start specified in the workflow definition has not been reached yet. If it has been set to resubmission. Work items in the waiting status are not displayed in the workflow inbox.</td>
</tr>
<tr>
<td>ready</td>
<td>The work item has been released for execution and appears in the workflow inbox of all recipients.</td>
</tr>
<tr>
<td>reserved</td>
<td>The work item has been received by one of its recipients with the result that its status has changed from ready to reserved. A work item in the reserved status is then displayed to this recipient only. It is no longer displayed in the workflow inboxes of the other recipients.</td>
</tr>
<tr>
<td>In process</td>
<td>The work item is currently being processed by a recipient or in a different mode. A work item also has this status if the work item is waiting for its terminating event. If the user cancelled the method. If the method was ended with a temporary exception for which no subsequent steps have been modeled. The point at which processing is completed cannot be detected by the workflow system in this status. As long as the status of the work item is set to in process, database changes have not been made.</td>
</tr>
<tr>
<td>Executed</td>
<td>The work item is awaiting explicit confirmation of its completion. The work item only has this status if it is necessary to confirm that it has been completed. A work item with executed status can be executed or forwarded several times until it is set to the status done in the Business Workplace. In this way, groupware components are realized in SAP Business Workflow.</td>
</tr>
<tr>
<td>completed</td>
<td>The execution of the work item is completed. The result of the task represented by the work item is correct, i.e. the result modeled in the workflow definition. Work items in the completed status are not displayed in the workflow inbox of the Business Workplace.</td>
</tr>
<tr>
<td>Logically deleted</td>
<td>Execution of the work item is no longer meaningful or required by the workflow logic. A work item changes to the logically deleted status in the following way: Termination in parallel processing branches. When the required number of processing paths has been executed in a fork, the work items in the other paths that have not yet reached the completed status are automatically set to the logically deleted status. Intervention by an administrator. An administrator can only set a work item to the logically deleted status if it has not yet reached the completed status and is not part of a higher-level workflow.</td>
</tr>
</tbody>
</table>
### Status of a Dialog Work Item

<table>
<thead>
<tr>
<th>Status</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Logically deleted</td>
<td>Execution of the work item is no longer meaningful or required by the workflow logic.</td>
</tr>
<tr>
<td></td>
<td>A work item changes to the <em>logically deleted</em> status in the following way:</td>
</tr>
<tr>
<td></td>
<td>- Termination in parallel processing branches</td>
</tr>
<tr>
<td></td>
<td>When the required number of processing paths has been executed in a fork, the work items in</td>
</tr>
<tr>
<td></td>
<td>the other paths that have not yet reached the <em>completed</em> status are automatically set to</td>
</tr>
<tr>
<td></td>
<td>the <em>logically deleted</em> status.</td>
</tr>
<tr>
<td></td>
<td>- Intervention by an administrator</td>
</tr>
<tr>
<td></td>
<td>An administrator can only set a work item to the <em>logically deleted</em> status if it has not</td>
</tr>
<tr>
<td></td>
<td>yet reached the <em>completed</em> status and is not part of a higher-level workflow.</td>
</tr>
<tr>
<td></td>
<td>Work items in the <em>logically deleted</em> status are not displayed in the workflow inbox of the</td>
</tr>
<tr>
<td></td>
<td>Business Workplace.</td>
</tr>
<tr>
<td>error</td>
<td>Execution of the work item was terminated with an error.</td>
</tr>
</tbody>
</table>

In addition to the statuses given above, a work item can be **locked against execution**. This is possible in any status and involves an administrative function which is selected for a work item via the *Change* option.
Status Transitions of a Dialog Work Item

The diagram below shows the possible status transitions that a dialog work item (type W) can undergo:

The arrows are labeled with the functions used by a user to trigger the respective status transition.

**Comments**

For further information, refer to [Status of a Dialog Work Item][1].

Transition from status *waiting*

The transition from the status *waiting* to the status *ready* is performed automatically by the work item manager when the requested start is reached. A workflow system administrator can set a work item to the status *ready* manually.

Transition from status *ready*

From this status, the work item passes either to the status *reserved* or via the status *in process* to the status *completed*.

Transition from status *reserved*

A work item with the status *reserved* can be reset to the status *ready*.

Transition to and from status *in process*

A work item with the status *in process* can be reset to the status *ready*. A workflow system administrator can reset a work item manually. This function is available when changing the work item.

Transition from status *executed*

After confirmation of end of processing, the work item assumes the status *completed*.

---

[1]: Status of a Dialog Work Item [Seite 1373]
Status Transitions of a Dialog Work Item

**Transition from status completed**
Work items with the status *completed* can no longer be set to another status even if a workflow system administrator intervenes.

**Transition from status error**
A workflow system administrator can intervene and set work items with errors to the status *in process* or the status *logically deleted* (possibly after eliminating the error).

**Transition from status logically deleted**
Work items with the status *logically deleted* can no longer be set to another status even if a workflow system administrator intervenes.
Missed Deadline Work Item (Type D)

Definition
Notification of a deadline recipient if the runtime system detects that the deadline for a certain work item has been exceeded.

Use
This work item informs its recipients that a deadline (start or end deadline) of a monitored work item has been exceeded. The recipients are informed by means of a missed deadline work item (type D) in the workflow inbox of the Business Workplace.
When it is executed, this work item displays information on the monitored (and now late) work item.
The text for notifying the recipient is set by default.
When a deadline is monitored for an activity [Seite 1024] or user decision [Seite 1074], it is also possible to enter an individual text in the respective task definition.
Status of a Missed Deadline Work Item

The valid statuses for missed deadline work items (type D) are listed in the following table:

<table>
<thead>
<tr>
<th>Work item status</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Ready</td>
<td>The work item appears in the Business Workplace of the recipients.</td>
</tr>
<tr>
<td>In process</td>
<td>One of the recipients has executed the work item</td>
</tr>
<tr>
<td>Completed</td>
<td>The execution of the work item is completed. Work items in the <em>completed</em> status are not displayed in the workflow inbox of the Business Workplace.</td>
</tr>
</tbody>
</table>
Status Transitions of a Missed Deadline Work Item

The following status transitions are possible for missed deadline work items (type D):

**Transition to and from status in process**

The status of a missed deadline work item changes from *ready* to *in process* when it is executed by a user. Execution of a missed deadline work item displays the most important information on the monitored work item. The missed deadline work item remains in the *in process* status until end of processing is confirmed explicitly. Until then, the work item can be executed or forwarded several times.

**Transition to completed status**

The status of the missed deadline work item changes to the *completed* status when completion of processing has been confirmed explicitly.
Work Queue Work Item (Type A)

Definition
Work item that represents a work queue. A work queue comprises a list of objects that are to be processed once together within a specified period. The work queue serves as a framework for the individual entries to be processed and manages the list of objects to be processed including their statuses and the tasks to be performed on them. Work queue work items are displayed in the Business Workplace. The work item status indicates the overall processing status of the work queue.

Use
Once you have created the work queue work item and you know its work item ID, you have the following options:
- Process the work queue within a workflow.
- Control the release, processing and status evaluation of the work queue with function modules.
- Control the release and processing of the work queue by processing the work queue work item directly.

Integration
To create the work queue work item from the list, you call the function module SWZ_AI_CREATE.
Status of a Work Queue Work Item

A work queue work item can have the following statuses:

<table>
<thead>
<tr>
<th>Work item status</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>waiting</td>
<td>The work queue has been created but cannot be released yet.</td>
</tr>
<tr>
<td>ready</td>
<td>The work queue work item (type A) is displayed in the Business Workplace of the recipients passed in the table AGENTS of the function module SWZ_AI_CREATE.</td>
</tr>
<tr>
<td></td>
<td>If the work queue is to be processed via a workflow and therefore not appear as a separate work item in any Business Workplace, a non-existent user must be specified in the table when the work queue is created.</td>
</tr>
<tr>
<td></td>
<td>The work queue must be reserved for further processing.</td>
</tr>
<tr>
<td>reserved</td>
<td>A person with release authorization has reserved the work queue.</td>
</tr>
<tr>
<td>in process</td>
<td>Work queue processing has begun.</td>
</tr>
<tr>
<td></td>
<td>− Dialog work items (type W) have been created for entries with dialog.</td>
</tr>
<tr>
<td></td>
<td>− For entries without dialog marked accordingly, the methods have been executed directly.</td>
</tr>
<tr>
<td>completed</td>
<td>All lines in the work queue have the status COMPLETED or CANCELLED and have reported back accordingly to the work queue.</td>
</tr>
<tr>
<td></td>
<td>A work queue work item (type A) that assumes this status automatically creates the event created.</td>
</tr>
<tr>
<td>logically deleted</td>
<td>Further processing of the work queue is invalid (and therefore no longer possible).</td>
</tr>
<tr>
<td>error</td>
<td>At least one line of the work queue has the status error.</td>
</tr>
</tbody>
</table>
Workflow Work Item (Type F)

Definition
Work item that represents a multistep task at runtime.

Use
For every multistep task started there is one type F work item. The workflow log and the workflow container can be accessed:
- For error diagnosis and error correction if no work items are displayed
- for information on steps of a workflow (including their current agents, notes, and ad hoc objects) already processed
- for modifying an ongoing workflow by changing the workflow container
- for reporting on completed processes
Type F work items are not displayed in the Business Workplace but can be found using the work item selection [Seite 1490].

Structure
A workflow consists of a sequence of work items that are executed by agents or the system. The work items represent the steps in the workflow definition that refer to a particular task. These are steps of the types activity [Seite 1024] or user decision [Seite 1074].

Integration

The Work Item Manager manages the processing of work items and monitors deadlines. To automate workflow processes, activities in the workflow can also refer to object methods which run in the
background. If this is the case, the work item manager initiates the calling of the background processes. The work items whose execution requires dialog can be accessed by the selected agents (determined from the organizational model and role resolution) from their worklists in order to select them for processing. This worklist is displayed and managed in the workflow inbox of the Business Workplace. [Seite 1368].
Status of a Workflow Work Item

The valid statuses for workflow work items (type F) are listed in the following table:

<table>
<thead>
<tr>
<th>Work item status</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Waiting</td>
<td>The (sub) workflow is scheduled but its requested start has not yet been reached.</td>
</tr>
<tr>
<td>Ready</td>
<td>Temporary interim status</td>
</tr>
<tr>
<td>In process</td>
<td>Execution of the (sub) workflow has begun.</td>
</tr>
<tr>
<td>Completed</td>
<td>The end of the (sub) workflow has been reached.</td>
</tr>
<tr>
<td>Logically deleted</td>
<td>Execution of the (sub) workflow is no longer required or meaningful. A status of a workflow changes to <em>logically deleted</em> in the following way:</td>
</tr>
<tr>
<td></td>
<td>• Intervention by an administrator.</td>
</tr>
<tr>
<td></td>
<td>• Termination in parallel processing branches</td>
</tr>
<tr>
<td></td>
<td>When the required number of processing paths has been executed in a fork, the work items in the other paths that have not yet reached the <em>completed</em> status are set to the <em>logically deleted</em> status.</td>
</tr>
<tr>
<td></td>
<td>A workflow set to the <em>logically deleted</em> status is recursively scanned for dialog and workflow work items (type W or F) that do not yet have the status &quot;completed&quot;. These work items are then also set to the status <em>logically deleted</em>.</td>
</tr>
<tr>
<td></td>
<td>A work item changed from the <em>executed</em> status to the <em>logically deleted</em> status may already have caused database changes or other actions (generate event, send notification). These changes are <em>not</em> canceled automatically (compensated), but are recorded in the log.</td>
</tr>
<tr>
<td>Errors</td>
<td>An error occurred in workflow control.</td>
</tr>
</tbody>
</table>

Workflows and their statuses are **not** displayed in the Business Workplace. Information about the workflow can be accessed via the subordinate dialog work items.
Status Transitions of a Workflow Work Item

The statuses of a workflow work item and the permissible status transitions always concern an entire workflow. A step (activity) in the workflow definition can reference both a task and another workflow. A workflow can therefore also contain subordinate subworkflow items.

Comments

Transition from status waiting
The work item manager automatically carries out the transition from the waiting status to the ready status when the requested start date/time of the workflow has been reached. A workflow system administrator can set a work item to the status ready manually.

Transition from status ready
The ready status is a temporary interim status because it only exists until the first work item of the workflow has been created.

Transition to and from status in process
The status of the workflow changes to in process as soon as the first work item of this workflow has been created. The workflow remains in this status until the entire workflow definition has been processed.

Transition to incorrect status
An error occurs during workflow control or coordination.

Role resolution for determining an agent does not return a result which can be used.
A workflow system administrator can intervene and set workflows with errors to the status *in process* or the status *logically deleted* (possibly after eliminating the error). If a workflow is incorrect, the responsible workflow system administrator specified either globally in Customizing or in the basic data of each workflow definition is notified by mail.

This status does *not* mean that a dialog work item of this workflow has the *incorrect* status.

**Transition to completed status**

A workflow is set to the status *completed* when the last step of the relevant workflow has been completed.
Background Work Item (Type B)

Definition
Work item that represents a single-step task at runtime whose execution does not require a dialog and, therefore, can be controlled automatically by the system.

Integration
Type B work items are not displayed in the Business Workplace.
Status of a Background Work Item

The valid statuses for background work items (type B) are listed in the following table.

<table>
<thead>
<tr>
<th>Work item status</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Waiting</td>
<td>The work item has been scheduled for its requested start. A work item has this status after it has been created until the requested start specified in the workflow definition is reached.</td>
</tr>
<tr>
<td>Ready</td>
<td>Temporary intermediate status of a background work item. The system calls the associated object method as soon as a background work item can be processed. The status of the background work item then changes to in process immediately.</td>
</tr>
<tr>
<td>In process</td>
<td>The work item is currently being processed. A work item also has this status</td>
</tr>
<tr>
<td></td>
<td>• if the method was left with a temporary exception. In this case, special Error Handling for Background Work Items with Temporary Errors [Seite 1388] is carried out.</td>
</tr>
<tr>
<td></td>
<td>• if the work item is waiting for its terminating event. The point at which processing is completed cannot be detected by the workflow system in this status.</td>
</tr>
<tr>
<td>Completed</td>
<td>The execution of the work item is completed. The result of the task represented by the work item is correct, i.e. the result modeled in the workflow definition.</td>
</tr>
<tr>
<td>Logically deleted</td>
<td>Execution of the work item with regard to the process logic is no longer meaningful or necessary for the process to continue. A work item changes to the logically deleted status in the following way:</td>
</tr>
<tr>
<td></td>
<td>• Termination in parallel processing branches</td>
</tr>
<tr>
<td></td>
<td>When the required number of processing paths has been executed in a fork, the work items in the other paths that have not yet reached the completed status are automatically set to the logically deleted status.</td>
</tr>
<tr>
<td></td>
<td>• Intervention by an administrator</td>
</tr>
<tr>
<td></td>
<td>The administrator can only set a work item to the logically deleted status if it has not yet reached the completed status and is not part of a higher-level workflow.</td>
</tr>
<tr>
<td></td>
<td>A work item with the logically deleted status may have caused database changes or other actions (generate event, send notification). These changes are not canceled automatically (compensated).</td>
</tr>
<tr>
<td>error</td>
<td>Execution of the work item was terminated with an error.</td>
</tr>
</tbody>
</table>

In addition to the statuses given above, a work item can be locked against execution. This is possible in any status and involves an administrative function which is selected for a work item via the Change option.
Status of a Background Work Item
Error Handling for Background Work Items

Use
Error handling for background work items must be carried out by the workflow system
- because background work items (with errors) are not displayed in the Business Workplace and are therefore detected too late, if at all.
- because background work items that are temporarily incorrect, for which another execution attempt can (theoretically) be successful, cannot be restarted by a user.

(Temporarily incorrect work items are work items whose method was terminated with a temporary exception for which no subsequent step has been defined.)

Features
In Customizing for the workflow system, you can define how often the workflow system attempts to restart a temporarily incorrect work item. You can also define the interval between two repetition attempts and activate the monitoring function.
Monitoring is also activated as part of automatic Customizing [Extern] (Tools → Business Workflow → Development → Utilities → Customizing).
You can also specify the maximum number of repetition attempts (“repetition counter”) separately for each individual background step in the respective workflow definition. This setting overrides the value in Customizing if it is greater than zero.
If an error (method exception) occurs when background work items are executed by the workflow system, the system responds as follows depending on the type of exception and error modeling settings:

<table>
<thead>
<tr>
<th>In the workflow definition...</th>
<th>The exception is defined for the object method as a ...</th>
<th>System or application error</th>
</tr>
</thead>
<tbody>
<tr>
<td>a subsequent step is modeled for the exception.</td>
<td>temporary error</td>
<td>The step has status completed and the modeled subsequent step is executed.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Background work items for which a repetition counter is explicitly modeled in the workflow definition are restarted first by the system. If all of the attempts are unsuccessful, the status of the work item changes to completed and the modeled subsequent step is executed.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>The step has status completed and the modeled subsequent step is executed.</td>
</tr>
</tbody>
</table>
### Error Handling for Background Work Items

The step is not yet completed. The respective work item retains the status *in process*. *Background work items* are restarted by the system. The number of repetition attempts is determined either by the repetition counter in the step definition or - if this is equal to zero - by the repetition counter set in Customizing. If all of the attempts are unsuccessful, the work item status changes to *incorrect*.

<table>
<thead>
<tr>
<th>Step Status</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>No subsequent step is modeled for the exception</td>
<td>The step is not yet completed. The respective work item retains the status <em>in process</em>. <em>Background work items</em> are restarted by the system. The number of repetition attempts is determined either by the repetition counter in the step definition or - if this is equal to zero - by the repetition counter set in Customizing. If all of the attempts are unsuccessful, the work item status changes to <em>incorrect</em>.</td>
</tr>
<tr>
<td>Workflow and work item assume the error status.</td>
<td></td>
</tr>
</tbody>
</table>

### Processing Incorrect Work Items

The workflow system sends a mail to the relevant system administrator for every background work item with the status *incorrect*.

### Processing Work Items That Have Been Started

The workflow system determines all of the background work items that have been *in process* for longer than 30 minutes. An error message is then sent to the workflow system administrator for all of these background work items, since the system assumes that processing has been cancelled. However, this does not necessarily mean that an error has occurred.
Wait Step Work Item (Type E)

Definition
Work item that represents a wait step or a workflow at runtime, which is waiting for an event to occur.

Use
Type E work items are not displayed in the Business Workplace.
Work Item with Express Notification

Use
When the system creates a work item with priority 1, each of its recipients [Extern] receives an express notification (dialog box with appropriate text) on the screen. The recipient can call the Business Workplace directly from the express message. Excluded agents do not receive a message.

Constraints
The system does not create an express message
- if the user processes the work item immediately due to advance with immediate dialog [Seite 1453].
- if the work item represents a general task [Extern] that is not restricted to certain agents.
- if the work item was forwarded.

The system only creates express notifications when a work item is created. An express notification is not created for the new recipients of priority 1 work items that are forwarded.

If one of the selected agents processes the work item, the other agents still receive an express message.

Features
Express messages are only sent for dialog steps as soon as the system has created them with the ready status. If the work item is created first with the waiting status because its requested start has not been reached yet, the express message is not sent until the status changes from waiting to ready.

Activities
How is the priority set?
The priority of a work item can be determined for steps that require a dialog with the user. It is defined in the tab page Miscellaneous of the step definition.
Workflow Settings

Use
You use the workflow settings to maintain the special workflow functions in the Business Workplace.

Features
The following functions are available:

- Personal settings [Seite 1395]
- Display organizational assignment [Seite 1397]
- Refresh organizational environment [Seite 1397]
- Adopt substitution [Seite 1400]
- End substitution [Seite 1400]
- Maintain substitute [Seite 1400]
- Activate substitute [Seite 1400]
- Adopt view [Seite 1404]
- Exit view [Seite 1404]

Activities
You can access the workflow settings within the Business Workplace [Seite 1368] via Settings → Workflow settings.
Personal Settings

Use
The personal settings for workflow enable you to adapt the runtime system to suit your requirements.

Features

Work item display

<table>
<thead>
<tr>
<th>Setting</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>User view with ActiveX (32-bit)</td>
<td>Both of these settings define the work item display as the default. Note that the ActiveX variant is only available on 32-bit platforms.</td>
</tr>
<tr>
<td>User view without ActiveX</td>
<td>This setting defines the technical work item display as the default.</td>
</tr>
</tbody>
</table>

Workflow log

<table>
<thead>
<tr>
<th>Setting</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>User view with ActiveX (32-bit)</td>
<td>User view of the workflow log without any technical aspects. This view uses ActiveX controls and is therefore only available on 32-bit platforms.</td>
</tr>
<tr>
<td>User view without ActiveX</td>
<td>User view of the workflow log without any technical aspects. This view uses the hierarchical list display of the ABAP List Viewer.</td>
</tr>
<tr>
<td>Technical view</td>
<td>In addition to the main semantic information, the technical view of the workflow log also displays technical numbers and texts. The technical view is intended primarily for system administrators.</td>
</tr>
</tbody>
</table>

Further settings

<table>
<thead>
<tr>
<th>Setting</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Display work item texts in logon language</td>
<td>All work item texts in the Business Workplace are always displayed in the user’s logon language.</td>
</tr>
<tr>
<td></td>
<td>If the user’s logon language is different to the language in which the work item was created, additional database accesses are required that can delay display of the Business Workplace.</td>
</tr>
<tr>
<td>Enable forwarding of work items to several users</td>
<td>The work item can be forwarded to several agents.</td>
</tr>
<tr>
<td></td>
<td>The term “agent” is used in a broad sense in this context. You specify the agent as an organizational object (organizational unit, job, position, user) when forwarding the work item. This organizational object can consist of several persons. For further information, refer to Forwarding.</td>
</tr>
<tr>
<td>(Double)-clicking on an object displays the object in the same window.</td>
<td>You can choose an object in the work item display, which is then displayed in the current window.</td>
</tr>
</tbody>
</table>
(Double)-clicking on an object displays the object in the same window.

<table>
<thead>
<tr>
<th>Activities</th>
</tr>
</thead>
</table>

You can choose an object in the work item display, which is then displayed in the current session window and replaces the work item display. When you exit the object display, you return to the work item display.

If you do not set the indicator, a new session is created to display the object.

This setting does not apply if you work with the SAP GUI for HTML.

<table>
<thead>
<tr>
<th>No tips &amp; tricks in workplace</th>
</tr>
</thead>
</table>

The Business Workplace [Seite Error! Bookmark not defined.] includes tips and tricks for working with work items and workflows in the work item preview. If you set this indicator, the tips and tricks are not displayed.

<table>
<thead>
<tr>
<th>No HTML in execution of decision tasks.</th>
</tr>
</thead>
</table>

You use this indicator to decide whether the execution of decision tasks is displayed in HTML or not. Deactivating the HTML display may make sense, for example, if you have problems displaying HTML documents generally because of the settings in your local HTML browser.

Activities

To change the personal settings for workflow, you choose Settings → Workflow settings → Personal settings in the Business Workplace.

The settings are saved as user parameters and take effect the next time the work item display or workflow log is called.

You can change the standard values for these personal workflow settings in Customizing for SAP Business Workflow. These standard values are always used if users have not maintained settings of their own.
Organizational Assignment of a User

Use
The worklist displayed is formatted specifically for the user who is logged on. You can query this user's organizational assignment.

Prerequisites
The functionality described depends on a setting in the PD system table [Seite 1533].

Features
The organizational assignment of a user is buffered and read at each new logon or once a day (but not each time the Business Workplace is called).
If the organizational assignment of a user changes while the user is logged on and must be updated, you choose Settings → Workflow settings → Refresh organizational environment in the Business Workplace.

Activities
To view current information on the assignment of the user logged on in the organizational plan of the enterprise, choose Settings → Workflow settings → Display organizational assignment in the Business Workplace.

The following information is displayed for the user logged on:

- The name maintained in the user address
- The organizational unit
- The position the user occupies
- The job describing the position

By double-clicking on an entry, you can display the staff assignments or the job description.
Substitutes for Workflows

Use

*SAP Business Workflow* implements a substitution arrangement to deal with a user's duties in their absence.

Prerequisites

The automatic appearance of work items in the substitute's worklist is dependent on a setting in the PD system table [Seite 1533]. It is possible to define which objects can be entered as position-related substitutes in Administration of Organizational Management [Seite 1530].

Features

Who is involved?

Substitution always involves two users:

- One user specifies a substitute: User B
- The other user (the substitute) 'adopts' the substitution: User A

The substitute can process “foreign” work items for the duration of the substitution. It is irrelevant whether they are assigned to the underlying single-step tasks as a possible agent.

How does substitution work?

Substitution works in two ways:

5. User A adopts substitution for user B and for the duration of the substitution sees only the work items seen previously by B in their workflow inbox.

   User B must have entered user A as their substitute for a particular period without activating the substitution. During this period, A can adopt substitution for B at any time without having to confer with B.

   Instead of their own worklist, user A is displayed user B's worklist. B's name is in the column *Substitution for* in the worklist, as long as this column is included in the current configuration of B's Business Workplace.

6. User A automatically sees not only the work items previously seen by employee B, but also their own work items in their Business Workplace. This also applies to all work items generated for B in the future ("automatic forwarding").

   User B must enter user A as substitute and activate the substitution. No further action on the part of user A is required for this kind of substitution.

   A can continue working as usual. They see a Business Workplace to which B's work items are added dynamically. A can recognize these work items by the fact that B's name is in the column *Substitution for*, as long as this column is included in the current configuration.

In both cases, user A can perform operations on these "foreign" work items within the scope of the substitute profile assigned to them by B. It is irrelevant whether A is assigned to the underlying single-step tasks as a possible agent.

User B can continue working without any restrictions in both cases.
Activities
You can access substitute maintenance in the Business Workplace [Seite 1368] by choosing Settings → Workflow settings. For information on the individual functions, refer to Maintaining and Activating Substitutes [Seite 1400]
Maintaining and Activating Substitutes

Maintaining substitutes

3. In the Business Workplace [Seite 1368], choose Settings → Workflow settings → Maintain substitute.

   The Maintain substitute dialog box appears.

   To facilitate maintenance of the substitution, the Personal substitutes entry and the positions you occupy are displayed in the hierarchy on the Maintain substitutes screen.

   You specify your substitute either as a personal substitute or as a position-related substitute.

   - **Personal substitute**
     A personal substitute can see and execute all your work items, including those assigned to you via a personal agent assignment.
     You can only specify one other user as a personal substitute.

   - **Position-related substitute**
     A position-related substitute can only see and execute the work items you have received on the basis of agent assignment at the level of position, job or organizational unit.
     You can enter another position or a user as a position-related substitute.

12. Position the cursor either on the entry Personal substitutes or on the relevant position, and select Create substitute.

   You can maintain existing entries by double-clicking on the relevant entry.

   If you want to maintain a position-related substitution, decide whether you want another position or another user as a substitute. Select either position or user as the substitute type and specify the number or the user name of the substitute.

13. Specify the validity period for the substitution on the detail screen for substitution.

   Only within this period can the substitute adopt the substitution.

14. Specify a substitute profile in the dialog box Detail screen substitution.

   Irrespective of whether you have created your substitute as a personal or position-related substitute, you can limit the scope of the work items displayed to your substitute by specifying a substitute profile.

15. Select the field Substitution active if applicable.

   You must activate the substitution if you want work items to be visible for the substitute automatically from now on. If you do not activate the substitution, the substitute must adopt the substitution explicitly to see your work items.

16. Save your entries and exit substitute maintenance.
Activating substitutes
5. In the Business Workplace, choose Settings → Workflow settings → Activate substitute.
   The Activate substitute dialog box appears.
6. Select the substitutes that you want to activate, and choose the function Activate.

Deactivating substitutes
5. In the Business Workplace, choose Settings → Workflow settings → Activate substitute.
   The Activate substitute dialog box appears.
6. Select the substitutes that you want to deactivate, and choose the function Deactivate.

Adopting substitution
7. In the Business Workplace, choose Settings → Workflow settings → Adopt substitution.
   The Choose substitution dialog box appears.
8. Select the user(s) for which you want to adopt substitution.
9. Exit the dialog box.

Ending substitution
3. In the Business Workplace, choose Settings → Workflow settings → End substitution.
   The substitution is ended.
Views

Use
The Business Workplace [Seite 1368] provides various views on the work items displayed in the workflow inbox.

Using an appropriate view, a superior can "see" and process the work items of their employees.

Choosing a particular view also gives you the opportunity to see work items of other users in your workflow inbox and process them with full functionality, although the underlying tasks are not organizationally assigned to you.

Prerequisites
Views only require action by an employee: This employee chooses a view from a catalog of defined views. The employee must have the authorization required to choose a particular view.

Authorizations
To choose a particular view, you require a corresponding authorization based on authorization object S_WF_LVIEW.

To maintain a particular view, you must have the relevant authorization. This is an authorization based on the authorization object S_TABU_DIS to maintain tables for authorization group SWES.

Features
Views are always based on an evaluation path [Seite 1534] starting from the employee who wishes to adopt a view and leading to the employees whose inboxes could be viewed. The employee adopts a view by selecting another employee from the result list of the evaluation.

Activities
The activities associated with this function include:

- Adopting and Exiting Views [Seite 1404]
- Maintaining Views [Seite 1403]
Maintaining Views

Prerequisites
A view [Seite 1402] is always based on an evaluation path [Seite 1534]. This evaluation path describes which relationships are traced from the user who wants to adopt the view to the users whose Business Workplaces can be viewed.
You can use one of the evaluation paths available in the system. If there are no suitable evaluation paths, you can define an evaluation path [Seite 1535] of your own.

⚠️
Different views can only be maintained as a customer setting.

Procedure
15. To call table maintenance for views, choose Tools → Business Workflow → Development → Definition tools → Worklist client → Maintain views.

This displays the screen Change View "View for Maintaining Views": Overview".

Maintaining views is an activity that is described in the Implementation Guide and can be performed in Customizing.

16. Create a new view. To do this, choose the function New entries.

17. Assign a unique name to the view.

18. Specify an evaluation path.

19. Describe the view with a long text.

20. Specify a start evaluation path.

This selection is optional. The start evaluation path is used to get an initial selection of objects, which is then evaluated further via the first evaluation path.

21. Select Save.
Adopting and Exiting Views

Procedure

Adopting views

7. To adopt a view as the standard view (= view on your own work items), choose Settings → Workflow settings → Adopt view in the Business Workplace.

8. Choose a view.

9. From the result list of this selection, choose the object whose workflow inbox you want to view.

Exiting views

A view is only active while the Business Workplace is displayed. The next time you call the Business Workplace, you are asked if you wish to adopt the view previously set.

To return to the standard view when working in the workflow inbox of the Business Workplace, choose Settings → Workflow settings → Exit view.
Dynamic Columns for the Business Workplace

Use
Up to 6 columns in the workflow inbox of the Business Workplace [Seite 1368] can be filled on a task-specific basis with contents that are determined dynamically at runtime. The standard functions for filtering, sorting, and grouping are available for these columns. Please compare with Selectable Columns for the Business Workplace [Seite 1406].

Features
If you want to include one of the “dynamic columns” into the workflow inbox display, you must specify an element from the task container for each task, from which the content of the column is established at runtime.

Work items that belong to different tasks are then also displayed with different information. Work items that belong to tasks for which this functionality is not used are displayed with a blank entry.

Activities
The column contents are maintained via Tools → Business Workflow → Development → Definition tools → Worklist client → Dynamic columns for worklist.

You can also define the column headings. These headings are displayed if all of the work items displayed in the Business Workplace refer to the same task.
Selectable Columns for the Business Workplace: Workflow

The columns displayed essentially determine the appearance and information content of the workflow inbox of the Business Workplace. Detailed knowledge of the columns is also required to make full use of the filter and grouping criteria.

You can determine the selection of columns via display variants. The following columns are available:

<table>
<thead>
<tr>
<th>Column</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>Work item is executable</td>
<td>Indicator denoting that the work item is executable. (●).</td>
</tr>
<tr>
<td>Work item title</td>
<td>Title of work item.</td>
</tr>
<tr>
<td>Status indicator</td>
<td>Status of work item. In symbol form.</td>
</tr>
<tr>
<td>Creation date</td>
<td>Date when the work item was created with the status ready or waiting for the first time. A work item is only created with status waiting if a requested start was declared for the work item and the work item is created before the requested start.</td>
</tr>
<tr>
<td>Creation time</td>
<td>Creation time of a work item.</td>
</tr>
<tr>
<td>Priority</td>
<td>Priority of the work item.</td>
</tr>
<tr>
<td>Attachments exist</td>
<td>Attachments are shown with the symbol.</td>
</tr>
<tr>
<td>End of processing must be confirmed</td>
<td>Indicator denoting that the end of processing must be confirmed explicitly. (●).</td>
</tr>
<tr>
<td>Work item overdue</td>
<td>Indicator denoting that a deadline has been missed for the work item. (●)</td>
</tr>
<tr>
<td>ID</td>
<td>Unique number of a work item, which is assigned internally by the system.</td>
</tr>
<tr>
<td>Type</td>
<td>Type of work item. In abbreviation form.</td>
</tr>
<tr>
<td>Task</td>
<td>Identification for the task represented by the work item (for example TS00008323).</td>
</tr>
<tr>
<td>Technical status</td>
<td>Status of work item. Technical name.</td>
</tr>
<tr>
<td>Work item type</td>
<td>Type of work item. Name</td>
</tr>
<tr>
<td>Task name</td>
<td>Name of task</td>
</tr>
<tr>
<td>Work item status text</td>
<td>Status of work item. Name</td>
</tr>
</tbody>
</table>
| Deadline status                             | The deadline status specifies whether one of the deadlines has been missed. The possible values in this column are therefore:  
  • None  
  • Latest start  
  • Requested end  
  • Latest end  
  • Other |
<p>| Current agent                               | Name of the user who last reserved or processed the work item.          |</p>
<table>
<thead>
<tr>
<th align="left"><strong>Selectable Columns for the Business Workplace: Workflow</strong></th>
</tr>
</thead>
<tbody>
<tr>
<td align="left"><strong>Latest end date</strong></td>
</tr>
<tr>
<td align="left"><strong>Latest end time</strong></td>
</tr>
<tr>
<td align="left"><strong>Forwarder</strong></td>
</tr>
<tr>
<td align="left"><strong>Substitution for</strong></td>
</tr>
<tr>
<td align="left"><strong>Work item content</strong></td>
</tr>
<tr>
<td align="left"><strong>Group object</strong></td>
</tr>
<tr>
<td align="left"><strong>Execution can be rejected</strong></td>
</tr>
<tr>
<td align="left"><strong>Dynamic columns</strong></td>
</tr>
</tbody>
</table>
Workflow Inbox

Use
The worklist of the user currently logged on to the Business Workplace is displayed in the workflow inbox.

Integration
As is the case for the workflow resubmissions [Seite 1443] and the workflow outbox [Seite 1441], the workflow inbox is an integral part of the Business Workplace.

Features

Views in the workflow inbox
A user's worklist can be displayed as an overview or according to the following grouping criteria:

<table>
<thead>
<tr>
<th>Grouped according to task</th>
<th>The work items are grouped according to the tasks to which they belong.</th>
</tr>
</thead>
<tbody>
<tr>
<td>Grouped according to content</td>
<td>The work items are grouped according to the object instances to which they belong.</td>
</tr>
<tr>
<td>Grouped according to content type</td>
<td>The work items are grouped according to the object types to which they belong.</td>
</tr>
<tr>
<td>Grouped according to sort key</td>
<td>The work items are grouped according to sort keys. Please refer to Grouping According to Sort Keys [Seite 1432].</td>
</tr>
</tbody>
</table>

You can also choose from the following views:
- Overdue entries
- Deadline messages
- Incorrect entries

Workflow Functions
The screen area in which the worklist is displayed has a toolbar containing the workflow functions and functions for configuring the list. For information about these functions, refer to the documentation on the ABAP List Viewer [Extern].

The workflow functions can be accessed in the Business Workplace via the toolbar or the relevant context menu (right-hand mouse button). The following functions can be executed on work items:

- **Update**
  This function updates the worklist [Extern] of the user currently logged on to the Business Workplace.

- **Execute**
  To be executed, a work item must have either the status **ready** or the status **in process**.

  When a dialog work item [Extern] is executed, the object method to which the single-step task for this work item refers is carried out.

  When a missed deadline work item [Extern] is executed, the most important information on the monitored work item is displayed.

- **Display work item**
  This function goes directly to the work item display [Seite 1411].
• **Reserve** (dialog work items only)
  This reserves a work item for execution by the end user in question. The work item must have the status *ready*. This work item is then no longer visible to the other recipients who could previously see it in status *ready*. The status of the work item changes from *ready* to *reserved*.

• **Replace** (dialog work items only)
  Replacing releases the work item from its reserved status and makes it available again to all of the original recipients. The status of the work item changes from *reserved* back to *ready*. The status of the work item must be *reserved* before it can be replaced.

• **Forward**
  [Forwarding [Seite 1435]] passes a work item on to another employee for execution.

• **Resubmit**
  If a user chooses this function, the selected work item is placed in [workflow resubmissions [Seite 1443]].

• **Display workflow log**
  This function displays the [workflow log [Seite 1420]].

• **Manage attachments** ... (Functions for [attachment management [Seite 1429]], dialog work items only)
  - Display attachments
  - Create attachments
  - Change attachments
  - Delete attachments

• **More functions**
  - [Set to 'Done' [Seite 1440]]
  - [Reject execution [Seite 1430]]
  - [Execute together [Seite 1431]]
  - [Change priority [Seite 1433]]
  - [Send mail [Seite 1425]]
  - [Change work item [Seite 1438]]

• **Environment**
  - *Display objects in workflow* (dialog work items only)
  - [Display workflow relationships [Seite 1436]]
  - [Start Workflow [Seite 1450]]
  - [Create link [Seite 1434]]
Activities
You can access the workflow inbox in the tree on the left in the Business Workplace via Workplace → Inbox.
Work Item Display: Standard View

Use
The objective of the work item display is to display all the information and activities that are relevant to an end user in the environment of the displayed work item in a clear and concise manner. The work item display supports the agent not only in the processing of the current work item but also in the compilation of an activity list, which then functions as the agent's personal worklist.

Integration
A technical work item display is available for work items of all other types. You can make this technical work item display standard for dialog work items as well. The standard view of the work item display can be shown with or without ActiveX. You make this setting for the work item display variant in your personal workflow settings.

Features
The work item display is primarily designed for displaying dialog work items. It contains details about deadlines, statuses, agents, attachments and linked objects for a work item. The work item display also enables an end user - providing they have the relevant authorization - to compile an activity list as their personal worklist.

The work item display has three tab pages (Basic data, Activities, and Available objects).

- **Tab page Basic data**
- **Tab page Activities**
- **Tab page Available objects**
- **Customer-defined tab page for work item display**

You can define another tab page, which is then displayed as the first tab page when the work item display is called.

Application toolbar functions

- **Execute**

  To be executed, a work item must have either the status ready or the status in process.

  When a dialog work item is executed, the object method to which the single-step task for this work item refers is carried out.

  When missed deadline work items are executed, the most important information on the monitored work item is displayed.

- **Display last message**

  The return code that was returned to the workflow system after the object method was executed can be retrieved for processed work items using the Messages function.

- **Forward**

  Forwarding passes a work item on to another employee for execution.

- **Resubmit**
Work Item Display: Standard View

If a user chooses this function, the selected work item is placed in workflow resubmissions [Seite 1443].

- **Change priority**
  Refer to Changing Priorities [Seite 1433].

- **Change deadlines**
  Refer to Changing Deadlines [Seite 1428].

- **Display/create/change attachments**
  For attachments, refer to Attachment Management [Seite 1429].

- **Reserve** (dialog work items only)
  This reserves a work item for execution by the end user in question. The work item must have the status *ready*. This work item is then no longer visible to the other recipients who could previously see it in status *ready*. The status of the work item changes from *ready* to *reserved*.

- **Replace** (dialog work items only)
  Replacing releases the work item from its reserved status and makes it available again to all of the original recipients. The status of the work item changes from *reserved* back to *ready*. The status of the work item must be *reserved* before it can be replaced.

- **Mail**
  Refer to Send Mail [Seite 1425].

- **Display workflow log**
  This function displays the workflow log [Seite 1420].

- **Display graphical workflow log**
  This function displays the graphical workflow log [Seite 1424].

**Additional functions in the menus**

**Work Item**

- **End resubmission**
  The work item is put back into the workflow inbox. It has the status *reserved*.

- **Create link** [Seite 1434]
- **Reject execution** [Seite 1430]
- **Set to ‘Done’** [Seite 1440]

**Goto**

- **Methods**
  You can use this function to execute the defined secondary methods [Extern] of the work item displayed from the work item display.

  A prerequisite for this is that the work item represents an activity that is part of a workflow. At least one secondary method must be defined for this activity.
Work Item Display: Standard View

- **Workflow description**
  The description text of the superordinate multistep task, the "process description", is displayed.
  The work item must be part of a workflow and a description text must be maintained for the workflow.

- **Technical work item display [Seite 1418]**

**Extras**

- **Displaying tasks**
  This function can be used to display the definition of the task represented by the work item. Refer to [Definition of a Single-Step Task [Seite 1175]](##) and [Definition of a Multistep Task [Seite 1194]].

- **Technical data**
  Technical data about the work item, for example its ID, its texts or the ID of the superordinate work items are displayed.

- **Organizational Assignment [Seite 1397]**
- **Displaying Agents [Seite 1427]**

**Activities**

You can access the work item display by:

- Selecting a work item in the Business Workplace and choosing 📊.
- Double-clicking one of the entries for a step in the workflow log.
- Selecting an entry that does not represent a workflow in the [work item selection [Seite 1490]](##) hit list displayed. (If you choose a workflow, the workflow log is displayed.)
Tab Page Basic Data

Use
The information on this tab page of the work item display [Seite 1411] is mostly self-explanatory. Except for the priority, you cannot make any changes here.

Features

Deadlines
These are the deadlines monitored by the runtime system. (Deadlines that are not set are displayed without a date.)
- Start by (latest start)
- End by (latest end)
Depending on whether the work item represents a step in the workflow or a task, the deadlines were either specified when this step was defined in the workflow definition or when the task was started online. To display all of the deadlines of the work item, choose Work item → Deadlines.

Further Information
- Forwarded by
  If the work item was forwarded to you, the name of the forwarder is entered here.
- Priority
  The priority of a work item is derived from the definition of this step in the workflow definition. The priority is used as a sort criterion for positioning the work item in the Business Workplace.
  The priority can be changed here.
- Status
  The current processing status is expressed by the work item status [Seite 1373].
- Creation date (created on) and processing date (processed from)
  These are the actual dates and times (when the work item was created and when processing was started).
Using the Messages function, you can display the return codes for processed work items, which were returned to the workflow system after execution of the object method.

Work Item Description
A description of the work item to be executed is provided at the bottom left of this tab page.
The task description is entered in the task definition. It is used for information purposes and generally contains instructions and recommendations on processing the work item displayed.

Attachments
The titles of all the attachments added to this work item or, if the work item is part of a workflow, to the preceding work items are shown in the lower right of this tab page.
Tab Page Activities

Use

The Activities (not yet processed) list contains all the activities that are relevant for processing this work item. The tab page Activities is part of the work item display [Seite 1411].

Features

The work item text of the task represented by the work item is generally at the start of the list ("main activity"). Once this activity has been processed (and completion of processing has been confirmed, if necessary), the status of the work item changes to completed. No other actions can then be carried out in the work item display.

Activities

You can extend the activity list and in this way create a worklist. The activities added represent your "personal worklist" as end user (agent).

<table>
<thead>
<tr>
<th>Function</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Done</strong></td>
<td>An activity selected from the list is reported to be done. This functionality cannot be executed on the leading activity (first line) of this work item. The agent is responsible for reporting that an activity is done. The actual processing is not checked.</td>
</tr>
<tr>
<td><strong>Execute</strong></td>
<td>An activity selected from the list is executed. The main activity (first line) of the work item can also be executed via the menu path Work item → Execute.</td>
</tr>
<tr>
<td><strong>Create</strong></td>
<td>Another activity is added to the list. This activity can entail</td>
</tr>
<tr>
<td></td>
<td>• Executing a method on an existing object (created invoice, created material, etc.)</td>
</tr>
<tr>
<td></td>
<td>• Creating a new object (text, etc.)</td>
</tr>
<tr>
<td></td>
<td>When you create an activity, you are given step-by-step support in dialog by a &quot;wizard&quot;.</td>
</tr>
<tr>
<td><strong>Delete</strong></td>
<td>An activity is deleted from the list.</td>
</tr>
</tbody>
</table>

Scrolling in the activity list: The activity list contains extensive information on each activity, which you can view by scrolling to the right. This information includes the following:

- Description
- Creator of the activity with date and time
- Actual agent of the activity with date and time
- Method and object type
Tab Page Available Objects

Use
All objects that are related to the work item are displayed in the list of available objects. These objects are stored in container elements of the task container as object references. You can display these objects or add new objects to the container element. There are:

- **Ad hoc objects**
  Objects added to a work item in this or one of the preceding steps of the workflow (container element _Adhoc_Objects of the task container).

- **Attachment objects**
  Documents added to a work item in this or one of the preceding steps of the workflow (container element _Attach_Objects of the task container).

- **Process objects**
  - The object currently being processed (container element _WI_Object_ID of the task container)
  - The object added for grouping purposes (container element _WI_Group_ID of the task container)
  - Objects that are referenced in other elements of the task container

The tab page *Available objects* is part of the work item display [Seite 1411].

Features

Displaying objects with their default attributes
Each object referenced in the work item container is displayed with its default attribute [Extern]. The default method [Extern] of each object can be executed upon request. If no default attribute was defined for the object type, the key fields of the object are displayed.

Adding objects
You can extend and process the list of objects. The main purpose of this is to make the relevant information available to the agents of the subsequent steps in the workflow as well. Only object types [Extern] that support the IFFIND interface can be selected. You identify an actual object [Extern] of this type by specifying its key fields [Extern].

Activities
To execute the functions displayed, proceed as follows:

<table>
<thead>
<tr>
<th>Function</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Create</td>
<td>An object is added to the list. When you add an object, you are given step-by-step support in dialog by a &quot;wizard&quot;.</td>
</tr>
<tr>
<td>Display</td>
<td>The default method of an object selected from the list is executed.</td>
</tr>
<tr>
<td>Remove</td>
<td>An object is deleted from the list.</td>
</tr>
</tbody>
</table>
Work Item Display: Technical View

Use
The technical work item display shows all information from the environment of the work item displayed. The technical work item display is intended in particular for workflow system administrators.

The data is always formatted and displayed in a similar way irrespective of the work item type. You should note the work item type shown on the screen.

Features

General

Work item information
- Information derived from the definition of the relevant task: Work item type, work item text, way in which processing is completed.
- Status of the work item as current processing information
- Actual agent (available after dialog work items have been executed).

In the case of processed work items, the Messages function can be used to display the return codes returned to the workflow system after execution of the object method.

Deadline data
Here you can find the current dates/times (creation date/time of work item and date/time processing started and ended) as well as the deadlines that are monitored by the runtime system (requested and latest start and end deadlines).

A monitored deadline shown with red is in the past. If the symbol is displayed as well, a defined action was triggered.

Description
This description displayed here is taken from the task definition. It is used to inform the selected agents and contains instructions and recommendations on processing the work item displayed.

If you have selected a work item using the work item selection or the workflow outbox rather than your workflow inbox, you can execute it provided you are one of the possible agents.

You do not need to be one of the recipients in this case.

Additional functions
As well as the functions available in the standard view of the work item display, the following additional functions are also available:

- Display/create/delete object

Each object referenced in the work item container is displayed with its default attribute. The default method of each object can be executed upon request. If no default attribute is defined for the object type, the key fields of the object are displayed instead.

April 2001
You can extend and process the list of objects. The main purpose of this is to make the relevant information available to the agents of the subsequent steps in the workflow as well.

Only object types [Extern] that support the IFFIND interface can be selected. You identify an actual object [Extern] of this type by specifying its key fields [Extern].

- **Type-specific data**
  Type-specific data only applies to work queue and wait step work items.
  For work queue work items (type A), the objects and tasks contained in the work queue are listed.
  For wait step work items (type E), the number of events expected (information taken from the workflow definition) and the number of events that have already occurred are specified.

- **Execute for testing**
  In order to make it possible to check the runtime system's program execution (work item manager and workflow manager) in debugging mode after execution of an object method, internal communication within the workflow system cannot take place asynchronously. To achieve this, execute the work item for test purposes. Enter /h in the command field first to go to the debugging mode.

- **Monitored work item**
  The work item (type F or W) whose deadlines or completion are monitored by the workflow system is displayed.
  This function is only possible and active from missed deadline work items [Extern]. This function displays the work item. Full display functionality, including all navigation and change options, is also available here for this work item.

- **Instance linkage**
  You go to the relevant line of the instance linkage table, in which the expected event (identified by object type and event ID) and the object (identified by object reference) are specified.
  This function is only possible and active for those work items that wait for an event that completes them. This may apply to dialog work items (type W) and background work items (type B). This does apply to wait step work items (type E).

- **Container**
  This function displays the content of the task container.

- **Change work item [Seite 1438]**

**Activities**
You can go to the technical view of the work item display by choosing Goto → Technical work item display in the work item display or by having this display variant as a presetting in your personal workflow settings [Seite 1395].
Workflow Log: Standard View

Use
The workflow log [Extern] enables you to format all the information that is generated or collected during the execution of one active business process in one central location. The standard view described here is intended, in particular, for agents and process controllers who want to get an overview of the steps processed so far.

Prerequisites
To use the view described here, you must have set a view with ActiveX in your personal workflow settings [Seite 1395].

Integration
Other options for displaying the workflow log are:
- Workflow Log: Technical View [Seite 1422]
- Graphical Workflow Log [Seite 1424]

Features

Application toolbar functions
The following functions are available on the application toolbar:
- Update
- List with technical details
  This function takes you to the technical view of the workflow log.
- Graphical workflow log
- Personal workflow settings
  You can use this function to maintain your personal workflow settings.

Tab pages
The system shows the various views on the workflow log on different tab pages.

Tab page Workflow Chronicle (, what was processed when?)
The tab page Workflow Chronicle shows a hierarchical display of all steps in the workflow, which have been processed so far or are due to be processed. If the workflow has a subworkflow structure, the subworkflows are also displayed.
The Details function (symbol) lists the following information about each step in the lower part of the screen:
- Who carried out what detailed actions for these work items and with what results.
- When these actions were carried out.
- Which objects were involved.

The Agents function (symbol) displays the selected and possible agents of a step.
The Graphic function (symbol) displays the graphical workflow log.
Tab page Workflow Agents (ภาวะผู้ปฏิบัติ, ใครที่เลิกงาน?)
The tab page Workflow Agents shows the employees involved in this workflow up to now. The following is displayed for each employee:

- What action was carried out in what step.
- When this action was carried out.
- Which objects were involved.

This view shows how an employee was involved in the execution of a workflow.

Tab page Workflow Objects (สิ่งที่ถูกจัดรูปแบบ, อะไรที่ถูกจัดฟอร์ม?)
The tab page Workflow Objects lists the objects related to the workflow or addressed up to now in the execution of the workflow. These objects include:

- The “leading” object of the workflow.
- Any attachments and objects added in the individual steps of the workflow.

The following is displayed for each object:

- Who carried out what detailed action for what task.
- When this action was carried out.

This view shows what information was generated and processed, and how.

Information at the click of a mouse
You can view all the information provided in the workflow log using the mouse.
You can also go to the work item display [Seite 1411] for each dialog step. You can display address data for agents as well as the contents of work item attachments or the result of actions that have been executed.

Activities
You can access the workflow log from the work item display or the Business Workplace [Seite 1368] via the icon.
Workflow Log: Technical View

Use
The workflow log [Extern] enables you to format all the information that is generated or collected during the execution of one active business process in one central location.
The view described here, in particular the technical view described below, is intended for workflow system administrators.

Integration
Other options for displaying the workflow log are:
- Workflow Log: Standard View [Seite 1420]
- Graphical Workflow Log [Seite 1424]

Prerequisites
To use the view described here, you must have set the view without ActiveX in your personal workflow settings [Seite 1395].
If you have chosen technical view in your settings, the standard view with technical details is displayed (see below). Otherwise, the two workflow log views are identical.

Features
The system uses a two-level, hierarchical list from the ABAP List Viewer [Extern] to display the various workflow log views. You can adapt the appearance of the list to suit your requirements using display variants.
By single-clicking on an entry or a symbol, you can then branch to the workflow container or the work item display [Seite 1411], for example.
The following views are available:

- **Workflow chronicle**
  The first level of the Workflow chronicle view shows all the workflow steps that have already been processed or are currently pending. If the workflow has a subworkflow structure, the subworkflows are also displayed.
  The second level (detail view) shows the following for each step:
  - Who carried out what detailed actions for these work items and with what results.
  - When these actions were carried out.
  - Which objects were involved.
  This view is used to determine what activities were carried out in a workflow and in what order.

- **Workflow agents**
  The first level of the Workflow agents view shows the employees involved in this workflow up to now.
  The second level (detail view) shows the following for each employee:
  - What action was carried out in what step.
  - When this action was carried out.
• Which objects were involved.
  This view shows how an employee was involved in the execution of a workflow.

• **Workflow objects**
  The first level of the *Workflow objects* view lists the objects that are related to the workflow or that have been addressed up to now during execution of the workflow. These objects include:
  • The “leading” object of the workflow.
  • Attachments and objects added in the individual workflow steps.
  The second level shows the following for each object:
  • Who carried out what detailed action for what task.
  • When this action was carried out.
  This view shows what information was generated and processed, and how.

• ... **With technical details (technical view)**
  The *technical view* shows technical control information for execution of a workflow, as required by workflow administrators, for instance.
  Based on the workflow chronicle, the *technical view* shows technical nodes and control structures, and makes additional data available, such as container elements (>Lorem Ipsum), employee data (Lorem Ipsum), and workflow data (Lorem Ipsum).
  The status of the work item is also displayed.

• ... **With subworkflow structure**
  Here you can choose whether or not to display any subworkflows and their structure.

• ... **With error indicators**
  If you activate the function *View with error indicators*, errors are marked in the log with the symbol. The standard indicator is Lorem Ipsum.

**Activities**
You can access the workflow log from the workflow inbox of the Business Workplace via the symbol, or using the context menu (click the right-hand mouse button).
You can maintain the individual views (such as the technical view) within the workflow log via Views or Views ➔ Other views.
Graphical Workflow Log

Use
The workflow log enables you to format all the information that is generated or collected during the execution of one active business process in one central location.

Integration
Other options for displaying the workflow log are:
- Workflow Log: Technical View [Seite 1422]
- Workflow Log: Standard View [Seite 1420]

Features
The graphical workflow log adds to the textual information. The workflow steps already processed are shown with in a graphical representation of the workflow definition. You can see at a glance which “route” a workflow instance has taken and which activities are processed in parallel to your own within a business process. Unlike the text version of the workflow log, the graphical workflow log also shows the subsequent flow of a workflow instance.

The screen of the graphical workflow log is split into the workflow area (left) and the overview area (right).

The following functions are available:
- **Refresh**
  Refreshes the display.
- **Display node**
  If you select an executed step and choose this function, the system goes to the technical work item display.
  If the step has not yet been executed, the relevant activity is displayed.
- **Align...**
  The workflow is centered within the workflow area.
- **Zoom in**
  The size of the workflow is increased within the workflow area.
- **Zoom out**
  The size of the workflow is decreased within the workflow area.
- **Complete view**
  The whole of the workflow is shown within the workflow area.

For other functions, refer to the documentation on the Workflow Builder [Seite 1002].

Activities
You can call the graphical workflow log from within the workflow log [Seite 1420] via .
Send Mail

Purpose
You want to send a mail to another user in connection with a work item that requires processing. The work item being processed is therefore also to be made accessible to this user for information purposes.

The Send mail function enables you to send mails to any recipients with a text which can be entered freely. These mails are automatically linked to your work item in such a way that when the mail is executed by the recipient the work item is displayed.

Process Flow

Procedure as sender of mail
1. In the Business Workplace's workflow inbox, position the cursor on a work item and choose Other functions → Send mail.
2. If you are already in the work item display, choose Work item → Send mail.
3. Enter the text of the mail.
4. Send the mail to any recipients who are available as mail addressees
   The mail text is added to the work item as an attachment. The work item remains in your Business Workplace with the same status.
5. You can still complete the work item despite the fact that you have sent a mail. Since this makes the mail which has been sent superfluous, the recipient of the mail receives a second mail informing him/her that the (first) mail no longer needs to be dealt with.

Reply ends resubmission
Optional: If you do not want to process the work item until the mail has been replied to, define a resubmission for the work item. To do this, position the cursor on the entry, choose Resubmit and enter a date in the future.
The work item is resubmitted irrespective of this date and appears in your workflow inbox when the reply to the mail is received.

Procedure as recipient of a mail
You receive a mail that can be executed.
1. Read the mail.
2. Execute the mail, if applicable. The work item referred to by the sender in their mail is displayed. For further information on the object to be processed, choose Goto → Object to display the default attribute or execute the default method (generally Display) for the processed object.
   You can execute this work item if you are one of the possible agents of this task.
3. Reply to the mail. To do this, choose Document → Reply or Document → Reply w/reference when the mail is displayed. Then enter your reply and save your entries.
4. Choose Document → Send. On the send screen, the work item to which the mail referred is entered as the "recipient". Do not change this entry. Send the reply to the proposed recipient.
**Result**

The reply sent ends a resubmission which may have been defined for the recipient work item. This work item is appended as an (additional) attachment and can be read as such by the original user and taken into account.
Display Agents

Use
This function displays information about the agents of the work item.

Features
The system displays the following agents for the work item:

- **Recipient [Extern]**
- **Possible agents [Extern]**
- **Excluded agents [Extern]**

You can choose a compressed display - containing only the user names (*users only* function) - or a complete display with additional information on the relationships used by the system to determine the agents (*overall view* function). You can also display the organizational assignment of each user.

Note that this information is only available for work items of types for which an agent is logical and necessary.

Activities
You can display the agents from the *work item display* by choosing *Goto → Agents → ...*
Changing Deadlines

Use
You use this function if you want to change the deadlines for a work item at workflow runtime. This dialog box displays all the deadlines of work item processing.
Please also refer to Current Dates/Times of a Work Item [Extern].

Features
Depending on whether the work item represents a step in the workflow or a single-step task, the deadlines were either specified when this step was defined in the workflow definition or when the single-step task was started in dialog.
The dialog box is split into the following sections:

Deadlines
- Start by (latest start)
- End by (latest end)

Planned deadlines
- Start by (requested start)
- End by (requested end)

A monitored deadline shown in blue is in the past. If it is also displayed, the appropriate action has been initiated. This generally involves informing the deadline recipient.

Actual dates/times
- Created on: Creation date/time of the work item
  (Technically: The work item is created with the status ready or - if the requested start date has not yet been reached - with the status waiting).
- Processed from: Start of processing
  (Technically: Transition of the work item to the status in process).
- Completed on: End of processing or date when set to ‘done’).
  (Technically: Transition of the work item to the status completed.)

Activities
To execute this function, choose the function in the work item display [Seite 1411].
Attachment Management

Use

One or more attachments can be assigned to each work item that appears in the Business Workplace's workflow inbox.

Attachments are documents written either with a SAPscript editor (document classes RAW, SCR) or with a PC application (document classes DOC, URL, PPT, XLS, PDF, ...) and then imported. You can enter new documents as attachments or create attachments from existing files.

Features

General

The attachment is automatically
- Added to the work item container
- Added to the container of the superordinate workflow
- Added to the containers of the subsequent work items in the workflow

You can define default documents for the individual document classes. For further information, refer to Default Documents [Extern].

Attachments can be displayed by the recipients of the subsequent steps. But they cannot be changed and, therefore, have a document character.

A superior who is to make a decision on releasing a budget can enter an attachment justifying their decision. The selected agents of the subsequent steps can display this attachment.

If a work item has attachments, this is indicated by a symbol in the Attachments column in the Business Workplace. You can also execute the function for processing an attachment by double-clicking in this column (column header AT). If an attachment already exists, it is displayed.

If a work item has attachments, this is indicated by the symbol in the work item display.

Functions on attachments

<table>
<thead>
<tr>
<th>Display</th>
<th>Displays the attachment selected in the dialog box Existing attachments.</th>
</tr>
</thead>
<tbody>
<tr>
<td>Create</td>
<td>Creates a new attachment. If you want to import an existing document from your PC to the attachment, choose Import.</td>
</tr>
<tr>
<td>Change</td>
<td>Displays the selection of Changeable attachments.</td>
</tr>
<tr>
<td>Delete</td>
<td>Displays the selection of Deletable attachments.</td>
</tr>
</tbody>
</table>

Activities

You can access attachment maintenance in the Business Workplace by choosing or the relevant context menu (right-hand mouse button).
Reject Execution

Use
You can use this function if you need to reject execution of a work item for business or technical reasons. This function is only available for work items of type W.

The table entry to be processed already exists or the material whose master data is to be changed is no longer used.

Prerequisites
This function is only available if the property processing rejectable has been selected for the related activity in the workflow definition.

Features
Processing of the work item is terminated with the reject execution function. The subsequent steps defined in the workflow definition are executed.
Do not use this function if you do not want to or cannot process the work item for personal reasons (not responsible, not competent). In this case, replace the reserved work item or forward it.

Activities
You can access this function in the Business Workplace [Seite 1368] by choosing Other functions or the work item display (Work item → Other functions → Reject processing).
Execute Together

Use
You can select various work items and then execute them together. This function is only available for work items of type W.

The work items do not necessarily have to belong to the same workflow.

Prerequisites
The work items must refer to the same single-step task.
For information about other prerequisites for this function, in particular with regard to the underlying method, refer to Creating an "Execute Together" Method [Extern].

Features
Only the first of the selected work items is proposed for execution. The entries you make here also apply to the other work items to be executed together.

Activities
Select the work items that you want to execute together. Then choose Execute together via Other functions... in the Business Workplace [Seite 1368].
Grouping According to Sort Key

Use

Each work item carries the two container elements _WI_Object_ID and _WI_Group_ID in its container. Both elements have been defined to hold an object reference.

- The container element _WI_Object_ID automatically contains the reference to the object to be processed in the work item.
- The container element _WI_Group_ID can contain an object reference, which must be assigned to this container element in a binding or via initial value assignment.

The object reference assigned via _WI_Group_ID is generally not identical to _WI_Object_ID, but is derived from the work item execution environment. It is used to group work items that refer to different objects or object types but are nevertheless connected.

Features

Display default attributes in workflow inbox

The default attributes [Extern] of the objects referenced in _WI_Object_ID and _WI_Group_ID are available in the Business Workplace's workflow inbox [Seite 1408] under the column headers Object or Group for grouping, sorting and filtering purposes.

Material master data is processed in several instances of a workflow. The container element _WI_Object_ID in the containers of the individual work items therefore always contains the reference to the object of type BUS1001 (material) to be processed.

The container element _WI_Group_ID is assigned the expression &Material.Labor& (laboratory/drawing office for material as object reference) in the relevant steps of the workflow definition. The default attribute of the laboratory is therefore available as a group.

Default methods

The default method [Extern] of the objects referenced in _WI_Object_ID and _WI_Group_ID is executed by double-clicking in the Object or Group column.
Change Priority

Use
The priority of a work item is a measure of its urgency. It can be used as a sort criterion for organizing the workflow inbox. This function is only available for dialog work items.

Features
End users can display and change the priority of a work item. The priority is between 1 (highest) and 9 (lowest). The change in priority can also be passed on automatically to the superordinate workflow and then to all work items created subsequently. The Pass on priority to subsequent steps indicator must be selected for this to take effect.

If a higher priority (lower number) is defined in the workflow definition for one of the subsequent steps, it is not changed.

Activities
You can access the Change priority function in the Business Workplace by choosing Other functions...
Create Link

Use
By storing links to work items in folders [Extern], you can organize your work effectively using a personal folder hierarchy.

Features
The work item is added to a personal or shared folder as a link. This work item can be displayed from this folder and executed by its recipients. Work items can have an unlimited number of links.

Activities
You can access this function in the Business Workplace [Seite 1368] by choosing Environment...
Forward Work Item

Use
Forwarding passes a work item on to another user for execution.

Integration
The range of users to whom a work item can be forwarded is determined from the task definition as follows:

<table>
<thead>
<tr>
<th>Task definition:</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Task is a general task [Extern]</td>
<td>Work item may be forwarded to all users.</td>
</tr>
<tr>
<td>The indicator General forwarding allowed is set for the task</td>
<td>Work item may be forwarded to all users.</td>
</tr>
<tr>
<td>The indicator General forwarding not allowed is set for the task</td>
<td>Work item may only be forwarded to the recipients [Extern] of the task.</td>
</tr>
</tbody>
</table>

Features
The user has the following search options to select the new recipient:

- The F4 input help can be used to determine the new recipient by performing a generic search for name components.
- The structure search can be used to determine the new recipient using the graphical display of the organizational plan.

The new recipient does not have to be an actual user. (The work item can also be forwarded to an organizational unit or a job.)

Activities
You can access this function in the Business Workplace [Seite 1368] by choosing 📞.
Change Work Item

Use
The function enables you to change information and data associated with the selected work item.

Features

Changes in the initial screen
In general, the changes that are allowed depend on the type and status of the work item. The following changes are possible:

- **Work item text**
- **Priority**
- **Deadline data**
  
  You can set and change deadline data for the start and end deadlines of the work item. You can only make specifications for deadlines that are still being monitored and are in the future. For example, dialog work items with the *ready* status can no longer be assigned a requested start.

Manual intervention into the processing of work items
If you are an administrator with the appropriate authorization, you can intervene manually into the processing and therefore into the process flow of a workflow. This is particularly helpful for dealing with errors. The following changes are possible:

- **Set to ‘ready’ manually**
  The work item status is changed from *waiting* to *ready*.
  The work item therefore appears in the workflow inboxes of the selected agents.

- **Complete manually**
  The work item status is changed to *completed*.
  In the case of dialog work items, this status change (together with the current work item container) is passed on to the workflow system for evaluation. If the object method to be executed is a synchronous method with a result, the possible result values are displayed for selection.

- **Replace manually**
  The status of the work item is reset to *ready*.
  The work item is therefore displayed again in the workflow inboxes of all selected agents.

- **Restart after error**
  Work items and, in particular, workflows with errors can be restarted after the errors have been corrected.
  The information written by the workflow system in a log when the error occurred is now taken into account.

- **Delete logically**
  The status of the work item is changed to *logically deleted*. 
In the case of dialog work items, this status change is passed on to the workflow system.

**Locking/unlocking execution**
An administrator can lock or unlock the current work item for execution. To do this, you select the appropriate function from the *Edit* menu.

**Deleting deadlines**
To delete a deadline you have changed or entered and reset it to its initial values, choose *Edit* → *Delete deadline*.

**Changing work item containers**
Choose *Edit* → *Change container* to go to the editor for changing the work item container. The contents of the container for the relevant work item are displayed. You can change the current, runtime-specific data for the particular work item. The container may still have elements that do not currently have a value. These elements are hidden as standard. The presence of elements that are not displayed is shown by an indicator. To display these elements, choose *Edit* → *Show elements*. To add more lines to container elements defined as multiline, choose *Edit* → *Additional line*.

**Activities**
You can access this function in the Business Workplace [Seite 1368] by choosing Other functions, the relevant context menu or the technical work item display [Seite 1418].
Set Work Item to Done

Use
This function is used by the recipient of a work item to confirm explicitly that processing of this work item has been completed. As long as explicit confirmation has not been provided, the work item has executed status and can be executed again or forwarded.

The status of the work item changes from executed → completed.

Prerequisites
This function is only possible on a work item if a setting was made in the definition of the associated single-step task stating that the end of processing must be confirmed explicitly.

Activities
You use this Business Workplace function if the status of the work item has been changed to executed by a terminating event.
In general, however, the work item is not set to done via the Business Workplace. This function is usually provided as a dialog box directly after a work item has been processed.
Missed deadline work items [Extern] must always be confirmed explicitly. After execution, they remain in the status in process until they are set to set to done.
You can access this function in the Business Workplace [Seite 1368] by choosing Other functions.
Workflow Outbox

Use
The workflow outbox lists the workflows started and the work items forwarded and executed by the current user.

Integration
As is the case for the workflow resubmissions [Seite 1443] and the workflow inbox [Seite 1408], the workflow outbox is an integral part of the Business Workplace.

Features
Views in the workflow outbox

Started workflows
This view shows work items for the tasks started by you as a user in dialog or by a triggering event whose event container contains your user name as _Evt_Creator.

Work items executed by me
The work items executed by you are displayed in this view.

Forwarded work items
The work items forwarded by you are displayed in this view.

Workflow Functions
The screen area in which the worklist is displayed has a toolbar containing the workflow functions and functions for configuring the list. For information about these functions, refer to the documentation on the ABAP List Viewer [Extern].

The following functions are available for the different workflow outbox views:

- **Update**
  You use this function to update the workflow outbox.

- **Display work item** (not in Started workflows view)
  This function goes directly to the work item display [Seite 1411]

- **Change agent**
  You use this function to perform an ad-hoc agent assignment [Seite 1097]. [Seite 1411]

- **Display workflow log**
  This function displays the workflow log [Seite 1420].

- **Display attachments** (refer to attachment management [Seite 1429])

- **Environment**
  - **Display objects in workflow** (dialog work items only)
  - **Display workflow relationships** [Seite 1436]
  - **Start Workflow** [Seite 1450]
• Create link [Seite 1434]

• Change selection period
  • Today only
  • Last 7 days
  • Last 30 days
  • User-defined ...

The functions described above can be called using the relevant context menu.

**Activities**

You can access the workflow outbox in the overview tree in the Business Workplace [Seite 1368] via Workplace → Outbox.
Workflow Resubmissions

Use
The workflow resubmissions [Extern] function can be used to resubmit work items for processing at a later point in time or periodically and to display these work items in a list. Technically, the following applies to work items which appear in your workflow resubmission folder: the work item status is set to waiting, your name is entered as the actual agent, and the requested start date is set to the resubmission date.

Integration
As is the case for the workflow inbox [Seite 1408] and the workflow outbox [Seite 1441], the workflow resubmissions is an integral part of the Business Workplace.

Features

Workflow Functions
The screen area in which the worklist is displayed has a toolbar containing the workflow functions and functions for configuring the list. For information about these functions, refer to the documentation on the ABAP List Viewer [Extern].
The following functions are available in the workflow resubmissions:

- **Update**
  You use this function to update the workflow resubmissions.

- **Display work item**
  This function goes directly to the work item display [Seite 1411]

- **Display workflow log**
  This function displays the workflow log [Seite 1420].

- **Display attachments**
  This function is only available if the work item has attachments. Refer to attachment management [Seite 1429].

- **End resubmission**
  The work item is put back into the workflow inbox. It has the status reserved.

- **Environment**
  - Display objects in workflow (dialog work items only)
  - Display workflow relationships [Seite 1436]
  - Start Workflow [Seite 1450]
  - Create link [Seite 1434]

The functions described above can be called using the relevant context menu.

Activities
You can access the workflow resubmissions in the overview tree in the Business Workplace [Seite 1368] via Workplace → Resubmissions.
You are asked for a date for resubmission of the work item. The work item then disappears automatically from the workflow inbox and is not displayed there again until the date specified.
Work Item Preview

Use
The work item preview is in the bottom right corner of the Business Workplace screen and provides a preview of the properties of a work item. All the information on a work item is displayed if you choose the function Display work item in the worklist of the workflow inbox [Seite 1408] or the workflow resubmissions [Seite 1443]. This function goes directly to the work item display [Seite 1411]. A concise view of the workflow log [Seite 1420] is offered in the workflow outbox [Seite 1441] when the started workflows are displayed.

Integration
The work item preview is part of the Business Workplace [Seite 1368].

Features

Workflow functions
The description of the work item is displayed in the work item preview. It is also possible to access objects and attachments of the work item directly from the preview. The concise view of the workflow log enables you to access processed steps (individual work items), their agents and information objects addressed so far. For more information about the concise view of the workflow log, please refer to Workflow Relationships [Seite 1436].

Tips & tricks
"Tips & tricks" are displayed when you position the mouse pointer on the title of a "tips & tricks" entry. Whenever the work item preview is updated, a new title is offered. You can activate/deactivate "tips & tricks" in your personal workflow settings [Seite 1395] (Settings → Workflow settings → Personal settings).

User exit
The work item preview can be adapted to customer-specific requirements using a user exit [Extern].

Activities
The work item preview is activated/updated whenever you select a work item in your worklist.
Transport and Client Copy

Process Flow

Transport of cross-client tasks
As cross-client transport objects, standard tasks [Extern] (task type TS) and workflow templates [Extern] (task type WS) are always connected to the transport system and are automatically included in a transport request, if the client settings (table T000) allow changes to cross-client objects.

Transport of client-specific tasks
Customer tasks [Extern] (task type T) and workflow tasks [Extern] (task type WF) are only included automatically in a transport request in a particular client if Automatic recording of changes is set for this client in table T000.
In other clients, you can include customer tasks and workflow tasks in a transport request manually. To do this, choose Task → Transport or Workflow task → Transport in the relevant definition environment.

Transport of settings for tasks
You maintain the following settings for tasks on a client-specific basis:
- Assignments between tasks and their possible agents
- Event receiver linkages and their activation
- Workflow extension
The system automatically includes these settings in a transport request if Automatic recording of changes is set for the relevant client in table T000.

Client copy
With a client copy [Extern], the assignments between tasks and their possible agents and the event receiver linkages are copied.

The event receiver linkages are always deactivated in the target client after copying.
If you want to copy the activation indicator of each individual event receiver linkage as well, you must specify this explicitly as a parameter option for copying tables of class A.

Event linkages with installation or upgrade
With a new installation or an upgrade, the event receiver linkages are copied into all target clients. But the activation entries are only copied into client 000. Any existing entries are overwritten in this operation.
Workflow Agent

Definition
Person within the system of workflow roles [Seite 94] who appears as an end user in productive workflows. The workflow agent starts workflows and processes work items. Their work area is the Business Workplace.

Use
This part of the documentation is to be used in conjunction with the user menu for the workflow agent. This user can access this user menu on the screen SAP Easy Access via . Assign the role SAP_BC_SRV_USER to the user.

Structure
The documentation for this role consists of:
- Working with the Business Workplace [Seite 806]
  Here you will find vital documentation for workflow agents.

Integration
This role is part of the role-based procedure model containing the following other roles:
Working with the Business Workplace

Purpose
Working with the Business Workplace does not only concern workflow agents, but all other roles as well.

Process Flow

When working with *SAP Business Workflow*, workflow agents use only the Business Workplace. From the point of view of the workflow agent, starting workflows and executing work items means creating or approving a notification of absence, for example. All the technical aspects are irrelevant for agents. In the event of problems of a technical nature, the *workflow system administrator* [Seite 683] should be contacted. Any questions about the business process should be addressed to the *process controller* [Seite 884].
Business Workplace: Workflow Functions

Use
You use this part of the Business Workplace [Extern] if you want to use the functions of SAP Business Workflow. The Business Workplace is the main interface between an end user and the workflow system. All dialog and missed deadline work items to which the user is assigned as a recipient are displayed in the user’s workflow inbox.

As a head of department, you are responsible for approving leave requests. The relevant approval process is implemented using a workflow in your enterprise.

The requests (in the form of work items) appear in your worklist (workflow inbox) and must be rejected or approved there.

The rejected or approved requests (executed work items) are not only returned to the applicants after processing, but are also put into your workflow outbox (under Work items executed by me). You can therefore check the requests you have processed.

Features

Workflow settings
You can configure the workflow functions in the Business Workplace using the personal workflow settings [Seite 1394].

Business Workplace screen areas
The Business Workplace has three screen areas, which are used in the following manner by SAP Business Workflow:
Business Workplace: Workflow Functions

Overview tree
The following workflow functions are available under the Inbox node, which is under the initial node Workplace:

- Workflow
  - Grouped according to task
  - Grouped according to content
  - Grouped according to content type
  - Grouped according to sort key
- Overdue entries
- Deadline messages
- Incorrect entries

For information on these functions, refer to Workflow Inbox [Seite 1408].

The following functions are available under the Outbox node, which is under the initial node Workplace:

- Started workflows
- Work items executed by me
- Forwarded work items

For information on these functions, refer to Workflow Outbox [Seite 1441].

The Resubmissions node is located under the initial node Workplace and contains the:

- Workflow resubmissions [Seite 1443]
**Business Workplace: Workflow Functions**

**Worklist**
The worklist [Extern] is displayed in the upper right corner of the Business Workplace screen. Depending on whether you are in the workflow inbox, the workflow outbox or the workflow resubmissions, you have various functions available to you, which are described at the respective locations.

**Work item preview**
In the lower right corner of the Business Workplace screen, a work item selected in the worklist is displayed in a preview [Seite 1445]. Not all the functions of the work item display or the workflow log are available.

A user exit [Extern] can be used to configure the work item preview to suit your individual requirements.

**Support for context menus**
All workflow functions can be called using the relevant context menu.

**Workflow Toolbox**
*SAP Business Workflow's Workflow Toolbox [Seite 1446]* enables the user to access workflow functions even during a workflow-driven application transaction.

**E-mail notification for new work items**
The report RSWUWFML can be used to inform an employee by mail that there is a new work item in their Business Workplace inbox. This function is therefore beneficial to all employees who do not work with their Business Workplace on a daily basis.

**Activities**
To access the Business Workplace from the *SAP Easy Access* screen, choose one of the following options:

- **Menu → Business Workplace**
- **Tools → Business Workflow → Development → Runtime Tools → Business Workplace.**
# Work Item

## Definition

Object that represents a task or action in the workflow system at runtime.

## Use

Work items are subdivided into a specific work item type according to their assignments. The internal processing procedures are controlled via this work item type. The work item type determines which statuses and transitions are valid. Depending on the work item type, some of these work items are displayed in a user's work list. Other work items, on the other hand, are only used and processed internally.

## Structure

### Work item types displayed in the Business Workplace

<table>
<thead>
<tr>
<th>Work item type</th>
<th>Short text</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>W</td>
<td>Work item with dialog</td>
<td>Runtime representation of single-step tasks that require interaction with the user. Runtime representation of user decisions</td>
</tr>
<tr>
<td>D</td>
<td>Missed deadline</td>
<td>Work item for notification of missed deadline</td>
</tr>
<tr>
<td>A</td>
<td>Work Queue</td>
<td>A work queue is a list of objects to be processed once and together in a limited time frame.</td>
</tr>
</tbody>
</table>

### Other work item types

<table>
<thead>
<tr>
<th>Work item type</th>
<th>Short text</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>F</td>
<td>Workflow</td>
<td>Runtime representation of a multistep task</td>
</tr>
<tr>
<td>B</td>
<td>Batch item</td>
<td>Runtime presentation of a single-step task that runs in the background</td>
</tr>
<tr>
<td>E</td>
<td>Wait step work item</td>
<td>Runtime representation of a wait step in the workflow definition</td>
</tr>
<tr>
<td>C</td>
<td>Container anchor</td>
<td>This type of work item is required as a special development in the EDI environment. It does not normally appear in the workflow environment. Work items of this type should be regularly deleted or archived.</td>
</tr>
</tbody>
</table>
Dialog Work Items (Type W)

Definition
Work item that represents a task at runtime that requires interaction with the user.

Since the user decision is also represented internally by a task, a dialog work item can also represent a user decision.

When a dialog work item is executed, the underlying object method of the task is called. The deadlines for executing dialog work items are monitored.

Use
A dialog work item is displayed with ready status in the workflow inbox of the Business Workplace. It is removed from the integrated inboxes of the other agents when the recipient reserves, executes, or processes this work item with other functions.

The database oriented approach used in SAP Business Workflow allows a work item to be seen by several recipients equally authorized in organizational terms in their inboxes and executed from there. However, only one recipient can actually reserve this work item for processing and execute it. The work item is then no longer available to any other recipients.

Integration
A task represented by a dialog work item can be

- a step in a workflow definition:
  In the workflow definition, reference is made to tasks in the activity and user decision steps.

- started as single steps via an event or in dialog:
  Tasks can be started as elementary activities directly in dialog or via a triggering event. These tasks are then also represented by a dialog work item in the workflow inbox.
# Status of a Dialog Work Item

The valid statuses for dialog work items (type W) are listed in the following table.

<table>
<thead>
<tr>
<th>Work item status</th>
<th>Description</th>
</tr>
</thead>
</table>
| Waiting          | The work item has been scheduled for its requested start.  
A work item has this status  
- if it already exists but the requested start specified in the workflow definition has not been reached yet.  
- if it has been set to resubmission  
Work items in the waiting status are not displayed in the workflow inbox. |
| Ready            | The work item has been released for execution and appears in the workflow inbox of all recipients. |
| Reserved         | The work item has been received by one of its recipients with the result that its status has changed from ready to reserved.  
A work item in the reserved status is then displayed to this recipient only. It is no longer displayed in the workflow inboxes of the other recipients. |
| In process       | The work item is currently being processed by a recipient or in a different mode.  
A work item also has this status  
- if the work item is waiting for its terminating event.  
- if the user cancelled the method.  
- if the method was ended with a temporary exception for which no subsequent steps have been modeled.  
The point at which processing is completed cannot be detected by the workflow system in this status. As long as the status of the work item is set to in process, database changes have not been made. |
| Executed         | The work item is awaiting explicit confirmation of its completion.  
The work item only has this status if it is necessary to confirm that it has been completed. A work item with executed status can be executed or forwarded several times until it is set to the status done in the Business Workplace.  
In this way, groupware components are realized in SAP Business Workflow. |
| Completed        | The execution of the work item is completed.  
The result of the task represented by the work item is correct, i.e. the result modeled in the workflow definition.  
Work items in the completed status are not displayed in the workflow inbox of the Business Workplace. |
| Logically deleted| Execution of the work item is no longer meaningful or required by the workflow logic.  
A work item changes to the logically deleted status in the following way:  
- Termination in parallel processing branches  
  When the required number of processing paths has been executed in a fork, the work items in the other paths that have not yet reached the completed status are automatically set to the logically deleted status.  
- Intervention by an administrator  
  An administrator can only set a work item to the logically deleted status if it has not yet reached the completed status and is not part of a higher-level workflow. |
Status of a Dialog Work Item

<table>
<thead>
<tr>
<th>Status</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>logically deleted</td>
<td>Work items in the <em>logically deleted</em> status are not displayed in the workflow inbox of the Business Workplace.</td>
</tr>
<tr>
<td></td>
<td>A work item with the <em>logically deleted</em> status may have caused database changes or other actions (generate event, send notification). These changes are not canceled automatically (compensated).</td>
</tr>
<tr>
<td>error</td>
<td>Execution of the work item was terminated with an error.</td>
</tr>
</tbody>
</table>

In addition to the statuses given above, a work item can be **locked against execution**. This is possible in any status and involves an administrative function which is selected for a work item via the *Change* option.
Status Transitions of a Dialog Work Item

The diagram below shows the possible status transitions that a dialog work item (type W) can undergo:

The arrows are labeled with the functions used by a user to trigger the respective status transition.

**Comments**

For further information, refer to Status of a Dialog Work Item [Seite 1373].

**Transition from status waiting**

The transition from the status waiting to the status ready is performed automatically by the work item manager when the requested start is reached. A workflow system administrator can set a work item to the status ready manually.

**Transition from status ready**

From this status, the work item passes either to the status reserved or via the status in process to the status completed.

**Transition from status reserved**

A work item with the status reserved can be reset to the status ready.

**Transition to and from status in process**

A work item with the status in process can be reset to the status ready. A workflow system administrator can reset a work item manually. This function is available when changing the work item.

**Transition from status executed**

After confirmation of end of processing, the work item assumes the status completed.
Status Transitions of a Dialog Work Item

Transition from status completed
Work items with the status completed can no longer be set to another status even if a workflow system administrator intervenes.

Transition from status error
A workflow system administrator can intervene and set work items with errors to the status in process or the status logically deleted (possibly after eliminating the error).

Transition from status logically deleted
Work items with the status logically deleted can no longer be set to another status even if a workflow system administrator intervenes.
Missed Deadline Work Item (Type D)

Definition
Notification of a deadline recipient if the runtime system detects that the deadline for a certain work item has been exceeded.

Use
This work item informs its recipients that a deadline (start or end deadline) of a monitored work item has been exceeded. The recipients are informed by means of a missed deadline work item (type D) in the workflow inbox of the Business Workplace.
When it is executed, this work item displays information on the monitored (and now late) work item. The text for notifying the recipient is set by default.
When a deadline is monitored for an activity [Seite 1024] or user decision [Seite 1074], it is also possible to enter an individual text in the respective task definition.
Status of a Missed Deadline Work Item

The valid statuses for missed deadline work items (type D) are listed in the following table:

<table>
<thead>
<tr>
<th>Work item status</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>📚 Ready</td>
<td>The work item appears in the Business Workplace of the recipients.</td>
</tr>
<tr>
<td>🕹️ In process</td>
<td>One of the recipients has executed the work item</td>
</tr>
<tr>
<td>📜 Completed</td>
<td>The execution of the work item is completed. Work items in the <em>completed</em> status are not displayed in the workflow inbox of the Business Workplace.</td>
</tr>
</tbody>
</table>
Status Transitions of a Missed Deadline Work Item

The following status transitions are possible for missed deadline work items (type D):

**Transition to and from status in process**

The status of a missed deadline work item changes from *ready* to *in process* when it is executed by a user. Execution of a missed deadline work item displays the most important information on the monitored work item.

The missed deadline work item remains in the *in process* status until end of processing is confirmed explicitly. Until then, the work item can be executed or forwarded several times.

**Transition to completed status**

The status of the missed deadline work item changes to the *completed* status when completion of processing has been confirmed explicitly.
**Workflow Work Item (Type F)**

**Definition**

Work item that represents a multistep task at runtime.

**Use**

For every multistep task started there is one type F work item. The workflow log and the workflow container can be accessed:

- For error diagnosis and error correction if no work items are displayed
- For information on steps of a workflow (including their current agents, notes, and ad hoc objects) already processed
- For modifying an ongoing workflow by changing the workflow container
- For reporting on completed processes

Type F work items are not displayed in the Business Workplace but can be found using the work item selection [Seite 1490].

**Structure**

A workflow consists of a sequence of work items that are executed by agents or the system. The work items represent the steps in the workflow definition that refer to a particular task. These are steps of the types activity [Seite 1024] or user decision [Seite 1074].

**Integration**

The Work Item Manager manages the processing of work items and monitors deadlines. To automate workflow processes, activities in the workflow can also refer to object methods which run in the
background. If this is the case, the work item manager initiates the calling of the background processes. The work items whose execution requires dialog can be accessed by the selected agents (determined from the organizational model and role resolution) from their worklists in order to select them for processing. This worklist is displayed and managed in the workflow inbox of the Business Workplace [Seite 1368].
Status of a Workflow Work Item

The valid statuses for workflow work items (type F) are listed in the following table:

<table>
<thead>
<tr>
<th>Work item status</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Waiting</td>
<td>The (sub) workflow is scheduled but its requested start has not yet been reached.</td>
</tr>
<tr>
<td>Ready</td>
<td>Temporary interim status</td>
</tr>
<tr>
<td>In process</td>
<td>Execution of the (sub) workflow has begun.</td>
</tr>
<tr>
<td>Completed</td>
<td>The end of the (sub) workflow has been reached.</td>
</tr>
<tr>
<td>Logically deleted</td>
<td>Execution of the (sub) workflow is no longer required or meaningful. A status of a workflow changes to logically deleted in the following way:</td>
</tr>
<tr>
<td></td>
<td>• Intervention by an administrator.</td>
</tr>
<tr>
<td></td>
<td>An administrator can only set a workflow item to the logically deleted status if it has not yet reached the completed status.</td>
</tr>
<tr>
<td></td>
<td>• Termination in parallel processing branches</td>
</tr>
<tr>
<td></td>
<td>When the required number of processing paths has been executed in a fork, the work items in the other paths that have not yet reached the completed status are set to the logically deleted status.</td>
</tr>
<tr>
<td></td>
<td>A workflow set to the logically deleted status is recursively scanned for dialog and workflow work items (type W or F) that do not yet have the status &quot;completed&quot;. These work items are then also set to the status logically deleted.</td>
</tr>
<tr>
<td></td>
<td>A work item changed from the executed status to the logically deleted status may already have caused database changes or other actions (generate event, send notification). These changes are not canceled automatically (compensated), but are recorded in the log.</td>
</tr>
<tr>
<td>Errors</td>
<td>An error occurred in workflow control.</td>
</tr>
</tbody>
</table>

Errors

Workflows and their statuses are not displayed in the Business Workplace. Information about the workflow can be accessed via the subordinate dialog work items.
Status Transitions of a Workflow Work Item

The diagram shows the possible status transitions.

### Status and Status Transitions of Work Items (F)

- **waiting**
- **ready**
- **incorrect:**
- **in process**
- **completed**

The statuses of a workflow work item and the permissible status transitions always concern an entire workflow.

A step (activity) in the workflow definition can reference both a task and another workflow. A workflow can therefore also contain subordinate subworkflow items.

### Comments

**Transition from status waiting**

The work item manager automatically carries out the transition from the *waiting* status to the *ready* status when the requested start date/time of the workflow has been reached.

A workflow system administrator can set a work item to the status *ready* manually.

**Transition from status ready**

The *ready* status is a temporary interim status because it only exists until the first work item of the workflow has been created.

**Transition to and from status in process**

The status of the workflow changes to *in process* as soon as the first work item of this workflow has been created. The workflow remains in this status until the entire workflow definition has been processed.

**Transition to incorrect status**

An error occurs during workflow control or coordination.

Role resolution for determining an agent does not return a result which can be used.
A workflow system administrator can intervene and set workflows with errors to the status *in process* or the status *logically deleted* (possibly after eliminating the error). If a workflow is incorrect, the responsible workflow system administrator specified either globally in Customizing or in the basic data of each workflow definition is notified by mail.

This status does *not* mean that a dialog work item of this workflow has the *incorrect* status.

**Transition to completed status**

A workflow is set to the status *completed* when the last step of the relevant workflow has been completed.
Background Work Item (Type B)

**Definition**

Work item that represents a single-step task at runtime whose execution does not require a dialog and, therefore, can be controlled automatically by the system.

**Integration**

Type B work items are not displayed in the Business Workplace.
### Status of a Background Work Item

The valid statuses for background work items (type B) are listed in the following table.

<table>
<thead>
<tr>
<th>Work item status</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Waiting</strong></td>
<td>The work item has been scheduled for its <em>requested start</em>. A work item has this status after it has been created until the <em>requested start</em> specified in the workflow definition is reached.</td>
</tr>
<tr>
<td><strong>Ready</strong></td>
<td>Temporary intermediate status of a background work item. The system calls the associated object method as soon as a background work item can be processed. The status of the background work item then changes to <em>in process</em> immediately.</td>
</tr>
</tbody>
</table>
| **In process**       | The work item is currently being processed. A work item also has this status:  
  - if the method was left with a temporary exception. In this case, special [Error Handling for Background Work Items with Temporary Errors](Seite 1388) is carried out.  
  - if the work item is waiting for its terminating event. The point at which processing is completed cannot be detected by the workflow system in this status. |
| **Completed**        | The execution of the work item is completed. The result of the task represented by the work item is correct, i.e. the result modeled in the workflow definition. |
| **Logically deleted**| Execution of the work item with regard to the process logic is no longer meaningful or necessary for the process to continue. A work item changes to the *logically deleted* status in the following way:  
  - Termination in parallel processing branches  
    When the required number of processing paths has been executed in a fork, the work items in the other paths that have not yet reached the *completed* status are automatically set to the *logically deleted* status.  
  - Intervention by an administrator  
    The administrator can only set a work item to the *logically deleted* status if it has not yet reached the *completed* status and is not part of a higher-level workflow.  
    A work item with the *logically deleted* status may have caused database changes or other actions (generate event, send notification). These changes are not canceled automatically (compensated). |
| **error**            | Execution of the work item was terminated with an error. |

In addition to the statuses given above, a work item can be **locked against execution**. This is possible in any status and involves an administrative function which is selected for a work item via the *Change* option.
Error Handling for Background Work Items

Use
Error handling for background work items must be carried out by the workflow system

- because background work items (with errors) are not displayed in the Business Workplace and are therefore detected too late, if at all.
- because background work items that are temporarily incorrect, for which another execution attempt can (theoretically) be successful, cannot be restarted by a user.

(Temporarily incorrect work items are work items whose method was terminated with a temporary exception for which no subsequent step has been defined.)

Features
In Customizing for the workflow system, you can define how often the workflow system attempts to restart a temporarily incorrect work item. You can also define the interval between two repetition attempts and activate the monitoring function.

Monitoring is also activated as part of automatic Customizing [Extern] (Tools → Business Workflow → Development → Utilities → Customizing).

You can also specify the maximum number of repetition attempts (“repetition counter”) separately for each individual background step in the respective workflow definition. This setting overrides the value in Customizing if it is greater than zero.

If an error (method exception) occurs when background work items are executed by the workflow system, the system responds as follows depending on the type of exception and error modeling settings:

<table>
<thead>
<tr>
<th>In the workflow definition...</th>
<th>The exception is defined for the object method as a...</th>
<th>System or application error</th>
</tr>
</thead>
<tbody>
<tr>
<td>a subsequent step is modeled for the exception.</td>
<td>The step has status completed and the modeled subsequent step is executed. Background work items for which a repetition counter is explicitly modeled in the workflow definition are restarted first by the system. If all of the attempts are unsuccessful, the status of the work item changes to completed and the modeled subsequent step is executed.</td>
<td>The step has status completed and the modeled subsequent step is executed.</td>
</tr>
</tbody>
</table>
Error Handling for Background Work Items

| no subsequent step is modeled for the exception | The step is not yet completed. The respective work item retains the status in process. Background work items are restarted by the system. The number of repetition attempts is determined either by the repetition counter in the step definition or - if this is equal to zero - by the repetition counter set in Customizing. If all of the attempts are unsuccessful, the work item status changes to incorrect. | Workflow and work item assume the error status. |

**Processing Incorrect Work Items**

The workflow system sends a mail to the relevant system administrator for every background work item with the status incorrect.

**Processing Work Items That Have Been Started**

The workflow system determines all of the background work items that have been in process for longer than 30 minutes. An error message is then sent to the workflow system administrator for all of these background work items, since the system assumes that processing has been cancelled. However, this does not necessarily mean that an error has occurred.
Wait Step Work Item (Type E)

Definition
Work item that represents a wait step or a workflow at runtime, which is waiting for an event to occur.

Use
Type E work items are not displayed in the Business Workplace.
Work Item with Express Notification

Use
When the system creates a work item with priority 1, each of its recipients receives an express notification (dialog box with appropriate text) on the screen. The recipient can call the Business Workplace directly from the express message. Excluded agents do not receive a message.

Constraints
The system does not create an express message
- if the user processes the work item immediately due to advance with immediate dialog [Seite 1453].
- if the work item represents a general task that is not restricted to certain agents.
- if the work item was forwarded.

If one of the selected agents processes the work item, the other agents still receive an express message.

Features
Express messages are only sent for dialog steps as soon as the system has created them with the ready status. If the work item is created first with the waiting status because its requested start has not been reached yet, the express message is not sent until the status changes from waiting to ready.

Activities
How is the priority set?
The priority of a work item can be determined for steps that require a dialog with the user. It is defined in the tab page Miscellaneous of the step definition.
Workflow Settings

Use
You use the workflow settings to maintain the special workflow functions in the Business Workplace.

Features
The following functions are available:

- **Personal settings** [Seite 1395]
- **Display organizational assignment** [Seite 1397]
- **Refresh organizational environment** [Seite 1397]
- **Adopt substitution** [Seite 1400]
- **End substitution** [Seite 1400]
- **Maintain substitute** [Seite 1400]
- **Activate substitute** [Seite 1400]
- **Adopt view** [Seite 1404]
- **Exit view** [Seite 1404]

Activities
You can access the workflow settings within the **Business Workplace** [Seite 1368] via **Settings** → **Workflow settings**.
## Personal Settings

### Use
The personal settings for workflow enable you to adapt the runtime system to suit your requirements.

### Features

#### Work item display
<table>
<thead>
<tr>
<th>Setting</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>User view with ActiveX (32-bit)</td>
<td>Both of these settings define the work item display as the default. Note that the ActiveX variant is only available on 32-bit platforms.</td>
</tr>
<tr>
<td>User view without ActiveX</td>
<td>This setting defines the technical work item display as the default.</td>
</tr>
</tbody>
</table>

#### Workflow log
<table>
<thead>
<tr>
<th>Setting</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>User view with ActiveX (32-bit)</td>
<td>User view of the workflow log without any technical aspects. This view uses ActiveX controls and is therefore only available on 32-bit platforms.</td>
</tr>
<tr>
<td>User view without ActiveX</td>
<td>User view of the workflow log without any technical aspects. This view uses the hierarchical list display of the ABAP List Viewer.</td>
</tr>
<tr>
<td>Technical view</td>
<td>In addition to the main semantic information, the technical view of the workflow log also displays technical numbers and texts that may not be available in the logon language of the user. The technical view is intended primarily for system administrators.</td>
</tr>
</tbody>
</table>

#### Further settings
<table>
<thead>
<tr>
<th>Setting</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Display work item texts in logon language</td>
<td>All work item texts in the Business Workplace are always displayed in the user’s logon language. If the user’s logon language is different to the language in which the work item was created, additional database accesses are required that can delay display of the Business Workplace.</td>
</tr>
<tr>
<td>Enable forwarding of work items to several users</td>
<td>The work item can be forwarded to several agents. The term “agent” is used in a broad sense in this context. You specify the agent as an organizational object (organizational unit, job, position, user) when forwarding the work item. This organizational object can consist of several persons. For further information, refer to Forwarding.</td>
</tr>
<tr>
<td>(Double)-clicking on an object displays the object in the same window</td>
<td>You can choose an object in the work item display, which is then displayed in the current window.</td>
</tr>
</tbody>
</table>
(Double)-clicking on an object displays the object in the same window.

You can choose an object in the work item display, which is then displayed in the current session window and replaces the work item display. When you exit the object display, you return to the work item display. If you do not set the indicator, a new session is created to display the object.

This setting does not apply if you work with the SAP GUI for HTML.

No tips & tricks in workplace

The Business Workplace [Seite 1368] includes tips and tricks for working with work items and workflows in the work item preview. If you set this indicator, the tips and tricks are not displayed.

No HTML in execution of decision tasks.

You use this indicator to decide whether the execution of decision tasks is displayed in HTML or not. Deactivating the HTML display may make sense, for example, if you have problems displaying HTML documents generally because of the settings in your local HTML browser.

### Activities

To change the personal settings for workflow, you choose `Settings → Workflow settings → Personal settings` in the Business Workplace.

The settings are saved as user parameters and take effect the next time the work item display or workflow log is called.

You can change the standard values for these personal workflow settings in Customizing for SAP Business Workflow. These standard values are always used if users have not maintained settings of their own.
Organizational Assignment of a User

Use
The worklist displayed is formatted specifically for the user who is logged on. You can query this user's organizational assignment.

Prerequisites
The functionality described depends on a setting in the PD system table [Seite 1533].

Features
The organizational assignment of a user is buffered and read at each new logon or once a day (but not each time the Business Workplace is called).
If the organizational assignment of a user changes while the user is logged on and must be updated, you choose Settings → Workflow settings → Refresh organizational environment in the Business Workplace.

Activities
To view current information on the assignment of the user logged on in the organizational plan of the enterprise, choose Settings → Workflow settings → Display organizational assignment in the Business Workplace.

The following information is displayed for the user logged on:
- The name maintained in the user address
- The organizational unit
- The position the user occupies
- The job describing the position

By double-clicking on an entry, you can display the staff assignments or the job description.
Substitutes for Workflows

Use
SAP Business Workflow implements a substitution arrangement to deal with a user's duties in their absence.

Prerequisites
The automatic appearance of work items in the substitute's worklist is dependent on a setting in the PD system table [Seite 1533]. It is possible to define which objects can be entered as position-related substitutes in Administration of Organizational Management [Seite 1530].

Features

Who is involved?
Substitution always involves two users:
- One user specifies a substitute: User B
- The other user (the substitute) 'adopts' the substitution: User A

The substitute can process “foreign” work items for the duration of the substitution. It is irrelevant whether they are assigned to the underlying single-step tasks as a possible agent.

How does substitution work?
Substitution works in two ways:
7. User A adopts substitution for user B and for the duration of the substitution sees only the work items seen previously by B in their workflow inbox.
   
   User B must have entered user A as their substitute for a particular period without activating the substitution. During this period, A can adopt substitution for B at any time without having to confer with B.
   
   Instead of their own worklist, user A is displayed user B's worklist. B's name is in the column Substitution for in the worklist, as long as this column is included in the current configuration of B's Business Workplace.

8. User A automatically sees not only the work items previously seen by employee B, but also their own work items in their Business Workplace. This also applies to all work items generated for B in the future (“automatic forwarding”).

   User B must enter user A as substitute and activate the substitution. No further action on the part of user A is required for this kind of substitution.

   A can continue working as usual. They see a Business Workplace to which B's work items are added dynamically. A can recognize these work items by the fact that B's name is in the column Substitution for, as long as this column is included in the current configuration.

In both cases, user A can perform operations on these "foreign" work items within the scope of the substitute profile assigned to them by B. It is irrelevant whether A is assigned to the underlying single-step tasks as a possible agent.

User B can continue working without any restrictions in both cases.
Activities
You can access substitute maintenance in the Business Workplace [Seite 1368] by choosing Settings → Workflow settings. For information on the individual functions, refer to Maintaining and Activating Substitutes [Seite 1400]
Maintaining and Activating Substitutes

Maintaining substitutes

4. In the Business Workplace [Seite 1368], choose Settings → Workflow settings → Maintain substitute.

The Maintain substitute dialog box appears.

To facilitate maintenance of the substitution, the Personal substitutes entry and the positions you occupy are displayed in the hierarchy on the Maintain substitutes screen.

You specify your substitute either as a personal substitute or as a position-related substitute.

- **Personal substitute**
  
  A personal substitute can see and execute all your work items, including those assigned to you via a personal agent assignment.
  
  You can only specify one other user as a personal substitute.

- **Position-related substitute**

  A position-related substitute can only see and execute the work items you have received on the basis of agent assignment at the level of position, job or organizational unit.

  You can enter another position or a user as a position-related substitute.

17. Position the cursor either on the entry Personal substitutes or on the relevant position, and select Create substitute.

You can maintain existing entries by double-clicking on the relevant entry.

If you want to maintain a position-related substitution, decide whether you want another position or another user as a substitute. Select either position or user as the substitute type and specify the number or the user name of the substitute.

18. Specify the validity period for the substitution on the detail screen for substitution.

Only within this period can the substitute adopt the substitution.

19. Specify a substitute profile in the dialog box Detail screen substitution.

Irrespective of whether you have created your substitute as a personal or position-related substitute, you can limit the scope of the work items displayed to your substitute by specifying a substitute profile.

20. Select the field Substitution active if applicable.

You must activate the substitution if you want work items to be visible for the substitute automatically from now on. If you do not activate the substitution, the substitute must adopt the substitution explicitly to see your work items.

21. Save your entries and exit substitute maintenance.
Activating substitutes
   The Activate substitute dialog box appears.
8. Select the substitutes that you want to activate, and choose the function Activate.

Deactivating substitutes
   The Activate substitute dialog box appears.
8. Select the substitutes that you want to deactivate, and choose the function Deactivate.

Adopting substitution
10. In the Business Workplace, choose Settings → Workflow settings → Adopt substitution.
    The Choose substitution dialog box appears.
11. Select the user(s) for which you want to adopt substitution.
12. Exit the dialog box.

Ending substitution
4. In the Business Workplace, choose Settings → Workflow settings → End substitution.
   The substitution is ended.
Views

Use
The Business Workplace [Seite 1368] provides various views on the work items displayed in the workflow inbox.

Using an appropriate view, a superior can "see" and process the work items of their employees.

Choosing a particular view also gives you the opportunity to see work items of other users in your workflow inbox and process them with full functionality, although the underlying tasks are not organizationally assigned to you.

Prerequisites
Views only require action by an employee: This employee chooses a view from a catalog of defined views. The employee must have the authorization required to choose a particular view.

Authorizations
To choose a particular view, you require a corresponding authorization based on authorization object S_WF_LVIEW.
To maintain a particular view, you must have the relevant authorization. This is an authorization based on the authorization object S_TABU_DIS to maintain tables for authorization group SWES.

Features
Views are always based on an evaluation path [Seite 1534] starting from the employee who wishes to adopt a view and leading to the employees whose inboxes could be viewed. The employee adopts a view by selecting another employee from the result list of the evaluation.

Activities
The activities associated with this function include:

- Adopting and Exiting Views [Seite 1404]
- Maintaining Views [Seite 1403]
Maintaining Views

Prerequisites

A view is always based on an evaluation path. This evaluation path describes which relationships are traced from the user who wants to adopt the view to the users whose Business Workplaces can be viewed.

You can use one of the evaluation paths available in the system. If there are no suitable evaluation paths, you can define an evaluation path of your own.

Different views can only be maintained as a customer setting.

Procedure


This displays the screen Change View "View for Maintaining Views": Overview.

Maintaining views is an activity that is described in the Implementation Guide and can be performed in Customizing.

23. Create a new view. To do this, choose the function New entries.

24. Assign a unique name to the view.

25. Specify an evaluation path.

26. Describe the view with a long text.

27. Specify a start evaluation path.

This selection is optional. The start evaluation path is used to get an initial selection of objects, which is then evaluated further via the first evaluation path.

28. Select Save.
Adopting and Exiting Views

Procedure

Adopting views

10. To adopt a view as the standard view (= view on your own work items), choose Settings → Workflow settings → Adopt view in the Business Workplace.

   You are now on the dialog box View: Choose Agent. Only the views for which you have an authorization are available.

11. Choose a view.

   On the basis of the evaluation path defined for the view, the system selects the positions, organizational units or users connected to you.

12. From the result list of this selection, choose the object whose workflow inbox you want to view.

Exiting views

A view is only active while the Business Workplace is displayed. The next time you call the Business Workplace, you are asked if you wish to adopt the view previously set.

To return to the standard view when working in the workflow inbox of the Business Workplace, choose Settings → Workflow settings → Exit view.
Dynamic Columns for the Business Workplace

Use
Up to 6 columns in the workflow inbox of the Business Workplace [Seite 1368] can be filled on a task-specific basis with contents that are determined dynamically at runtime. The standard functions for filtering, sorting, and grouping are available for these columns. Please compare with Selectable Columns for the Business Workplace [Seite 1406].

Features
If you want to include one of the “dynamic columns” into the workflow inbox display, you must specify an element from the task container for each task, from which the content of the column is established at runtime.

Work items that belong to different tasks are then also displayed with different information. Work items that belong to tasks for which this functionality is not used are displayed with a blank entry.

Activities
The column contents are maintained via Tools → Business Workflow → Development → Definition tools → Worklist client → Dynamic columns for worklist.

You can also define the column headings. These headings are displayed if all of the work items displayed in the Business Workplace refer to the same task.
Selectable Columns for the Business Workplace: Workflow

The columns displayed essentially determine the appearance and information content of the workflow inbox of the Business Workplace. Detailed knowledge of the columns is also required to make full use of the filter and grouping criteria.

You can determine the selection of columns via display variants. The following columns are available:

<table>
<thead>
<tr>
<th>Column</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>Work item is executable</td>
<td>Indicator denoting that the work item is executable.</td>
</tr>
<tr>
<td>Work item title</td>
<td>Title of work item.</td>
</tr>
<tr>
<td>Status indicator</td>
<td>Status of work item. In symbol form.</td>
</tr>
<tr>
<td>Creation date</td>
<td>Date when the work item was created with the status ready or waiting for the first time. A work item is only created with status waiting if a requested start was declared for the work item and the work item is created before the requested start.</td>
</tr>
<tr>
<td>Creation time</td>
<td>Creation time of a work item.</td>
</tr>
<tr>
<td>Priority</td>
<td>Priority of the work item.</td>
</tr>
<tr>
<td>Attachments exist</td>
<td>Attachments are shown with the symbol.</td>
</tr>
<tr>
<td>End of processing must be confirmed</td>
<td>Indicator denoting that the end of processing must be confirmed explicitly.</td>
</tr>
<tr>
<td>Work item overdue</td>
<td>Indicator denoting that a deadline has been missed for the work item.</td>
</tr>
<tr>
<td>ID</td>
<td>Unique number of a work item, which is assigned internally by the system.</td>
</tr>
<tr>
<td>Type</td>
<td>Type of work item. In abbreviation form.</td>
</tr>
<tr>
<td>Task</td>
<td>Identification for the task represented by the work item (for example TS00008323).</td>
</tr>
<tr>
<td>Technical status</td>
<td>Status of work item. Technical name.</td>
</tr>
<tr>
<td>Work item type</td>
<td>Type of work item. Name</td>
</tr>
<tr>
<td>Task name</td>
<td>Name of task</td>
</tr>
<tr>
<td>Work item status text</td>
<td>Status of work item. Name</td>
</tr>
<tr>
<td>Deadline status</td>
<td>The deadline status specifies whether one of the deadlines has been missed. The possible values in this column are therefore: None-Latest start-Requested end-Latest end-Other</td>
</tr>
<tr>
<td>Current agent</td>
<td>Name of the user who last reserved or processed the work item.</td>
</tr>
<tr>
<td>Selectable Columns for the Business Workplace: Workflow</td>
<td></td>
</tr>
<tr>
<td>------------------------------------------------------</td>
<td></td>
</tr>
<tr>
<td><strong>Latest end date</strong></td>
<td>Latest end of work item. The end is reached when the work item assumes the status <em>completed</em>.</td>
</tr>
<tr>
<td><strong>Latest end time</strong></td>
<td>Latest end time of a work item.</td>
</tr>
<tr>
<td><strong>Forwarder</strong></td>
<td>Name of the party who forwarded the work item.</td>
</tr>
<tr>
<td><strong>Substitution for</strong></td>
<td>Name of the substituted party whose work item is displayed.</td>
</tr>
<tr>
<td><strong>Work item content</strong></td>
<td>Column in which the default attribute of the object referenced in the container element <code>_WI_Object_ID</code> is displayed.</td>
</tr>
<tr>
<td><strong>Group object</strong></td>
<td>Column in which the default attribute of the object referenced in the container element <code>_WI_Group_ID</code> is displayed.</td>
</tr>
<tr>
<td><strong>Execution can be rejected</strong></td>
<td>Indicator denoting whether execution of the work item can be rejected ( ).</td>
</tr>
<tr>
<td><strong>Dynamic columns</strong></td>
<td>Refer to Dynamic Columns for the Business Workplace [Seite 1405].</td>
</tr>
</tbody>
</table>
Workflow Inbox

Use
The worklist of the user currently logged on to the Business Workplace is displayed in the workflow inbox.

Integration
As is the case for the workflow resubmissions [Seite 1443] and the workflow outbox [Seite 1441], the workflow inbox is an integral part of the Business Workplace.

Features

Views in the workflow inbox
A user's worklist can be displayed as an overview or according to the following grouping criteria:

<table>
<thead>
<tr>
<th>Grouped according to task</th>
<th>The work items are grouped according to the tasks to which they belong.</th>
</tr>
</thead>
<tbody>
<tr>
<td>Grouped according to content</td>
<td>The work items are grouped according to the object instances to which they belong.</td>
</tr>
<tr>
<td>Grouped according to content type</td>
<td>The work items are grouped according to the object types to which they belong.</td>
</tr>
<tr>
<td>Grouped according to sort key</td>
<td>The work items are grouped according to sort keys. Please refer to Grouping According to Sort Keys [Seite 1432].</td>
</tr>
</tbody>
</table>

You can also choose from the following views:
- Overdue entries
- Deadline messages
- Incorrect entries

Workflow Functions
The screen area in which the worklist is displayed has a toolbar containing the workflow functions and functions for configuring the list. For information about these functions, refer to the documentation on the ABAP List Viewer [Exterm].

The workflow functions can be accessed in the Business Workplace via the toolbar or the relevant context menu (right-hand mouse button). The following functions can be executed on work items:

- **Update**
  This function updates the worklist [Exterm] of the user currently logged on to the Business Workplace.

- **Execute**
  To be executed, a work item must have either the status ready or the status in process.
  When a dialog work item [Exterm] is executed, the object method to which the single-step task for this work item refers is carried out.
  When a missed deadline work item [Exterm] is executed, the most important information on the monitored work item is displayed.

- **Display work item**
  This function goes directly to the work item display [Seite 1411]
- ** Reserve** (dialog work items only)
  
  This reserves a work item for execution by the end user in question. The work item must have the status *ready*. This work item is then no longer visible to the other recipients who could previously see it in status *ready*. The status of the work item changes from *ready* to *reserved*.

- ** Replace** (dialog work items only)
  
  Replacing releases the work item from its reserved status and makes it available again to all of the original recipients. The status of the work item changes from *reserved* back to *ready*. The status of the work item must be *reserved* before it can be replaced.

- ** Forward**
  
  Forwarding [Seite 1435] passes a work item on to another employee for execution.

- ** Resubmit**
  
  If a user chooses this function, the selected work item is placed in workflow resubmissions [Seite 1443].

- ** Display workflow log**
  
  This function displays the workflow log [Seite 1420].

- ** Manage attachments** ... (Functions for attachment management [Seite 1429], dialog work items only)
  - Display attachments
  - Create attachments
  - Change attachments
  - Delete attachments

- ** More functions**
  - Set to 'Done' [Seite 1440]
  - Reject execution [Seite 1430]
  - Execute together [Seite 1431]
  - Change priority [Seite 1433]
  - Send mail [Seite 1425]
  - Change work item [Seite 1438]

- ** Environment**
  - Display objects in workflow (dialog work items only)
  - Display workflow relationships [Seite 1436]
  - Start Workflow [Seite 1450]
  - Create link [Seite 1434]
Activities

You can access the workflow inbox in the tree on the left in the Business Workplace via Workplace → Inbox.
Work Item Display: Standard View

Use
The objective of the work item display is to display all the information and activities that are relevant to an end user in the environment of the displayed work item in a clear and concise manner. The work item display supports the agent not only in the processing of the current work item but also in the compilation of an activity list, which then functions as the agent's personal worklist.

Integration
A technical work item display [Seite 1418] is available for work items of all other types. You can make this technical work item display standard for dialog work items as well. The standard view of the work item display can be shown with or without ActiveX. You make this setting for the work item display variant in your personal workflow settings [Seite 1395].

Features
The work item display is primarily designed for displaying dialog work items [Extern]. It contains details about deadlines, statuses, agents, attachments and linked objects for a work item. The work item display also enables an end user - providing they have the relevant authorization - to compile an activity list as their personal worklist. The work item display has three tab pages (Basic data, Activities, and Available objects).

- **Tab page Basic data [Seite 1414]**
- **Tab page Activities [Seite 1415]**
- **Tab page Available objects [Seite 1416]**
- **Customer-defined tab page for work item display [Extern]**
  
You can define another tab page, which is then displayed as the first tab page when the work item display is called.

Application toolbar functions

- **Execute**
  
To be executed, a work item must have either the status *ready* or the status *in process*. When a dialog work item is executed, the object method to which the single-step task for this work item refers is carried out.

  When missed deadline work items [Extern] are executed, the most important information on the monitored work item is displayed.

- **Display last message**
  
The return code that was returned to the workflow system after the object method was executed can be retrieved for processed work items using the *Messages* function.

- **Forward**
  
Forwarding [Seite 1435] passes a work item on to another employee for execution.

- **Resubmit**
If a user chooses this function, the selected work item is placed in workflow resubmissions [Seite 1443].

- **Change priority**
  Refer to Changing Priorities [Seite 1433].

- **Change deadlines**
  Refer to Changing Deadlines [Seite 1428].

- **Display/change attachments**
  For attachments, refer to Attachment Management [Seite 1429].

- **Reserve** (dialog work items only)
  This reserves a work item for execution by the end user in question. The work item must have the status *ready*. This work item is then no longer visible to the other recipients who could previously see it in status *ready*. The status of the work item changes from *ready* to *reserved*.

- **Replace** (dialog work items only)
  Replacing releases the work item from its reserved status and makes it available again to all of the original recipients. The status of the work item changes from *reserved* back to *ready*. The status of the work item must be *reserved* before it can be replaced.

- **Mail**
  Refer to Send Mail [Seite 1425].

- **Display workflow log**
  This function displays the workflow log [Seite 1420].

- **Display graphical workflow log**
  This function displays the graphical workflow log [Seite 1424].

### Additional functions in the menus

#### Work Item
- **End resubmission**
  The work item is put back into the workflow inbox. It has the status *reserved*.

- **Create link** [Seite 1434]
- **Reject execution** [Seite 1430]
- **Set to ‘Done’** [Seite 1440]

#### Goto
- **Methods**
  You can use this function to execute the defined secondary methods [Extern] of the work item displayed from the work item display.
  A prerequisite for this is that the work item represents an activity that is part of a workflow. At least one secondary method must be defined for this activity.
Workflow description

The description text of the superordinate multistep task, the "process description", is displayed.

The work item must be part of a workflow and a description text must be maintained for the workflow.

• Technical work item display [Seite 1418]

Extras

• Displaying tasks

This function can be used to display the definition of the task represented by the work item. Refer to Definition of a Single-Step Task [Seite 1175] and Definition of a Multistep Task [Seite 1194].

• Technical data

Technical data about the work item, for example its ID, its texts or the ID of the superordinate work items are displayed.

• Organizational Assignment [Seite 1397]

• Displaying Agents [Seite 1427]

Activities

You can access the work item display by:

• Selecting a work item in the Business Workplace and choosing .
• Double-clicking one of the entries for a step in the workflow log.
• Selecting an entry that does not represent a workflow in the work item selection [Seite 1490] hit list displayed. (If you choose a workflow, the workflow log is displayed.)
Tab Page Basic Data

Use
The information on this tab page of the work item display [Seite 1411] is mostly self-explanatory. Except for the priority, you cannot make any changes here.

Features

Deadlines
These are the deadlines monitored by the runtime system. (Deadlines that are not set are displayed without a date.)
- Start by (latest start)
- End by (latest end)
Depending on whether the work item represents a step in the workflow or a task, the deadlines were either specified when this step was defined in the workflow definition or when the task was started online.
To display all of the deadlines of the work item, choose Work item → Deadlines.

Further Information

- Forwarded by
  If the work item was forwarded to you, the name of the forwarder is entered here.
- Priority
  The priority of a work item is derived from the definition of this step in the workflow definition. The priority is used as a sort criterion for positioning the work item in the Business Workplace.
  The priority can be changed here.
- Status
  The current processing status is expressed by the work item status [Seite 1373].
- Creation date (created on) and processing date (processed from)
  These are the actual dates and times (when the work item was created and when processing was started).
Using the Messages function, you can display the return codes for processed work items, which were returned to the workflow system after execution of the object method.

Work Item Description
A description of the work item to be executed is provided at the bottom left of this tab page. The task description is entered in the task definition. It is used for information purposes and generally contains instructions and recommendations on processing the work item displayed.

Attachments
The titles of all the attachments added to this work item or, if the work item is part of a workflow, to the preceding work items are shown in the lower right of this tab page.
Tab Page Activities

Use

The Activities (not yet processed) list contains all the activities that are relevant for processing this work item. The tab page Activities is part of the work item display [Seite 1411].

Features

The work item text of the task represented by the work item is generally at the start of the list ("main activity"). Once this activity has been processed (and completion of processing has been confirmed, if necessary), the status of the work item changes to completed. No other actions can then be carried out in the work item display.

Activities

You can extend the activity list and in this way create a worklist. The activities added represent your "personal worklist" as end user (agent).

<table>
<thead>
<tr>
<th>Function</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>✔️ Done</td>
<td>An activity selected from the list is reported to be done. This functionality cannot be executed on the leading activity (first line) of this work item. The agent is responsible for reporting that an activity is done. The actual processing is not checked.</td>
</tr>
<tr>
<td>✗ Execute</td>
<td>An activity selected from the list is executed. The main activity (first line) of the work item can also be executed via the menu path Work item → Execute.</td>
</tr>
<tr>
<td>📖 Create</td>
<td>Another activity is added to the list. This activity can entail</td>
</tr>
<tr>
<td></td>
<td>- Executing a method on an existing object (created invoice, created material, etc.)</td>
</tr>
<tr>
<td></td>
<td>- Creating a new object (text, etc.)</td>
</tr>
<tr>
<td></td>
<td>When you create an activity, you are given step-by-step support in dialog by a &quot;wizard&quot;.</td>
</tr>
<tr>
<td>🗑️ Delete</td>
<td>An activity is deleted from the list.</td>
</tr>
</tbody>
</table>

Scrolling in the activity list: The activity list contains extensive information on each activity, which you can view by scrolling to the right. This information includes the following:

- Description
- Creator of the activity with date and time
- Actual agent of the activity with date and time
- Method and object type
Tab Page Available Objects

Use
All objects that are related to the work item are displayed in the list of available objects. These objects are stored in container elements of the task container as object references. You can display these objects or add new objects to the container element.
There are:
- Ad hoc objects
  Objects added to a work item in this or one of the preceding steps of the workflow (container element _Adhoc_Objects of the task container)
- Attachment objects
  Documents added to a work item in this or one of the preceding steps of the workflow (container element _Attach_Objects of the task container).
- Process objects
  - The object currently being processed (container element _WI_Object_ID of the task container)
  - The object added for grouping purposes (container element _WI_Group_ID of the task container)
  - Objects that are referenced in other elements of the task container

The tab page Available objects is part of the work item display [Seite 1411].

Features

Displaying objects with their default attributes
Each object referenced in the work item container is displayed with its default attribute [Extern]. The default method [Extern] of each object can be executed upon request. If no default attribute was defined for the object type, the key fields of the object are displayed.

Adding objects
You can extend and process the list of objects. The main purpose of this is to make the relevant information available to the agents of the subsequent steps in the workflow as well. Only object types [Extern] that support the IFFIND interface can be selected. You identify an actual object [Extern] of this type by specifying its key fields [Extern].

Activities
To execute the functions displayed, proceed as follows:

<table>
<thead>
<tr>
<th>Function</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Create</td>
<td>An object is added to the list. When you add an object, you are given step-by-step support in dialog by a &quot;wizard&quot;.</td>
</tr>
<tr>
<td>Display</td>
<td>The default method of an object selected from the list is executed.</td>
</tr>
<tr>
<td>Remove</td>
<td>An object is deleted from the list.</td>
</tr>
</tbody>
</table>
Workflow Log: Standard View

Use
The workflow log [Extern] enables you to format all the information that is generated or collected during the execution of one active business process in one central location. The standard view described here is intended, in particular, for agents and process controllers who want to get an overview of the steps processed so far.

Prerequisites
To use the view described here, you must have set a view with ActiveX in your personal workflow settings [Seite 1395].

Integration
Other options for displaying the workflow log are:
- Workflow Log: Technical View [Seite 1422]
- Graphical Workflow Log [Seite 1424]

Features

Application toolbar functions
The following functions are available on the application toolbar:
- Update
- List with technical details
  This function takes you to the technical view of the workflow log.
- Graphical workflow log
- Personal workflow settings
  You can use this function to maintain your personal workflow settings.

Tab pages
The system shows the various views on the workflow log on different tab pages.

Tab page Workflow Chronicle (what was processed when?)
The tab page Workflow Chronicle shows a hierarchical display of all steps in the workflow, which have been processed so far or are due to be processed. If the workflow has a subworkflow structure, the subworkflows are also displayed.
The Details function (symbol) lists the following information about each step in the lower part of the screen:
- Who carried out what detailed actions for these work items and with what results.
- When these actions were carried out.
- Which objects were involved.
The Agents function (symbol) displays the selected and possible agents of a step.
The Graphic function (symbol) displays the graphical workflow log.
Tab page Workflow Agents (who processed what?)
The tab page Workflow Agents shows the employees involved in this workflow up to now. The following is displayed for each employee:
- What action was carried out in what step.
- When this action was carried out.
- Which objects were involved.
This view shows how an employee was involved in the execution of a workflow.

Tab page Workflow Objects (what was processed?)
The tab page Workflow Objects lists the objects related to the workflow or addressed up to now in the execution of the workflow. These objects include:
- The “leading” object of the workflow.
- Any attachments and objects added in the individual steps of the workflow.
The following is displayed for each object:
- Who carried out what detailed action for what task.
- When this action was carried out.
This view shows what information was generated and processed, and how.

Information at the click of a mouse
You can view all the information provided in the workflow log using the mouse.
You can also go to the work item display [Seite 1411] for each dialog step. You can display address data for agents as well as the contents of work item attachments or the result of actions that have been executed.

Activities
You can access the workflow log from the work item display or the Business Workplace [Seite 1368] via the icon.
Send Mail

Purpose
You want to send a mail to another user in connection with a work item that requires processing. The work item being processed is therefore also to be made accessible to this user for information purposes. The Send mail function enables you to send mails to any recipients with a text which can be entered freely. These mails are automatically linked to your work item in such a way that when the mail is executed by the recipient the work item is displayed.

Process Flow

Procedure as sender of mail
10. In the Business Workplace's workflow inbox, position the cursor on a work item and choose Other functions → Send mail.
    If you are already in the work item display, choose Work item → Send mail.
11. Enter the text of the mail.
12. Send the mail to any recipients who are available as mail addressees
    The mail text is added to the work item as an attachment. The work item remains in your Business Workplace with the same status.
You can still complete the work item despite the fact that you have sent a mail. Since this makes the mail which has been sent superfluous, the recipient of the mail receives a second mail informing him/her that the (first) mail no longer needs to be dealt with.

Reply ends resubmission
Optional: If you do not want to process the work item until the mail has been replied to, define a resubmission for the work item. To do this, position the cursor on the entry, choose Resubmit and enter a date in the future.
The work item is resubmitted irrespective of this date and appears in your workflow inbox when the reply to the mail is received.

Procedure as recipient of a mail
You receive a mail that can be executed.
13. Read the mail.
14. Execute the mail, if applicable. The work item referred to by the sender in their mail is displayed. For further information on the object to be processed, choose Goto → Object to display the default attribute or execute the default method (generally Display) for the processed object.
    You can execute this work item if you are one of the possible agents of this task.
15. Reply to the mail. To do this, choose Document → Reply or Document → Reply w/reference when the mail is displayed. Then enter your reply and save your entries.
16. Choose Document → Send. On the send screen, the work item to which the mail referred is entered as the "recipient". Do not change this entry. Send the reply to the proposed recipient.
Result

The reply sent ends a resubmission which may have been defined for the recipient work item. This work item is appended as an (additional) attachment and can be read as such by the original user and taken into account.
Display Agents

Use
This function displays information about the agents of the work item.

Features
The system displays the following agents for the work item:

- **Recipient [Extern]**
- **Possible agents [Extern]**
- **Excluded agents [Extern]**

You can choose a compressed display - containing only the user names (*users only* function) - or a complete display with additional information on the relationships used by the system to determine the agents (*overall view* function). You can also display the organizational assignment of each user.

Note that this information is only available for work items of types for which an agent is logical and necessary.

Activities
You can display the agents from the work item display [Seite 1411] by choosing **Goto → Agents → ...**
Changing Deadlines

Use
You use this function if you want to change the deadlines for a work item at workflow runtime. This dialog box displays all the deadlines of work item processing.
Please also refer to Current Dates/Times of a Work Item [Extern].

Features
Depending on whether the work item represents a step in the workflow or a single-step task, the deadlines were either specified when this step was defined in the workflow definition or when the single-step task was started in dialog.
The dialog box is split into the following sections:

Deadlines
- **Start by** (latest start)
- **End by** (latest end)

Planned deadlines
- **Start by** (requested start)
- **End by** (requested end)

A monitored deadline shown in blue is in the past. If is also displayed, the appropriate action has been initiated. This generally involves informing the deadline recipient.

Actual dates/times
- **Created on**: Creation date/time of the work item
  (Technically: The work item is created with the status ready or - if the requested start date has not yet been reached - with the status waiting).
- **Processed from**: Start of processing
  (Technically: Transition of the work item to the status in process).
- **Completed on**: End of processing or date when set to ‘done’.
  (Technically: Transition of the work item to the status completed.)

Activities
To execute this function, choose in the work item display [Seite 1411].
Attachment Management

Use
One or more attachments can be assigned to each work item that appears in the Business Workplace's workflow inbox. Attachments are documents written either with a SAPscript editor (document classes RAW, SCR) or with a PC application (document classes DOC, URL, PPT, XLS, PDF, ...) and then imported. You can enter new documents as attachments or create attachments from existing files.

Features

General
The attachment is automatically
- Added to the work item container
- Added to the container of the superordinate workflow
- Added to the containers of the subsequent work items in the workflow

You can define default documents for the individual document classes. For further information, refer to Default Documents [Extern].

Attachments can be displayed by the recipients of the subsequent steps. But they cannot be changed and, therefore, have a document character.

A superior who is to make a decision on releasing a budget can enter an attachment justifying their decision. The selected agents of the subsequent steps can display this attachment.

If a work item has attachments, this is indicated by a symbol in the Attachments column in the Business Workplace. You can also execute the function for processing an attachment by double-clicking in this column (column header AT). If an attachment already exists, it is displayed.

If a work item has attachments, this is indicated by the symbol in the work item display.

Functions on attachments

<table>
<thead>
<tr>
<th>Display</th>
<th>Displays the attachment selected in the dialog box Existing attachments.</th>
</tr>
</thead>
<tbody>
<tr>
<td>Create</td>
<td>Creates a new attachment. If you want to import an existing document from your PC to the attachment, choose Import.</td>
</tr>
<tr>
<td>Change</td>
<td>Displays the selection of Changeable attachments.</td>
</tr>
<tr>
<td>Delete</td>
<td>Displays the selection of Deletable attachments.</td>
</tr>
</tbody>
</table>

Activities

You can access attachment maintenance in the Business Workplace by choosing or the relevant context menu (right-hand mouse button).
Reject Execution

Use
You can use this function if you need to reject execution of a work item for business or technical reasons. This function is only available for work items of type W.

The table entry to be processed already exists or the material whose master data is to be changed is no longer used.

Prerequisites
This function is only available if the property processing rejectable has been selected for the related activity in the workflow definition.

Features
Processing of the work item is terminated with the reject execution function. The subsequent steps defined in the workflow definition are executed.
Do not use this function if you do not want to or cannot process the work item for personal reasons (not responsible, not competent). In this case, replace the reserved work item or forward it.

Activities
You can access this function in the Business Workplace [Seite 1368] by choosing Other functions or the work item display (Work item → Other functions → Reject processing).
Execute Together

Use
You can select various work items and then execute them together. This function is only available for work items of type W.

The work items do not necessarily have to belong to the same workflow.

Prerequisites
The work items must refer to the same single-step task.
For information about other prerequisites for this function, in particular with regard to the underlying method, refer to Creating an "Execute Together" Method [Extern].

Features
Only the first of the selected work items is proposed for execution. The entries you make here also apply to the other work items to be executed together.

Activities
Select the work items that you want to execute together. Then choose Execute together via Other functions... in the Business Workplace [Seite 1368].
Grouping According to Sort Key

Use

Each work item carries the two container elements \_WI_Object_ID and \_WI_Group_ID in its container. Both elements have been defined to hold an object reference.

- The container element \_WI_Object_ID automatically contains the reference to the object to be processed in the work item.
- The container element \_WI_Group_ID can contain an object reference, which must be assigned to this container element in a binding or via initial value assignment.

The object reference assigned via \_WI_Group_ID is generally not identical to \_WI_Object_ID, but is derived from the work item execution environment. It is used to group work items that refer to different objects or object types but are nevertheless connected.

Features

Display default attributes in workflow inbox

The default attributes [Extern] of the objects referenced in \_WI_Object_ID and \_WI_Group_ID are available in the Business Workplace's workflow inbox [Seite 1408] under the column headers Object or Group for grouping, sorting and filtering purposes.

Material master data is processed in several instances of a workflow. The container element \_WI_Object_ID in the containers of the individual work items therefore always contains the reference to the object of type BUS1001 (material) to be processed.

The container element \_WI_Group_ID is assigned the expression &Material.Labor& (laboratory/drawing office for material as object reference) in the relevant steps of the workflow definition. The default attribute of the laboratory is therefore available as a group.

Default methods

The default method [Extern] of the objects referenced in \_WI_Object_ID and \_WI_Group_ID is executed by double-clicking in the Object or Group column.
Change Priority

Use
The priority of a work item is a measure of its urgency.
It can be used as a sort criterion for organizing the workflow inbox. This function is only available for
dialog work items.

Features
End users can display and change the priority of a work item. The priority is between 1 (highest) and 9
(lowest).
The change in priority can also be passed on automatically to the superordinate workflow and then to all
work items created subsequently. The Pass on priority to subsequent steps indicator must be selected for
this to take effect.

If a higher priority (lower number) is defined in the workflow definition for one of the
subsequent steps, it is not changed.

Activities
You can access the Change priority function in the Business Workplace by choosing Other
functions...
Create Link

Use
By storing links to work items in folders [Extern], you can organize your work effectively using a personal folder hierarchy.

Features
The work item is added to a personal or shared folder as a link.
This work item can be displayed from this folder and executed by its recipients.
Work items can have an unlimited number of links.

Activities
You can access this function in the Business Workplace [Seite 1368] by choosing Environment...
Forward Work Item

Use
Forwarding passes a work item on to another user for execution.

Integration
The range of users to whom a work item can be forwarded is determined from the task definition as follows:

<table>
<thead>
<tr>
<th>Task definition:</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Task is a general task [Extern]</td>
<td>Work item may be forwarded to all users.</td>
</tr>
<tr>
<td>The indicator General forwarding allowed is set for the task</td>
<td>Work item may be forwarded to all users.</td>
</tr>
<tr>
<td>The indicator General forwarding not allowed is set for the task</td>
<td>Work item may only be forwarded to the recipients [Extern] of the task.</td>
</tr>
</tbody>
</table>

Features
The user has the following search options to select the new recipient:

- The F4 input help can be used to determine the new recipient by performing a generic search for name components.

- The structure search can be used to determine the new recipient using the graphical display of the organizational plan.

The new recipient does not have to be an actual user. (The work item can also be forwarded to an organizational unit or a job.)

Activities
You can access this function in the Business Workplace [Seite 1368] by choosing  

Workflow Relationships

Use
You can use this function to establish the work items in which a particular object is being processed. The following work item types are taken into account:

- **Dialog work items** (type W)
- **Background work items** (type B)
- **Workflow work items** (type F)

The list displayed is purely runtime information.

Prerequisites

Authorization
You require appropriate authorization for the functions shown. This is an authorization based on the authorization object `S_PROGRAM` for executing a program for authorization group 'SWI_OA'.

Features
You can call the function for displaying linked workflows from several starting points.

Calling via SAP Easy Access
Two functions are available via SAP Easy Access:

- **Workflows for object**
  This function displays all workflows linked to a particular object (for example a specific notification of absence). You must specify this object beforehand.

- **Workflows for object type**
  This function displays all workflows linked to a particular object type (for example `FORMABSENC`). You must specify this object type beforehand.

Calling from the Business Workplace
In this case, you do not specify the object. The workflow relationships are displayed for the object processed in the selected work item.

Calling as a generic object service
In this case, you do not specify the object. The workflow relationships are displayed for the object being processed.

What is displayed?
The display of the workflow relationships has two parts. The upper part displays the workflows identified for the object or object type. The lower part displays data on the currently selected work item in the form of a simplified workflow log:

- **Steps so far**
The workflow steps processed so far are listed under *Step name*. The steps are linked to the work item preview of the relevant work item. The current agents of the work item are listed under *Agents*. Click once to go into detailed display of the user data.

- **Information objects addressed so far**
  
  All the objects and attachments belonging to the workflow are displayed here. Click once to display the information objects.

### Activities

<table>
<thead>
<tr>
<th>Calling...</th>
<th>Path</th>
</tr>
</thead>
<tbody>
<tr>
<td>... Via SAP Easy Access</td>
<td><strong>Tools</strong> → <strong>Business Workflow</strong> → <strong>Development</strong> → <strong>Runtime Tools</strong> → <strong>Workflows For Object</strong></td>
</tr>
<tr>
<td>... From the <strong>Business Workplace</strong></td>
<td><strong>Display workflow relationships</strong></td>
</tr>
<tr>
<td>... As a generic object service</td>
<td><strong>System</strong> → <strong>Workflow</strong> → <strong>Workflow overview</strong></td>
</tr>
</tbody>
</table>
Change Work Item

Use
The function enables you to change information and data associated with the selected work item.

Features

Changes in the initial screen
In general, the changes that are allowed depend on the type and status of the work item. The following changes are possible:

- **Work item text**
- **Priority**
- **Deadline data**
  You can set and change deadline data for the start and end deadlines of the work item. You can only make specifications for deadlines that are still being monitored and are in the future. For example, dialog work items with the *ready* status can no longer be assigned a requested start.

Manual intervention into the processing of work items
If you are an administrator with the appropriate authorization, you can intervene manually into the processing and therefore into the process flow of a workflow. This is particularly helpful for dealing with errors. The following changes are possible:

- **Set to ‘ready’ manually**
  The work item status is changed from *waiting* to *ready*.
  The work item therefore appears in the workflow inboxes of the selected agents.

- **Complete manually**
  The work item status is changed to *completed*.
  In the case of dialog work items, this status change (together with the current work item container) is passed on to the workflow system for evaluation. If the object method to be executed is a synchronous method with a result, the possible result values are displayed for selection.

- **Replace manually**
  The status of the work item is reset to *ready*.
  The work item is therefore displayed again in the workflow inboxes of all selected agents.

- **Restart after error**
  Work items and, in particular, workflows with errors can be restarted after the errors have been corrected.
  The information written by the workflow system in a log when the error occurred is now taken into account.

- **Delete logically**
  The status of the work item is changed to *logically deleted*. 
In the case of dialog work items, this status change is passed on to the workflow system.

**Locking/unlocking execution**
An administrator can lock or unlock the current work item for execution. To do this, you select the appropriate function from the Edit menu.

**Deleting deadlines**
To delete a deadline you have changed or entered and reset it to its initial values, choose Edit → *Delete deadline*.

**Changing work item containers**
Choose Edit → *Change container* to go to the editor for changing the work item container. The contents of the container for the relevant work item are displayed. You can change the current, runtime-specific data for the particular work item. The container may still have elements that do not currently have a value. These elements are hidden as standard. The presence of elements that are not displayed is shown by a indicator. To display these elements, choose Edit → *Show elements*. To add more lines to container elements defined as multiline, choose Edit → *Additional line*.

**Activities**
You can access this function in the Business Workplace [Seite 1368] by choosing Other functions, the relevant context menu or the technical work item display [Seite 1418].
Set Work Item to Done

Use
This function is used by the recipient of a work item to confirm explicitly that processing of this work item has been completed. As long as explicit confirmation has not been provided, the work item has executed status and can be executed again or forwarded.

The status of the work item changes from executed $\rightarrow$ completed.

Prerequisites
This function is only possible on a work item if a setting was made in the definition of the associated single-step task stating that the end of processing must be confirmed explicitly.

Activities
You use this Business Workplace function if the status of the work item has been changed to executed by a terminating event.
In general, however, the work item is not set to done via the Business Workplace. This function is usually provided as a dialog box directly after a work item has been processed.

Missed deadline work items [Extern] must always be confirmed explicitly. After execution, they remain in the status in process until they are set to done.
You can access this function in the Business Workplace [Seite 1368] by choosing Other functions.
Workflow Outbox

Use
The workflow outbox lists the workflows started and the work items forwarded and executed by the current user.

Integration
As is the case for the workflow resubmissions [Seite 1443] and the workflow inbox [Seite 1408], the workflow outbox is an integral part of the Business Workplace.

Features

Views in the workflow outbox

Started workflows
This view shows work items for the tasks started by you as a user in dialog or by a triggering event whose event container contains your user name as _Evt_Creator.

Work items executed by me
The work items executed by you are displayed in this view.

Forwarded work items
The work items forwarded by you are displayed in this view.

Workflow Functions
The screen area in which the worklist is displayed has a toolbar containing the workflow functions and functions for configuring the list. For information about these functions, refer to the documentation on the ABAP List Viewer [Extern].

The following functions are available for the different workflow outbox views:

- **Update**
  You use this function to update the workflow outbox.

- **Display work item** (not in Started workflows view)
  This function goes directly to the work item display [Seite 1411]

- **Change agent**
  You use this function to perform an ad-hoc agent assignment [Seite 1097], [Seite 1411]

- **Display workflow log**
  This function displays the workflow log [Seite 1420].

- **Display attachments** (refer to attachment management [Seite 1429])

- **Environment**
  - Display objects in workflow (dialog work items only)
  - Display workflow relationships [Seite 1436]
  - Start Workflow [Seite 1450]
Workflow Outbox

- Create link [Seite 1434]

- Change selection period
  - Today only
  - Last 7 days
  - Last 30 days
  - User-defined ...

The functions described above can be called using the relevant context menu.

**Activities**

You can access the workflow outbox in the overview tree in the Business Workplace [Seite 1368] via Workplace → Outbox.
Workflow Resubmissions

Use
The workflow resubmissions [Extern] function can be used to resubmit work items for processing at a later point in time or periodically and to display these work items in a list. Technically, the following applies to work items which appear in your workflow resubmission folder: the work item status is set to waiting, your name is entered as the actual agent, and the requested start date is set to the resubmission date.

Integration
As is the case for the workflow inbox [Seite 1408] and the workflow outbox [Seite 1441], the workflow resubmissions is an integral part of the Business Workplace.

Features

Workflow Functions
The screen area in which the worklist is displayed has a toolbar containing the workflow functions and functions for configuring the list. For information about these functions, refer to the documentation on the ABAP List Viewer [Extern].

The following functions are available in the workflow resubmissions:

- **Update**
  You use this function to update the workflow resubmissions.

- **Display work item**
  This function goes directly to the work item display [Seite 1411]

- **Display workflow log**
  This function displays the workflow log [Seite 1420].

- **Display attachments**
  This function is only available if the work item has attachments. Refer to attachment management [Seite 1429].

- **End resubmission**
  The work item is put back into the workflow inbox. It has the status reserved.

- **Environment**
  - Display objects in workflow (dialog work items only)
  - Display workflow relationships [Seite 1436]
  - Start Workflow [Seite 1450]
  - Create link [Seite 1434]

The functions described above can be called using the relevant context menu.

Activities
You can access the workflow resubmissions in the overview tree in the Business Workplace [Seite 1368] via Workplace ➔ Resubmissions.
You are asked for a date for resubmission of the work item. The work item then disappears automatically from the workflow inbox and is not displayed there again until the date specified.
Work Item Preview

Use
The work item preview is in the bottom right corner of the Business Workplace screen and provides a preview of the properties of a work item. All the information on a work item is displayed if you choose the function Display work item in the worklist of the workflow inbox [Seite 1408] or the workflow resubmissions [Seite 1443]. This function goes directly to the work item display [Seite 1411]. A concise view of the workflow log [Seite 1420] is offered in the workflow outbox [Seite 1441] when the started workflows are displayed.

Integration
The work item preview is part of the Business Workplace [Seite 1368].

Features
Workflow functions
The description of the work item is displayed in the work item preview. It is also possible to access objects and attachments of the work item directly from the preview. The concise view of the workflow log enables you to access processed steps (individual work items), their agents and information objects addressed so far. For more information about the concise view of the workflow log, please refer to Workflow Relationships [Seite 1436].

Tips & tricks
"Tips & tricks" are displayed when you position the mouse pointer on the title of a "tips & tricks" entry. Whenever the work item preview is updated, a new title is offered. You can activate/deactivate "tips & tricks" in your personal workflow settings [Seite 1395] (Settings → Workflow settings → Personal settings).

User exit
The work item preview can be adapted to customer-specific requirements using a user exit [Extern].

Activities
The work item preview is activated/updated whenever you select a work item in your worklist.
Starting Tasks and Workflows

Use
There are various options for starting tasks and workflows. You must be aware of these options when defining a task or workflow.

Integration
A started workflow is represented by a workflow work item [Extern], and a started task by a dialog work item [Extern] or a background work item [Extern].

Features
Tasks and workflows can be started in the following ways.

By an event
The system starts the task or workflow when a triggering event defined for this purpose occurs. The following requirements must be met:

- The event must be defined as a triggering event in the definition.
- The event receiver linkage must be activated.

The linkage may be deactivated by the system if the start was unsuccessful. If this occurs, try to analyze and resolve the error by simulating an event [Seite 1363].

To use data stored with the event in the execution, a binding can be defined from the event container to the workflow container or task container.

A task or workflow can have several triggering events. If one of the triggering events is generated in the system, the relevant execution is started.

Triggering events are only required for a task if the task is to be executed as a single step in response to an event. If you only want to use the task as a step in a workflow, the task does not require any triggering events.

For further information, refer to Creating Triggering Events [Seite 1206].

You can also start a task or workflow defined with a triggering event on a test basis using this event. For further information, refer to Creation of Events for Test Purposes [Seite 1365].

Directly in dialog
You start the task or workflow yourself. For further information, refer to:

- Starting Workflows in Dialog [Seite 1450] for the standard environment at runtime and starting workflows with reference to an application object.
- Starting Workflows (Test Environment) [Seite 1452] for a test environment that allows tasks and workflows to be started.
- Definition of Start Transactions [Seite 1455] for workflows.

A task or workflow started by a triggering event is normally provided with input data from the event container. If you start a task or workflow of this kind in dialog, you
must enter the input parameters manually. Therefore, you will usually only start a task or workflow with a triggering event in dialog for test purposes.

**From a message long text**

This function is only available with messages for which an appropriate assignment is maintained. A user can start the relevant workflow when the message is triggered. For further information, refer to [Starting Workflows or Tasks from Message Long Texts](#Seite 1456).

**By a workflow**

The system starts a task or workflow if the task or workflow is integrated in another workflow as a step. If the task is executed in dialog, possible agents [Extern] must be assigned to it, or it must be classified as a general task.

To execute a task as a step in a workflow, the task **does not** require triggering events.

For further information, refer to [Maintenance of Activities](#Seite 1024).
Starting Workflows in Dialog

Use
This function can be used to start the workflows you have created in dialog. "In dialog" means that the workflow is not started by a triggering event but manually. It is irrelevant whether triggering events are defined for the workflow.
For further information, refer to Advancing With Immediate Dialog [Seite 1453].

Prerequisites
You can only start tasks in dialog if you are entered as a possible agent [Extern] for them.

Features
Starting workflows
The screen for starting workflows has two parts. The workflows you have defined are listed on the left. The description of the workflow selected on the left is displayed on the right.
You can use the standard toolbar to go to the workflow outbox [Seite 1441] and the Business Workplace [Seite 1368].
You can start a workflow by selecting it and then choosing either in the left screen area or the Start function in the right screen area.

Starting with details
The function Start with details can be used to add information such as attachments, notes or deadlines to the workflow via three tab pages. The attachments and notes are available to all agents of the workflow.

- Tab page Properties
  You can maintain the priority of the workflow, its latest end and its requested start here. If you set the indicator Create flag for started workflow in inbox, you get a work item in your inbox when the workflow is started, which calls the relevant workflow log during execution. The system deletes this flag automatically when the workflow is terminated.

- Tab page Note at start
  The note that you create here is added to the workflow as an attachment.

- Tab page 0 Attachments
  Here you can create ( ), display ( ), import ( ) and delete ( ) attachments for the workflow.
You start the workflow with the function Start workflow.

Starting via start transaction
If a start transaction [Seite 1455] is defined for the workflow, the defined transaction is started instead of detail maintenance.
As with detail maintenance, you can change the priority, add attachments and maintain deadline data via . In contrast to detail maintenance, you cannot create any new attachments. However, it is possible to create ad hoc objects (in contrast to attachments, these are assigned to the container element _Adhoc_Objects).
Make the entries required in the input fields you have created. This assigns values to the import elements of the interface of the workflow to be started. Use the F4 input help if required.
Starting workflows with object references

Starting a workflow with an object reference allows you to extend a process at runtime and initiate additional subsequent activities on an ad hoc and flexible basis. This function is not available everywhere. The function assumes that the system recognizes as a reference object a particular application object (for example document, invoice, item of master data) that you are processing.

The process flow is as follows:

- The system establishes a reference object [Extern] from the current call context.
  
  The current call context is, for example, a work item selected in the workflow inbox of the Business Workplace. This work item can then contain the notification of absence xy as a reference object.

- The system establishes the object type for this reference object.

- The system displays all workflows that refer to this object type so you can make your selection. This uses the same function as Starting workflows (see above). If there is no appropriate workflow, you can create a new workflow ad hoc. For further information, refer to Defining Workflows Ad Hoc [Seite 1098].

- The system starts the selected workflow and passes the reference object to the task.

Activities

The function Start workflow can be accessed via Office → Start workflow or via Tools → Business Workflow → Development → Runtime tools → Start workflow.

The function Start workflows with object reference is available within the Business Workplace in the workflow inbox, the workflow outbox and the workflow resubmissions under Environment.
Advancing with Immediate Dialog

Use
The runtime system of SAP Business Workflow supports advancing through work items with immediate dialog as standard.

Features
Advancing with immediate dialog means that when the current agent of a work item in the workflow has executed this work item, the method of the next work item is presented to them directly in dialog provided they are one of the recipients of the next work item. The runtime system behaving in this way means that a user does not have to "detour" via the workflow inbox of the Business Workplace, which would first have to be refreshed, in order to reserve or execute the relevant work item.

A user decides in a user decision to revise the notification of absence rejected by their superior. Since the user is therefore also the recipient of the next activity "Revise notification of absence", the relevant method is executed for them immediately.

The runtime system behaving in this way also, however, means that a work item is no longer offered to all recipients for processing in the status ready, since the first agent in such a chain of work items automatically becomes the current agent of the next work items.

Starting workflows
If Advance with immediate dialog is activated, the method for the first work item appears immediately when a workflow is started in dialog. The initiator of the workflow must, of course, be one of the recipients of this work item.

Since advancing with immediate dialog can only be activated or deactivated for workflows, work items for single-step tasks are always started with immediate dialog.

For further information, refer to Starting Tasks [Seite 1448].

Conditions
The following conditions must be met:
- The task underlying the step references a synchronous method [Extern].
- The step does not have a requested start date in the future, but can be started immediately.

Technical details
At runtime, when a step has been completed (= work item has status completed), the system checks whether the next work item can be started immediately with dialog and whether the actual agent is also a recipient of the next work item.
If this is the case, the work item is reserved for the agent as soon as it is created and the relevant object method is executed.
This procedure continues until more than one work item or no work items are created that can be executed immediately by the actual agent. More than one can occur when forks are used, for example. A user can also leave the "chain" by canceling execution of the method.
Advancing with Immediate Dialog

**Activities**

You activate or deactivate *advance with immediate dialog* in the basic data of a workflow definition. This setting then refers to advancing for all steps in the workflows relating to this definition. The first time the Workflow Builder is called, *advance with immediate dialog* is always activated.

You can also activate or deactivate *advance with immediate dialog* within the definition of an activity in a workflow definition. This setting then refers only to advancing from the preceding step to this step.

If you have deactivated advancing for the workflow in the basic data, you cannot activate it at the step level.
Process Controller

Definition

Person within the system of workflow roles [Seite 94] who evaluates the workflow from a business perspective at runtime, and is responsible for suggesting improvements.

Use

This part of the documentation is to be used in conjunction with the user menu for the process controller. This user can access this user menu on the screen SAP Easy Access via \[\text{access symbol}\].

Assign the following roles to the user:

- SAP_BC_SRV_USER
- SAP_BC_BMT_WFM_CONTROLLER

Structure

The documentation for this role consists of:

- Evaluation of Workflows [Seite 885]

Here you will find vital documentation for process controllers, in particular the documentation on the Workflow Information System.

Integration

This role is part of the role-based procedure model containing the following other roles:
Evaluation of Workflows

Purpose
As business-oriented evaluation of workflows is particularly useful for the purposes of information and optimization. Questions such as the following are asked:
- Who did what and when?
- In what context and for what purpose was something done?
- What was done with respect to particular objects (for example notifications of absence) and how long did it take?

Prerequisites
You have one or more productive workflows in your R/3 System.

Process Flow

- **Workload Analysis [Seite 1485]**
  Using the workload analysis, you can establish the workload of individual employees, positions, jobs or organizational units.

- **Work Item Analysis [Seite 1484]**
  Using work item analysis, you can carry out statistical evaluations of work items. The work items to be analyzed can be limited according to time, type and task.
Result

Process controllers forward the analyses to management or make suggestions for improvement. This can create work for process consultants and workflow developers.
Work Item Analysis

Use

You can use work item analysis to carry out statistical evaluations on work items [Seite 1371]. The work items to be analyzed can be limited according to time, type and task.

Features

Work items per task

Number of work items of the specified type or for the specified tasks generated in the specified monitoring period. The list is sorted according to task.
Double-clicking on an entry in the list displays all the work items for the task in another list. From this list you can select to go to the work item display [Seite 1411] and to go to the workflow log [Seite 1420].

Work items with monitored deadlines

Number of work items of the specified type or specified tasks, which are subject to deadline monitoring. The work items are displayed in an overview tree sorted according to the type of deadline monitoring and task. Double-clicking on an entry displays the work item.

Work items by processing duration

Information on the processing duration of work items of the specified type or on the specified tasks whose processing ended in the monitoring period. The list is sorted according to task.
Provided there are appropriate work items, the current monitoring period is compared with a prior period of the same length. The variances and differences are shown.
For further information, refer to Technical Details on Calculation [Extern] and Threshold Value and Average Value [Extern].

Activities

To call work item analysis, choose Tools → Business Workflow → Development → Reporting → Work item analysis → ...
Workflow Relationships

Use
You can use this function to establish the work items in which a particular object is being processed. The following work item types are taken into account:

- Dialog work items (type W)
- Background work items (type B)
- Workflow work items (type F)

The list displayed is purely runtime information.

Prerequisites

Authorization
You require appropriate authorization for the functions shown. This is an authorization based on the authorization object S_PROGRAM for executing a program for authorization group 'SWI_OA'.

Features
You can call the function for displaying linked workflows from several starting points.

Calling via SAP Easy Access
Two functions are available via SAP Easy Access:

- Workflows for object
  This function displays all workflows linked to a particular object (for example a specific notification of absence). You must specify this object beforehand.

- Workflows for object type
  This function displays all workflows linked to a particular object type (for example FORMABSENCE). You must specify this object type beforehand.

  You can only specify objects whose object type definition supports the interface IFFIND.

Calling from the Business Workplace
In this case, you do not specify the object. The workflow relationships are displayed for the object processed in the selected work item.

Calling as a generic object service
In this case, you do not specify the object. The workflow relationships are displayed for the object being processed.

What is displayed?
The display of the workflow relationships has two parts. The upper part displays the workflows identified for the object or object type. The lower part displays data on the currently selected work item in the form of a simplified workflow log:

- Steps so far
Workflow Relationships

The workflow steps processed so far are listed under Step name. The steps are linked to the work item preview of the relevant work item. The current agents of the work item are listed under Agents. Click once to go into detailed display of the user data.

- Information objects addressed so far
  All the objects and attachments belonging to the workflow are displayed here. Click once to display the information objects.

Activities

<table>
<thead>
<tr>
<th>Calling...</th>
<th>Path</th>
</tr>
</thead>
<tbody>
<tr>
<td>... Via SAP Easy Access</td>
<td>Tools → Business Workflow → Development → Runtime Tools → Workflows For Object</td>
</tr>
<tr>
<td>... From the Business Workplace [Seite 1368]</td>
<td>Display workflow relationships</td>
</tr>
<tr>
<td>... As a generic object service</td>
<td>System → Workflow → Workflow overview</td>
</tr>
</tbody>
</table>
Workload Analysis

Use
Using the workload analysis, you can establish the workload of individual users, positions, jobs or organizational units. You can look into the past (“completed since...”) and into the future (“to be processed by...”) in order to establish workload.

Features

Workload for the past
The selection produces a list of work items completed by the members of the organizational entity before the specified date. A work item must meet the following criteria to be displayed in this list:
1. The work item must represent a single-step task that satisfies the selection criteria specified as a task filter.
2. The work item must be a dialog work item [Seite 1372].
3. The work item must have the status [Seite 1373] completed.
4. The actual agent [Extern] of the work item is a user who is assigned directly or indirectly to the organizational entity specified.

The result of the selection is displayed with the following information in the list Work items completed according to agent and task:
- Organizational unit of the current agent
- Agent
- Number of the single-step task represented by the work item
- Date of final completion
- Work item text
- Process duration

If you double-click on an entry, you go to the relevant work item display [Seite 1411].

Export as Microsoft Excel document
The list displayed can be exported as a Microsoft Excel document. To do this, choose the function Excel.

Formatting for statistical purposes
The information can be formatted for statistical purposes. To do this, position the cursor on one of the columns Org. unit, Agent, Task or Date, and select the function Statistics.

Depending on the column you selected, the following information is displayed:
- Organizational unit and number of work items completed by an employee from the organizational unit.
- Agent and number of work items completed by this agent.
- Task and number of work items for this task, which were completed.
- Date and number of work items completed on this date.
In addition, the following times are calculated:

- **10% threshold:**
  For 10% of the work items, the duration of processing was shorter than the time specified.

- **50% threshold:**
  For half of the work items, the duration of processing was shorter than the time specified.

- **90% threshold:**
  For 90% of the work items, the duration of processing was shorter than the time specified.

**Workload for the future**

The selection produces a list of work items that must be processed by the members of the organizational entity by the date entered.

A work item must meet the following criteria to be displayed in this list:

1. The work item must represent a single-step task that satisfies the selection criteria specified as a task filter.
2. The work item must be a dialog work item [Seite 1372].
3. The work item must have one of the following statuses [Seite 1373]: waiting, ready, in process, reserved or executed.
4. The actual agent [Extern] of the work item must be a user who is assigned directly or indirectly to the organizational entity specified.
   - This only applies to work items with the status waiting, in process, reserved or executed, since there can only be an actual agent in these cases.
5. There must be at least one user among the possible agents [Extern] of the work item, who is assigned directly or indirectly to the organizational entity specified.
   - (This only applies to work items with the status waiting or ready since these do not yet have a current agent.)
6. The latest end for processing the work item must be before the date entered.

The result of the selection is displayed in the list *Workload for ...*. The list of work items is grouped according to current agents and single-step tasks. At the end of the list, the work items and tasks for which no actual user exists are displayed under the header *Not reserved by an agent*.

**When no date is specified**, a user's workload corresponds to the contents of their worklist in the Business Workplace.

**Operations on the list**

You can go to the definition of a task by double-clicking.

You can display the work items for a task by double-clicking on the plus sign in front of the task.

You can go to the work item display [Seite 1411] by double-clicking on a work item.

**Activities**

To call workload analysis, choose *Tools → Business Workflow → Development → Reporting → Workload analysis.*
To determine the past workload, choose the option *Completed since* on the selection screen *Workload analysis*. For future workload choose *To be processed by*.

Use the F4 input help for all other specifications.
Business Workplace: Workflow Functions

Use
You use this part of the Business Workplace if you want to use the functions of SAP Business Workflow. The Business Workplace is the main interface between an end user and the workflow system. All dialog and missed deadline work items to which the user is assigned as a recipient are displayed in the user’s workflow inbox.

As a head of department, you are responsible for approving leave requests. The relevant approval process is implemented using a workflow in your enterprise.

The requests (in the form of work items) appear in your worklist (workflow inbox) and must be rejected or approved there.

The rejected or approved requests (executed work items) are not only returned to the applicants after processing, but are also put into your workflow outbox (under Work items executed by me). You can therefore check the requests you have processed.

Features

Workflow settings
You can configure the workflow functions in the Business Workplace using the personal workflow settings.

Business Workplace screen areas
The Business Workplace has three screen areas, which are used in the following manner by SAP Business Workflow:
The following workflow functions are available under the *Inbox* node, which is under the initial node *Workplace*:

- [Workflow](#)
  - Grouped according to task
  - Grouped according to content
  - Grouped according to content type
  - Grouped according to sort key
- Overdue entries
- Deadline messages
- Incorrect entries

For information on these functions, refer to [Workflow Inbox [Seite 1408]].

The following functions are available under the *Outbox* node, which is under the initial node *Workplace*:

- Started workflows
- Work items executed by me
- Forwarded work items

For information on these functions, refer to [Workflow Outbox [Seite 1441]].

The *Resubmissions* node is located under the initial node *Workplace* and contains the:

- [Workflow resubmissions [Seite 1443]](#)
Business Workplace: Workflow Functions

Worklist
The worklist [Extern] is displayed in the upper right corner of the Business Workplace screen. Depending on whether you are in the workflow inbox, the workflow outbox or the workflow resubmissions, you have various functions available to you, which are described at the respective locations.

Work item preview
In the lower right corner of the Business Workplace screen, a work item selected in the worklist is displayed in a preview [Seite 1445]. Not all the functions of the work item display or the workflow log are available.
A user exit [Extern] can be used to configure the work item preview to suit your individual requirements.

Support for context menus
All workflow functions can be called using the relevant context menu.

Workflow Toolbox
SAP Business Workflow's Workflow Toolbox [Seite 1446] enables the user to access workflow functions even during a workflow-driven application transaction.

E-mail notification for new work items
The report RSWUWFML can be used to inform an employee by mail that there is a new work item in their Business Workplace inbox.
This function is therefore beneficial to all employees who do not work with their Business Workplace on a daily basis.

Activities
To access the Business Workplace from the SAP Easy Access screen, choose one of the following options:

- 📑
- Menu → Business Workplace
Work Item

Definition
Object that represents a task or action in the workflow system at runtime.

Use
Work items are subdivided into a specific work item type according to their assignments. The internal processing procedures are controlled via this work item type. The work item type determines which statuses and transitions are valid.
Depending on the work item type, some of these work items are displayed in a user's work list. Other work items, on the other hand, are only used and processed internally.

Structure

Work item types displayed in the Business Workplace

<table>
<thead>
<tr>
<th>Work item type</th>
<th>Short text</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>W</td>
<td>Work item with dialog</td>
<td>Runtime representation of single-step tasks that require interaction with the user.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Runtime representation of user decisions</td>
</tr>
<tr>
<td>D</td>
<td>Missed deadline</td>
<td>Work item for notification of missed deadline</td>
</tr>
<tr>
<td>A</td>
<td>Work Queue</td>
<td>A work queue is a list of objects to be processed once and together in a limited time frame.</td>
</tr>
</tbody>
</table>

Other work item types

<table>
<thead>
<tr>
<th>Work item type</th>
<th>Short text</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>F</td>
<td>Workflow</td>
<td>Runtime representation of a multistep task</td>
</tr>
<tr>
<td>B</td>
<td>Batch item</td>
<td>Runtime presentation of a single-step task that runs in the background</td>
</tr>
<tr>
<td>E</td>
<td>Wait step work item</td>
<td>Runtime representation of a wait step in the workflow definition</td>
</tr>
<tr>
<td>C</td>
<td>Container anchor</td>
<td>This type of work item is required as a special development in the EDI environment. It does not normally appear in the workflow environment. Work items of this type should be regularly deleted or archived.</td>
</tr>
</tbody>
</table>

[Seite 1372]  
[Seite 1377]  
[Seite 1390]  
[Seite 1385]  
[Seite 1392]  

[Extern]  
[EDI environment]
Dialog Work Items (Type W)

Definition
Work item that represents a task at runtime that requires interaction with the user.

Since the *user decision* is also represented internally by a task, a dialog work item can also represent a user decision.

When a dialog work item is executed, the underlying object method of the task is called. The deadlines for executing dialog work items are monitored.

Use
A dialog work item is displayed with *ready* status in the workflow inbox of the Business Workplace. It is removed from the integrated inboxes of the other agents when the recipient reserves, executes, or processes this work item with other functions.

The database oriented approach used in SAP Business Workflow allows a work item to be *seen* by several recipients equally authorized in organizational terms in their inboxes and *executed* from there. However, only one recipient can actually *reserve* this work item for processing and *execute* it. The work item is then no longer available to any other recipients.

Integration
A task represented by a dialog work item can be
- a step in a workflow definition:
  
  In the workflow definition, reference is made to tasks in the *activity* and *user decision* steps.

- started as single steps via an event or in dialog:

  Tasks can be started as elementary activities directly in dialog or via a triggering event. These tasks are then also represented by a dialog work item in the workflow inbox.
# Status of a Dialog Work Item

The valid statuses for dialog work items (type W) are listed in the following table.

<table>
<thead>
<tr>
<th>Work item status</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>waiting</td>
<td>The work item has been scheduled for its <em>requested start</em>. A work item has this status if it already exists but the <em>requested start</em> specified in the workflow definition has not been reached yet. If it has been set to resubmission, work items in the <em>waiting</em> status are not displayed in the workflow inbox.</td>
</tr>
<tr>
<td>ready</td>
<td>The work item has been released for execution and appears in the workflow inbox of all recipients.</td>
</tr>
<tr>
<td>reserved</td>
<td>The work item has been received by one of its recipients with the result that its status has changed from <em>ready</em> to <em>reserved</em>. A work item in the <em>reserved</em> status is then displayed to this recipient only. It is no longer displayed in the workflow inboxes of the other recipients.</td>
</tr>
<tr>
<td>In process</td>
<td>The work item is currently being processed by a recipient or in a different mode. A work item also has this status if the work item is waiting for its terminating event or if the user cancelled the method. If the method was ended with a temporary exception for which no subsequent steps have been modeled, the point at which processing is completed cannot be detected by the workflow system in this status. As long as the status of the work item is set to <em>in process</em>, database changes have not been made.</td>
</tr>
<tr>
<td>Executed</td>
<td>The work item is awaiting explicit confirmation of its completion. The work item only has this status if it is necessary to confirm that it has been completed. A work item with <em>executed</em> status can be executed or forwarded several times until it is set to the status <em>done</em> in the Business Workplace. In this way, groupware components are realized in SAP Business Workflow.</td>
</tr>
<tr>
<td>completed</td>
<td>The execution of the work item is completed. The result of the task represented by the work item is correct, i.e. the result modeled in the workflow definition. Work items in the <em>completed</em> status are not displayed in the workflow inbox of the Business Workplace.</td>
</tr>
<tr>
<td>Logically deleted</td>
<td>Execution of the work item is no longer meaningful or required by the workflow logic. A work item changes to the <em>logically deleted</em> status in the following way: When the required number of processing paths has been executed in a fork, the work items in the other paths that have not yet reached the <em>completed</em> status are automatically set to the <em>logically deleted</em> status. The <em>intervention</em> by an administrator can only set a work item to the <em>logically deleted</em> status if it has not yet reached the <em>completed</em> status and is not part of a higher-level workflow.</td>
</tr>
</tbody>
</table>

---

April 2001 898
# Status of a Dialog Work Item

<table>
<thead>
<tr>
<th>Status</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Work items in the <em>logically deleted</em> status are not displayed in the workflow inbox of the Business Workplace.</td>
</tr>
<tr>
<td></td>
<td>A work item with the <em>logically deleted</em> status may have caused database changes or other actions (generate event, send notification). These changes are not canceled automatically (compensated).</td>
</tr>
<tr>
<td>error</td>
<td>Execution of the work item was terminated with an error.</td>
</tr>
</tbody>
</table>

In addition to the statuses given above, a work item can be **locked against execution**. This is possible in any status and involves an administrative function which is selected for a work item via the *Change* option.
Status Transitions of a Dialog Work Item

The diagram below shows the possible status transitions that a dialog work item (type W) can undergo:

The arrows are labeled with the functions used by a user to trigger the respective status transition.

**Comments**
For further information, refer to Status of a Dialog Work Item [Seite 1373].

**Transition from status waiting**
The transition from the status waiting to the status ready is performed automatically by the work item manager when the requested start is reached. A workflow system administrator can set a work item to the status ready manually.

**Transition from status ready**
From this status, the work item passes either to the status reserved or via the status in process to the status completed.

**Transition from status reserved**
A work item with the status reserved can be reset to the status ready.

**Transition to and from status in process**
A work item with the status in process can be reset to the status ready. A workflow system administrator can reset a work item manually. This function is available when changing the work item.

**Transition from status executed**
After confirmation of end of processing, the work item assumes the status completed.
Status Transitions of a Dialog Work Item

**Transition from status completed**
Work items with the status *completed* can no longer be set to another status even if a workflow system administrator intervenes.

**Transition from status error**
A workflow system administrator can intervene and set work items with errors to the status *in process* or the status *logically deleted* (possibly after eliminating the error).

**Transition from status logically deleted**
Work items with the status *logically deleted* can no longer be set to another status even if a workflow system administrator intervenes.
Missed Deadline Work Item (Type D)

Definition
Notification of a deadline recipient if the runtime system detects that the deadline for a certain work item has been exceeded.

Use
This work item informs its recipients that a deadline (start or end deadline) of a monitored work item has been exceeded. The recipients are informed by means of a missed deadline work item (type D) in the workflow inbox of the Business Workplace.
When it is executed, this work item displays information on the monitored (and now late) work item. The text for notifying the recipient is set by default.
When a deadline is monitored for an activity [Seite 1024] or user decision [Seite 1074], it is also possible to enter an individual text in the respective task definition.
Status of a Missed Deadline Work Item

The valid statuses for missed deadline work items (type D) are listed in the following table:

<table>
<thead>
<tr>
<th>Work item status</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>☐ Ready</td>
<td>The work item appears in the Business Workplace of the recipients.</td>
</tr>
<tr>
<td>☐ In process</td>
<td>One of the recipients has executed the work item</td>
</tr>
<tr>
<td>☐ Completed</td>
<td>The execution of the work item is completed.</td>
</tr>
<tr>
<td></td>
<td>Work items in the completed status are not displayed in the workflow inbox of the Business Workplace.</td>
</tr>
</tbody>
</table>
Status Transitions of a Missed Deadline Work Item

The following status transitions are possible for missed deadline work items (type D):

**Transition to and from status in process**

The status of a missed deadline work item changes from *ready* to *in process* when it is executed by a user. Execution of a missed deadline work item displays the most important information on the monitored work item. The missed deadline work item remains in the *in process* status until end of processing is confirmed explicitly. Until then, the work item can be executed or forwarded several times.

**Transition to completed status**

The status of the missed deadline work item changes to the *completed* status when completion of processing has been confirmed explicitly.
Work Queue Work Item (Type A)

Definition
Work item that represents a work queue. A work queue comprises a list of objects that are to be processed once together within a specified period. The work queue serves as a framework for the individual entries to be processed and manages the list of objects to be processed including their statuses and the tasks to be performed on them. Work queue work items are displayed in the Business Workplace. The work item status indicates the overall processing status of the work queue.

Use
Once you have created the work queue work item and you know its work item ID, you have the following options:

- Process the work queue within a workflow.
- Control the release, processing and status evaluation of the work queue with function modules.
- Control the release and processing of the work queue by processing the work queue work item directly.

Integration
To create the work queue work item from the list, you call the function module SWZ_AI_CREATE.
## Status of a Work Queue Work Item

A work queue work item can have the following statuses:

<table>
<thead>
<tr>
<th>Work item status</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>waiting</td>
<td>The work queue has been created but cannot be released yet.</td>
</tr>
</tbody>
</table>
| ready            | The work queue work item (type A) is displayed in the Business Workplace of the recipients passed in the table **AGENTS** of the function module **SWZ_AI_CREATE**.  
If the work queue is to be processed via a workflow and therefore not appear as a separate work item in any Business Workplace, a non-existent user must be specified in the table when the work queue is created. The work queue must be reserved for further processing. |
| reserved         | A person with release authorization has reserved the work queue. |
| In process       | Work queue processing has begun.  
– Dialog work items (type W) have been created for entries with dialog.  
– For entries without dialog marked accordingly, the methods have been executed directly. |
| completed        | All lines in the work queue have the status COMPLETED or CANCELLED and have reported back accordingly to the work queue.  
A work queue work item (type A) that assumes this status automatically creates the event **created**. |
| Logically deleted| Further processing of the work queue is invalid (and therefore no longer possible). |
| error            | At least one line of the work queue has the status **error**. |
Workflow Work Item (Type F)

Definition
Work item that represents a multistep task at runtime.

Use
For every multistep task started there is one type F work item. The workflow log and the workflow container can be accessed:
- For error diagnosis and error correction if no work items are displayed
- For information on steps of a workflow (including their current agents, notes, and ad hoc objects) already processed
- For modifying an ongoing workflow by changing the workflow container
- For reporting on completed processes
Type F work items are not displayed in the Business Workplace but can be found using the work item selection [Seite 1490].

Structure
A workflow consists of a sequence of work items that are executed by agents or the system. The work items represent the steps in the workflow definition that refer to a particular task. These are steps of the types activity [Seite 1024] or user decision [Seite 1074].

Integration

Workflow Execution

The Work Item Manager manages the processing of work items and monitors deadlines. To automate workflow processes, activities in the workflow can also refer to object methods which run in the
background. If this is the case, the work item manager initiates the calling of the background processes. The work items whose execution requires dialog can be accessed by the selected agents (determined from the organizational model and role resolution) from their worklists in order to select them for processing. This worklist is displayed and managed in the workflow inbox of the Business Workplace [Seite 1368].
Status of a Workflow Work Item

The valid statuses for workflow work items (type F) are listed in the following table:

<table>
<thead>
<tr>
<th>Work item status</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Waiting</strong></td>
<td>The (sub) workflow is scheduled but its requested start has not yet been reached.</td>
</tr>
<tr>
<td><strong>Ready</strong></td>
<td>Temporary interim status</td>
</tr>
<tr>
<td><strong>In process</strong></td>
<td>Execution of the (sub) workflow has begun.</td>
</tr>
<tr>
<td><strong>Completed</strong></td>
<td>The end of the (sub) workflow has been reached.</td>
</tr>
</tbody>
</table>
| **Logically deleted** | Execution of the (sub) workflow is no longer required or meaningful. A status of a workflow changes to *logically deleted* in the following way:  
  - Intervention by an administrator.  
    An administrator can only set a workflow item to the *logically deleted* status if it has not yet reached the *completed* status.  
  - Termination in parallel processing branches  
    When the required number of processing paths has been executed in a fork, the work items in the other paths that have not yet reached the *completed* status are set to the *logically deleted* status.  

  A workflow set to the *logically deleted* status is recursively scanned for dialog and workflow work items (type W or F) that do not yet have the status "completed". These work items are then also set to the status *logically deleted*.  

  A work item changed from the *executed* status to the *logically deleted* status may already have caused database changes or other actions (generate event, send notification). These changes are *not* canceled automatically (compensated), but are recorded in the log. |

| **Errors** | An error occurred in workflow control. |

Workflows and their statuses are *not* displayed in the Business Workplace. Information about the workflow can be accessed via the subordinate dialog work items.
Status Transitions of a Workflow Work Item

The diagram shows the possible status transitions.

Status and Status Transitions of Work Items (F)

- **waiting**
- **ready**
- **incorrect:**
- **in process**
- **completed**

The statuses of a workflow work item and the permissible status transitions always concern an entire workflow.

A step (activity) in the workflow definition can reference both a task and another workflow. A workflow can therefore also contain subordinate subworkflow items.

**Comments**

**Transition from status waiting**
The work item manager automatically carries out the transition from the *waiting* status to the *ready* status when the requested start date/time of the workflow has been reached. A workflow system administrator can set a work item to the status *ready* manually.

**Transition from status ready**
The *ready* status is a temporary interim status because it only exists until the first work item of the workflow has been created.

**Transition to and from status in process**
The status of the workflow changes to *in process* as soon as the first work item of this workflow has been created. The workflow remains in this status until the entire workflow definition has been processed.

**Transition to incorrect status**
An error occurs during workflow control or coordination.

Role resolution for determining an agent does not return a result which can be used.
A workflow system administrator can intervene and set workflows with errors to the status *in process* or the status *logically deleted* (possibly after eliminating the error). If a workflow is incorrect, the responsible workflow system administrator specified either globally in Customizing or in the basic data of each workflow definition is notified by mail.

This status does *not* mean that a dialog work item of this workflow has the *incorrect* status.

**Transition to completed status**

A workflow is set to the status *completed* when the last step of the relevant workflow has been completed.
Background Work Item (Type B)

Definition
Work item that represents a single-step task at runtime whose execution does not require a dialog and, therefore, can be controlled automatically by the system.

Integration
Type B work items are not displayed in the Business Workplace.
# Status of a Background Work Item

The valid statuses for background work items (type B) are listed in the following table.

<table>
<thead>
<tr>
<th>Work item status</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Waiting</td>
<td>The work item has been scheduled for its requested start. A work item has this status after it has been created until the requested start specified in the workflow definition is reached.</td>
</tr>
<tr>
<td>Ready</td>
<td>Temporary intermediate status of a background work item. The system calls the associated object method as soon as a background work item can be processed. The status of the background work item then changes to in process immediately.</td>
</tr>
<tr>
<td>In process</td>
<td>The work item is currently being processed. A work item also has this status</td>
</tr>
<tr>
<td></td>
<td>- if the method was left with a temporary exception. In this case, special Error Handling for Background Work Items with Temporary Errors [Seite 1388] is carried out.</td>
</tr>
<tr>
<td></td>
<td>- if the work item is waiting for its terminating event. The point at which processing is completed cannot be detected by the workflow system in this status.</td>
</tr>
<tr>
<td>Completed</td>
<td>The execution of the work item is completed. The result of the task represented by the work item is correct, i.e. the result modeled in the workflow definition.</td>
</tr>
<tr>
<td>Logically deleted</td>
<td>Execution of the work item with regard to the process logic is no longer meaningful or necessary for the process to continue. A work item changes to the logically deleted status in the following way:</td>
</tr>
<tr>
<td></td>
<td>- Termination in parallel processing branches</td>
</tr>
<tr>
<td></td>
<td>When the required number of processing paths has been executed in a fork, the work items in the other paths that have not yet reached the completed status are automatically set to the logically deleted status.</td>
</tr>
<tr>
<td></td>
<td>- Intervention by an administrator</td>
</tr>
<tr>
<td></td>
<td>The administrator can only set a work item to the logically deleted status if it has not yet reached the completed status and is not part of a higher-level workflow.</td>
</tr>
<tr>
<td></td>
<td>A work item with the logically deleted status may have caused database changes or other actions (generate event, send notification). These changes are not canceled automatically (compensated).</td>
</tr>
<tr>
<td>Error</td>
<td>Execution of the work item was terminated with an error.</td>
</tr>
</tbody>
</table>

In addition to the statuses given above, a work item can be locked against execution. This is possible in any status and involves an administrative function which is selected for a work item via the Change option.
Error Handling for Background Work Items

Use
Error handling for background work items must be carried out by the workflow system:

- because background work items (with errors) are not displayed in the Business Workplace and are therefore detected too late, if at all.
- because background work items that are temporarily incorrect, for which another execution attempt can (theoretically) be successful, cannot be restarted by a user.

(Temporarily incorrect work items are work items whose method was terminated with a temporary exception for which no subsequent step has been defined.)

Features
In Customizing for the workflow system, you can define how often the workflow system attempts to restart a temporarily incorrect work item. You can also define the interval between two repetition attempts and activate the monitoring function. Monitoring is also activated as part of automatic Customizing ([Extern](Tools → Business Workflow → Development → Utilities → Customizing)). You can also specify the maximum number of repetition attempts (“repetition counter”) separately for each individual background step in the respective workflow definition. This setting overrides the value in Customizing if it is greater than zero.

If an error (method exception) occurs when background work items are executed by the workflow system, the system responds as follows depending on the type of exception and error modeling settings:

<table>
<thead>
<tr>
<th>In the workflow definition...</th>
<th>The exception is defined for the object method as a temporary error</th>
<th>The step has status completed and the modeled subsequent step is executed.</th>
</tr>
</thead>
<tbody>
<tr>
<td>a subsequent step is modeled for the exception.</td>
<td>The step has status completed and the modeled subsequent step is executed. Background work items for which a repetition counter is explicitly modeled in the workflow definition are restarted first by the system. If all of the attempts are unsuccessful, the status of the work item changes to completed and the modeled subsequent step is executed.</td>
<td>The step has status completed and the modeled subsequent step is executed.</td>
</tr>
</tbody>
</table>
**Error Handling for Background Work Items**

| no subsequent step is modeled for the exception | The step is not yet completed. The respective work item retains the status *in process*. Background work items are restarted by the system. The number of repetition attempts is determined either by the repetition counter in the step definition or - if this is equal to zero - by the repetition counter set in Customizing. If all of the attempts are unsuccessful, the work item status changes to *incorrect*. | Workflow and work item assume the *error* status. |

**Processing Incorrect Work Items**

The workflow system sends a mail to the relevant system administrator for every background work item with the status *incorrect*.

**Processing Work Items That Have Been Started**

The workflow system determines all of the background work items that have been *in process* for longer than 30 minutes. An error message is then sent to the workflow system administrator for all of these background work items, since the system assumes that processing has been cancelled. However, this does not necessarily mean that an error has occurred.
Wait Step Work Item (Type E)

Definition
Work item that represents a wait step or a workflow at runtime, which is waiting for an event to occur.

Use
Type E work items are not displayed in the Business Workplace.
Work Item with Express Notification

Use
When the system creates a work item with priority 1, each of its recipients [Extern] receives an express notification (dialog box with appropriate text) on the screen. The recipient can call the Business Workplace directly from the express message. Excluded agents do not receive a message.

Constraints
The system does not create an express message
- if the user processes the work item immediately due to advance with immediate dialog [Seite 1453].
- if the work item represents a general task [Extern] that is not restricted to certain agents.
- if the work item was forwarded.

The system only creates express notifications when a work item is created. An express notification is not created for the new recipients of priority 1 work items that are forwarded.

If one of the selected agents processes the work item, the other agents still receive an express message.

Features
Express messages are only sent for dialog steps as soon as the system has created them with the ready status. If the work item is created first with the waiting status because its requested start has not been reached yet, the express message is not sent until the status changes from waiting to ready.

Activities
How is the priority set?
The priority of a work item can be determined for steps that require a dialog with the user. It is defined in the tab page Miscellaneous of the step definition.
Workflow Settings

Use
You use the workflow settings to maintain the special workflow functions in the Business Workplace.

Features
The following functions are available:

- Personal settings [Seite 1395]
- Display organizational assignment [Seite 1397]
- Refresh organizational environment [Seite 1397]
- Adopt substitution [Seite 1400]
- End substitution [Seite 1400]
- Maintain substitute [Seite 1400]
- Activate substitute [Seite 1400]
- Adopt view [Seite 1404]
- Exit view [Seite 1404]

Activities
You can access the workflow settings within the Business Workplace [Seite 1368] via Settings → Workflow settings.
## Personal Settings

### Use
The personal settings for workflow enable you to adapt the runtime system to suit your requirements.

### Features

#### Work item display

<table>
<thead>
<tr>
<th>Setting</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>User view with ActiveX (32-bit)</td>
<td>Both of these settings define the [work item display][Seite 1411] as the default. Note that the ActiveX variant is only available on 32-bit platforms.</td>
</tr>
<tr>
<td>User view without ActiveX</td>
<td>This setting defines the [technical work item display][Seite 1418] as the default.</td>
</tr>
<tr>
<td>Technical view</td>
<td>Both of these settings define the work item display as the default. Note that the ActiveX variant is only available on 32-bit platforms.</td>
</tr>
</tbody>
</table>

#### Workflow log

<table>
<thead>
<tr>
<th>Setting</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>User view with ActiveX (32-bit)</td>
<td>User view of the workflow log [Seite 1420] without any technical aspects. This view uses ActiveX controls and is therefore only available on 32-bit platforms.</td>
</tr>
<tr>
<td>User view without ActiveX</td>
<td>User view of the workflow log without any technical aspects. This view uses the hierarchical list display of the ABAP List Viewer [Extern].</td>
</tr>
<tr>
<td>Technical view</td>
<td>In addition to the main semantic information, the technical view of the workflow log also displays technical numbers and texts that may not be available in the logon language of the user. The technical view is intended primarily for system administrators.</td>
</tr>
</tbody>
</table>

#### Further settings

<table>
<thead>
<tr>
<th>Setting</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Display work item texts in logon language</td>
<td>All work item texts in the Business Workplace are always displayed in the user’s logon language. If the user’s logon language is different to the language in which the work item was created, additional database accesses are required that can delay display of the Business Workplace.</td>
</tr>
<tr>
<td>Enable forwarding of work items to several users</td>
<td>The work item can be forwarded to several agents. The term “agent” is used in a broad sense in this context. You specify the agent as an organizational object (organizational unit, job, position, user) when forwarding the work item. This organizational object can consist of several persons. For further information, refer to [Forwarding][Seite 1435].</td>
</tr>
<tr>
<td>(Double)-clicking on an object displays the object in the same window.</td>
<td>You can choose an object in the work item display, which is then displayed in the current...</td>
</tr>
</tbody>
</table>
### Personal Settings

| No tips & tricks in workplace | The Business Workplace [Seite 1368] includes tips and tricks for working with work items and workflows in the work item preview. If you set this indicator, the tips and tricks are not displayed. |
| No HTML in execution of decision tasks. | You use this indicator to decide whether the execution of decision tasks is displayed in HTML or not. Deactivating the HTML display may make sense, for example, if you have problems displaying HTML documents generally because of the settings in your local HTML browser. |

### Activities

To change the personal settings for workflow, you choose **Settings ‒ Workflow settings ‒ Personal settings** in the Business Workplace.

The settings are saved as user parameters and take effect the next time the work item display or workflow log is called.

You can change the standard values for these personal workflow settings in Customizing for SAP Business Workflow. These standard values are always used if users have not maintained settings of their own.
Organizational Assignment of a User

Use
The worklist displayed is formatted specifically for the user who is logged on. You can query this user's organizational assignment.

Prerequisites
The functionality described depends on a setting in the PD system table [Seite 1533].

Features
The organizational assignment of a user is buffered and read at each new logon or once a day (but not each time the Business Workplace is called).
If the organizational assignment of a user changes while the user is logged on and must be updated, you choose Settings → Workflow settings → Refresh organizational environment in the Business Workplace.

Activities
To view current information on the assignment of the user logged on in the organizational plan of the enterprise, choose Settings → Workflow settings → Display organizational assignment in the Business Workplace.

The following information is displayed for the user logged on:
- The name maintained in the user address
- The organizational unit
- The position the user occupies
- The job describing the position

By double-clicking on an entry, you can display the staff assignments or the job description.
Substitutes for Workflows

Use

*SAP Business Workflow* implements a substitution arrangement to deal with a user's duties in their absence.

Prerequisites

The automatic appearance of work items in the substitute's worklist is dependent on a setting in the PD system table. It is possible to define which objects can be entered as position-related substitutes in Administration of Organizational Management.

Features

Who is involved?

Substitution always involves two users:

- One user specifies a substitute: User B
  - The other user (the substitute) 'adopts' the substitution: User A

The substitute can process “foreign” work items for the duration of the substitution. It is irrelevant whether they are assigned to the underlying single-step tasks as a possible agent.

How does substitution work?

Substitution works in two ways:

9. User A adopts substitution for user B and for the duration of the substitution sees only the work items seen previously by B in their workflow inbox.

   User B must have entered user A as their substitute for a particular period without activating the substitution. During this period, A can adopt substitution for B at any time without having to confer with B.

   Instead of their own worklist, user A is displayed user B's worklist. B's name is in the column *Substitution for* in the worklist, as long as this column is included in the current configuration of B's Business Workplace.

10. User A automatically sees not only the work items previously seen by employee B, but also their own work items in their Business Workplace. This also applies to all work items generated for B in the future ("automatic forwarding").

   User B must enter user A as substitute and activate the substitution. No further action on the part of user A is required for this kind of substitution.

   A can continue working as usual. They see a Business Workplace to which B's work items are added dynamically. A can recognize these work items by the fact that B's name is in the column *Substitution for*, as long as this column is included in the current configuration.

In both cases, user A can perform operations on these "foreign" work items within the scope of the substitute profile assigned to them by B. It is irrelevant whether A is assigned to the underlying single-step tasks as a possible agent.

User B can continue working without any restrictions in both cases.
Activities
You can access substitute maintenance in the Business Workplace [Seite 1368] by choosing Settings → Workflow settings. For information on the individual functions, refer to Maintaining and Activating Substitutes [Seite 1400]
Maintaining and Activating Substitutes

Maintaining substitutes

5. In the Business Workplace [Seite 1368], choose Settings → Workflow settings → Maintain substitute.

The Maintain substitute dialog box appears.

To facilitate maintenance of the substitution, the Personal substitutes entry and the positions you occupy are displayed in the hierarchy on the Maintain substitutes screen.

You specify your substitute either as a personal substitute or as a position-related substitute.

- **Personal substitute**
  
  A personal substitute can see and execute all your work items, including those assigned to you via a personal agent assignment.
  
  You can only specify one other user as a personal substitute.

- **Position-related substitute**
  
  A position-related substitute can only see and execute the work items you have received on the basis of agent assignment at the level of position, job or organizational unit.

  You can enter another position or a user as a position-related substitute.

22. Position the cursor either on the entry Personal substitutes or on the relevant position, and select Create substitute.

You can maintain existing entries by double-clicking on the relevant entry.

If you want to maintain a position-related substitution, decide whether you want another position or another user as a substitute. Select either position or user as the substitute type and specify the number or the user name of the substitute.

23. Specify the validity period for the substitution on the detail screen for substitution.

Only within this period can the substitute adopt the substitution.

24. Specify a substitute profile in the dialog box Detail screen substitution.

Irrespective of whether you have created your substitute as a personal or position-related substitute, you can limit the scope of the work items displayed to your substitute by specifying a substitute profile.

25. Select the field Substitution active if applicable.

You must activate the substitution if you want work items to be visible for the substitute automatically from now on. If you do not activate the substitution, the substitute must adopt the substitution explicitly to see your work items.

26. Save your entries and exit substitute maintenance.
Activating substitutes

   The Activate substitute dialog box appears.
10. Select the substitutes that you want to activate, and choose the function Activate.

Deactivating substitutes

   The Activate substitute dialog box appears.
10. Select the substitutes that you want to deactivate, and choose the function Deactivate.

Adopting substitution

13. In the Business Workplace, choose Settings → Workflow settings → Adopt substitution.
    The Choose substitution dialog box appears.
14. Select the user(s) for which you want to adopt substitution.
15. Exit the dialog box.

Ending substitution

5. In the Business Workplace, choose Settings → Workflow settings → End substitution.
    The substitution is ended.
Views

Use
The Business Workplace [Seite 1368] provides various views on the work items displayed in the workflow inbox.

Using an appropriate view, a superior can "see" and process the work items of their employees.

Choosing a particular view also gives you the opportunity to see work items of other users in your workflow inbox and process them with full functionality, although the underlying tasks are not organizationally assigned to you.

Prerequisites
Views only require action by an employee: This employee chooses a view from a catalog of defined views. The employee must have the authorization required to choose a particular view.

Authorizations
To choose a particular view, you require a corresponding authorization based on authorization object S_WF_LVIEW.
To maintain a particular view, you must have the relevant authorization. This is an authorization based on the authorization object S_TABU_DIS to maintain tables for authorization group SWES.

Features
Views are always based on an evaluation path [Seite 1534] starting from the employee who wishes to adopt a view and leading to the employees whose inboxes could be viewed. The employee adopts a view by selecting another employee from the result list of the evaluation.

Activities
The activities associated with this function include:

- Adopting and Exiting Views [Seite 1404]
- Maintaining Views [Seite 1403]
Maintaining Views

Prerequisites

A view [Seite 1402] is always based on an evaluation path [Seite 1534]. This evaluation path describes which relationships are traced from the user who wants to adopt the view to the users whose Business Workplaces can be viewed.

You can use one of the evaluation paths available in the system. If there are no suitable evaluation paths, you can define an evaluation path [Seite 1535] of your own.

⚠️

Different views can only be maintained as a customer setting.

Procedure

29. To call table maintenance for views, choose Tools → Business Workflow → Development → Definition tools → Worklist client → Maintain views.

This displays the screen Change View "View for Maintaining Views": Overview".

Maintaining views is an activity that is described in the Implementation Guide and can be performed in Customizing.

30. Create a new view. To do this, choose the function New entries.

31. Assign a unique name to the view.

32. Specify an evaluation path.

33. Describe the view with a long text.

34. Specify a start evaluation path.

This selection is optional. The start evaluation path is used to get an initial selection of objects, which is then evaluated further via the first evaluation path.

35. Select Save.
Adopting and Exiting Views

Procedure

Adopting views

13. To adopt a view [Seite 1402] as the standard view (= view on your own work items), choose Settings → Workflow settings → Adopt view in the Business Workplace.

   You are now on the dialog box View: Choose Agent. Only the views for which you have an authorization are available.

14. Choose a view.

   On the basis of the evaluation path defined for the view, the system selects the positions, organizational units or users connected to you.

15. From the result list of this selection, choose the object whose workflow inbox you want to view.

Exiting views

A view is only active while the Business Workplace is displayed. The next time you call the Business Workplace, you are asked if you wish to adopt the view previously set.

To return to the standard view when working in the workflow inbox of the Business Workplace, choose Settings → Workflow settings → Exit view.
Dynamic Columns for the Business Workplace

Use
Up to 6 columns in the workflow inbox of the Business Workplace [Seite 1368] can be filled on a task-specific basis with contents that are determined dynamically at runtime.
The standard functions for filtering, sorting, and grouping are available for these columns.
Please compare with Selectable Columns for the Business Workplace [Seite 1406].

Features
If you want to include one of the “dynamic columns” into the workflow inbox display, you must specify an element from the task container for each task, from which the content of the column is established at runtime.

Work items that belong to different tasks are then also displayed with different information. Work items that belong to tasks for which this functionality is not used are displayed with a blank entry.

Activities
The column contents are maintained via Tools → Business Workflow → Development → Definition tools → Worklist client → Dynamic columns for worklist.

You can also define the column headings. These headings are displayed if all of the work items displayed in the Business Workplace refer to the same task.
Selectable Columns for the Business Workplace: Workflow

The columns displayed essentially determine the appearance and information content of the workflow inbox of the Business Workplace. Detailed knowledge of the columns is also required to make full use of the filter and grouping criteria.

You can determine the selection of columns via display variants. The following columns are available:

<table>
<thead>
<tr>
<th>Column</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>Work item is executable</td>
<td>Indicator denoting that the work item is executable. ( swore )</td>
</tr>
<tr>
<td>Work item title</td>
<td>Title of work item.</td>
</tr>
<tr>
<td>Status indicator</td>
<td>Status of work item. In symbol form.</td>
</tr>
<tr>
<td>Creation date</td>
<td>Date when the work item was created with the status ready or waiting for the first time. A work item is only created with status waiting if a requested start was declared for the work item and the work item is created before the requested start.</td>
</tr>
<tr>
<td>Creation time</td>
<td>Creation time of a work item.</td>
</tr>
<tr>
<td>Priority</td>
<td>Priority of the work item.</td>
</tr>
<tr>
<td>Attachments exist</td>
<td>Attachments are shown with the symbol.</td>
</tr>
<tr>
<td>End of processing must be confirmed</td>
<td>Indicator denoting that the end of processing must be confirmed explicitly. ( swore )</td>
</tr>
<tr>
<td>Work item overdue</td>
<td>Indicator denoting that a deadline has been missed for the work item. ( swore )</td>
</tr>
<tr>
<td>ID</td>
<td>Unique number of a work item, which is assigned internally by the system.</td>
</tr>
<tr>
<td>Type</td>
<td>Type of work item. In abbreviation form.</td>
</tr>
<tr>
<td>Task</td>
<td>Identification for the task represented by the work item (for example TS00008323).</td>
</tr>
<tr>
<td>Technical status</td>
<td>Status of work item. Technical name.</td>
</tr>
<tr>
<td>Work item type</td>
<td>Type of work item. Name</td>
</tr>
<tr>
<td>Task name</td>
<td>Name of task</td>
</tr>
<tr>
<td>Work item status text</td>
<td>Status of work item. Name</td>
</tr>
<tr>
<td>Deadline status</td>
<td>The deadline status specifies whether one of the deadlines has been missed. The possible values in this column are therefore:</td>
</tr>
<tr>
<td></td>
<td>• None</td>
</tr>
<tr>
<td></td>
<td>• Latest start</td>
</tr>
<tr>
<td></td>
<td>• Requested end</td>
</tr>
<tr>
<td></td>
<td>• Latest end</td>
</tr>
<tr>
<td></td>
<td>• Other</td>
</tr>
<tr>
<td>Current agent</td>
<td>Name of the user who last reserved or processed the work item.</td>
</tr>
</tbody>
</table>
Selective Columns for the Business Workplace: Workflow

<table>
<thead>
<tr>
<th>Column</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Latest end date</strong></td>
<td>Latest end of work item. The end is reached when the work item assumes the status <em>completed</em>.</td>
</tr>
<tr>
<td><strong>Latest end time</strong></td>
<td>Latest end time of a work item.</td>
</tr>
<tr>
<td><strong>Forwarder</strong></td>
<td>Name of the party who forwarded the work item.</td>
</tr>
<tr>
<td><strong>Substitution for</strong></td>
<td>Name of the substituted party whose work item is displayed</td>
</tr>
<tr>
<td><strong>Work item content</strong></td>
<td>Column in which the default attribute of the object referenced in the container element _WI_Object_ID is displayed.</td>
</tr>
<tr>
<td><strong>Group object</strong></td>
<td>Column in which the default attribute of the object referenced in the container element _WI_Group_ID is displayed.</td>
</tr>
<tr>
<td><strong>Execution can be rejected</strong></td>
<td>Indicator denoting whether execution of the work item can be rejected (X).</td>
</tr>
<tr>
<td><strong>Dynamic columns</strong></td>
<td>Refer to [Dynamic Columns for the Business Workplace][1405].</td>
</tr>
</tbody>
</table>

[1405]: Dynamic Columns for the Business Workplace [Seite 1405]
Workflow Inbox

Use
The worklist of the user currently logged on to the Business Workplace is displayed in the workflow inbox.

Integration
As is the case for the workflow resubmissions [Seite 1443] and the workflow outbox [Seite 1441], the workflow inbox is an integral part of the Business Workplace.

Features
Views in the workflow inbox
A user's worklist can be displayed as an overview or according to the following grouping criteria:

<table>
<thead>
<tr>
<th>Grouping</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Grouped according to task</td>
<td>The work items are grouped according to the tasks to which they belong.</td>
</tr>
<tr>
<td>Grouped according to content</td>
<td>The work items are grouped according to the object instances to which they belong.</td>
</tr>
<tr>
<td>Grouped according to content type</td>
<td>The work items are grouped according to the object types to which they belong.</td>
</tr>
<tr>
<td>Grouped according to sort key</td>
<td>The work items are grouped according to sort keys. Please refer to Grouping According to Sort Keys [Seite 1432].</td>
</tr>
</tbody>
</table>

You can also choose from the following views:
- Overdue entries
- Deadline messages
- Incorrect entries

Workflow Functions
The screen area in which the worklist is displayed has a toolbar containing the workflow functions and functions for configuring the list. For information about these functions, refer to the documentation on the ABAP List Viewer [Extern].

The workflow functions can be accessed in the Business Workplace via the toolbar or the relevant context menu (right-hand mouse button). The following functions can be executed on work items:

- **Update**
  This function updates the worklist [Extern] of the user currently logged on to the Business Workplace.

- **Execute**
  To be executed, a work item must have either the status ready or the status in process.
  When a dialog work item [Extern] is executed, the object method to which the single-step task for this work item refers is carried out.
  When a missed deadline work item [Extern] is executed, the most important information on the monitored work item is displayed.

- **Display work item**
  This function goes directly to the work item display [Seite 1411]
- **Reserve** (dialog work items only)
  This reserves a work item for execution by the end user in question. The work item must have the status `ready`. This work item is then no longer visible to the other recipients who could previously see it in status `ready`. The status of the work item changes from `ready` to `reserved`.

- **Replace** (dialog work items only)
  Replacing releases the work item from its reserved status and makes it available again to all of the original recipients. The status of the work item changes from `reserved` back to `ready`. The status of the work item must be `reserved` before it can be replaced.

- **Forward**
  Forwarding [Seite 1435] passes a work item on to another employee for execution.

- **Resubmit**
  If a user chooses this function, the selected work item is placed in workflow resubmissions [Seite 1443].

- **Display workflow log**
  This function displays the workflow log [Seite 1420].

- **Manage attachments** ... (Functions for attachment management [Seite 1429], dialog work items only)
  - Display attachments
  - Create attachments
  - Change attachments
  - Delete attachments

- **More functions**
  - Set to 'Done' [Seite 1440]
  - Reject execution [Seite 1430]
  - Execute together [Seite 1431]
  - Change priority [Seite 1433]
  - Send mail [Seite 1425]
  - Change work item [Seite 1438]

- **Environment**
  - Display objects in workflow (dialog work items only)
  - Display workflow relationships [Seite 1436]
  - Start Workflow [Seite 1450]
  - Create link [Seite 1434]
Activities
You can access the workflow inbox in the tree on the left in the Business Workplace via Workplace → Inbox.
Work Item Display: Standard View

Use
The objective of the work item display is to display all the information and activities that are relevant to an end user in the environment of the displayed work item in a clear and concise manner. The work item display supports the agent not only in the processing of the current work item but also in the compilation of an activity list, which then functions as the agent's personal worklist.

Integration
A technical work item display [Seite 1418] is available for work items of all other types. You can make this technical work item display standard for dialog work items as well. The standard view of the work item display can be shown with or without ActiveX. You make this setting for the work item display variant in your personal workflow settings [Seite 1395].

Features
The work item display is primarily designed for displaying dialog work items [Extern]. It contains details about deadlines, statuses, agents, attachments and linked objects for a work item. The work item display also enables an end user - providing they have the relevant authorization - to compile an activity list as their personal worklist. The work item display has three tab pages (Basic data, Activities, and Available objects).

- **Tab page Basic data [Seite 1414]**
- **Tab page Activities [Seite 1415]**
- **Tab page Available objects [Seite 1416]**
- **Customer-defined tab page for work item display [Extern]**

You can define another tab page, which is then displayed as the first tab page when the work item display is called.

Application toolbar functions

- **Execute**

  To be executed, a work item must have either the status **ready** or the status **in process**.

  When a dialog work item is executed, the object method to which the single-step task for this work item refers is carried out.

  When missed deadline work items [Extern] are executed, the most important information on the monitored work item is displayed.

- **Display last message**

  The return code that was returned to the workflow system after the object method was executed can be retrieved for processed work items using the **Messages** function.

- **Forward**

  Forwarding [Seite 1435] passes a work item on to another employee for execution.

- **Resubmit**
If a user chooses this function, the selected work item is placed in workflow resubmissions.

- **Change priority**
  Refer to Changing Priorities.

- **Change deadlines**
  Refer to Changing Deadlines.

- **Display/create/change attachments**
  For attachments, refer to Attachment Management.

- **Reserve** (dialog work items only)
  This reserves a work item for execution by the end user in question. The work item must have the status ready. This work item is then no longer visible to the other recipients who could previously see it in status ready. The status of the work item changes from ready to reserved.

- **Replace** (dialog work items only)
  Replacing releases the work item from its reserved status and makes it available again to all of the original recipients. The status of the work item changes from reserved back to ready. The status of the work item must be reserved before it can be replaced.

- **Mail**
  Refer to Send Mail.

- **Display workflow log**
  This function displays the workflow log.

- **Display graphical workflow log**
  This function displays the graphical workflow log.

**Additional functions in the menus**

**Work Item**

- **End resubmission**
  The work item is put back into the workflow inbox. It has the status reserved.

- **Create link**
- **Reject execution**
- **Set to ‘Done’**

**Goto**

- **Methods**
  You can use this function to execute the defined secondary methods of the work item displayed from the work item display.

  A prerequisite for this is that the work item represents an activity that is part of a workflow. At least one secondary method must be defined for this activity.
Work Item Display: Standard View

- **Workflow description**
  - The description text of the superordinate multistep task, the "process description", is displayed.
  - The work item must be part of a workflow and a description text must be maintained for the workflow.

- **Technical work item display** [Page 1418]

**Extras**

- **Displaying tasks**
  - This function can be used to display the definition of the task represented by the work item. Refer to Definition of a Single-Step Task [Page 1175] and Definition of a Multistep Task [Page 1194].

- **Technical data**
  - Technical data about the work item, for example its ID, its texts or the ID of the superordinate work items are displayed.

  - **Organizational Assignment** [Page 1397]
  - **Displaying Agents** [Page 1427]

**Activities**

You can access the work item display by:

- Selecting a work item in the Business Workplace and choosing 🏝.
- Double-clicking one of the entries for a step in the workflow log.
- Selecting an entry that does not represent a workflow in the work item selection [Page 1490] hit list displayed. (If you choose a workflow, the workflow log is displayed.)
Tab Page Basic Data

Use
The information on this tab page of the work item display [Seite 1411] is mostly self-explanatory. Except for the priority, you cannot make any changes here.

Features

Deadlines
These are the deadlines monitored by the runtime system. (Deadlines that are not set are displayed without a date.)
- Start by (latest start)
- End by (latest end)
Depending on whether the work item represents a step in the workflow or a task, the deadlines were either specified when this step was defined in the workflow definition or when the task was started online.
To display all of the deadlines of the work item, choose Work item → Deadlines.

Further Information
- Forwarded by
  If the work item was forwarded to you, the name of the forwarder is entered here.
- Priority
  The priority of a work item is derived from the definition of this step in the workflow definition. The priority is used as a sort criterion for positioning the work item in the Business Workplace.
  The priority can be changed here.
- Status
  The current processing status is expressed by the work item status [Seite 1373].
- Creation date (created on) and processing date (processed from)
  These are the actual dates and times (when the work item was created and when processing was started).
Using the Messages function, you can display the return codes for processed work items, which were returned to the workflow system after execution of the object method.

Work Item Description
A description of the work item to be executed is provided at the bottom left of this tab page.
The task description is entered in the task definition. It is used for information purposes and generally contains instructions and recommendations on processing the work item displayed.

Attachments
The titles of all the attachments added to this work item or, if the work item is part of a workflow, to the preceding work items are shown in the lower right of this tab page.
Tab Page Activities

Use
The *Activities (not yet processed)* list contains all the activities that are relevant for processing this work item.
The tab page *Activities* is part of the [work item display][Seite 1411].

Features
The work item text of the task represented by the work item is generally at the start of the list ("main activity"). Once this activity has been processed (and completion of processing has been confirmed, if necessary), the status of the work item changes to *completed*. No other actions can then be carried out in the work item display.

Activities
You can extend the activity list and in this way create a worklist. The activities added represent your "personal worklist" as end user (agent).

<table>
<thead>
<tr>
<th>Function</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>✅ <em>Done</em></td>
<td>An activity selected from the list is reported to be <em>done</em>. This functionality cannot be executed on the leading activity (first line) of this work item. The agent is responsible for reporting that an activity is <em>done</em>. The actual processing is not checked.</td>
</tr>
<tr>
<td>😊 <em>Execute</em></td>
<td>An activity selected from the list is executed. The main activity (first line) of the work item can also be executed via the menu path <em>Work item → Execute</em>.</td>
</tr>
</tbody>
</table>
| 🗿 *Create* | Another activity is added to the list. This activity can entail:  
  * Executing a method on an existing object (created invoice, created material, etc.)  
  * Creating a new object (text, etc.)  
When you create an activity, you are given step-by-step support in dialog by a "wizard". |
| 💰 *Delete* | An activity is deleted from the list. |

Scrolling in the activity list: The activity list contains extensive information on each activity, which you can view by scrolling to the right. This information includes the following:

- Description
- Creator of the activity with date and time
- Actual agent of the activity with date and time
- Method and object type
Tab Page Available Objects

Use
All objects that are related to the work item are displayed in the list of available objects. These objects are stored in container elements of the task container as object references. You can display these objects or add new objects to the container element. There are:

- **Ad hoc objects**
  
  Objects added to a work item in this or one of the preceding steps of the workflow (container element `_Adhoc_Objects` of the task container).

- **Attachment objects**
  
  Documents added to a work item in this or one of the preceding steps of the workflow (container element `_Attach_Objects` of the task container).

- **Process objects**
  
  - The object currently being processed (container element `_WI_Object_ID` of the task container)
  
  - The object added for grouping purposes (container element `_WI_Group_ID` of the task container)
  
  - Objects that are referenced in other elements of the task container

The tab page *Available objects* is part of the work item display [Seite 1411].

Features

**Displaying objects with their default attributes**
Each object referenced in the work item container is displayed with its default attribute [Extern]. The default method [Extern] of each object can be executed upon request. If no default attribute was defined for the object type, the key fields of the object are displayed.

**Adding objects**
You can extend and process the list of objects. The main purpose of this is to make the relevant information available to the agents of the subsequent steps in the workflow as well. Only object types [Extern] that support the `IFFIND` interface can be selected. You identify an actual object [Extern] of this type by specifying its key fields [Extern].

Activities
To execute the functions displayed, proceed as follows:

<table>
<thead>
<tr>
<th>Function</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><img src="create.png" alt="Create" /></td>
<td>Create: An object is added to the list. When you add an object, you are given step-by-step support in dialog by a &quot;wizard&quot;.</td>
</tr>
<tr>
<td><img src="display.png" alt="Display" /></td>
<td>Display: The default method of an object selected from the list is executed.</td>
</tr>
<tr>
<td><img src="remove.png" alt="Remove" /></td>
<td>Remove: An object is deleted from the list.</td>
</tr>
</tbody>
</table>
Work Item Display: Technical View

Use
The technical work item display shows all information from the environment of the work item displayed. The technical work item display is intended in particular for workflow system administrators.

The data is always formatted and displayed in a similar way irrespective of the work item type. You should note the work item type shown on the screen.

Features

General

Work item information
- Information derived from the definition of the relevant task: Work item type, work item text, way in which processing is completed.
- Status of the work item as current processing information
- Actual agent (available after dialog work items have been executed).

In the case of processed work items, the Messages function can be used to display the return codes returned to the workflow system after execution of the object method.

Deadline data
Here you can find the current dates/times [Extern] (creation date/time of work item and date/time processing started and ended) as well as the deadlines that are monitored by the runtime system (requested and latest start and end deadlines).

A monitored deadline shown with red is in the past. If the symbol is displayed as well, a defined action was triggered.

Description
This description displayed here is taken from the task definition. It is used to inform the selected agents and contains instructions and recommendations on processing the work item displayed.

If you have selected a work item using the work item selection [Seite 1490] or the workflow outbox rather than your workflow inbox, you can execute it provided you are one of the possible agents.

You do not need to be one of the recipients in this case.

Additional functions
As well as the functions available in the standard view of the work item display [Seite 1411], the following additional functions are also available:

- Display/create/delete object

Each object referenced in the work item container is displayed with its default attribute [Extern]. The default method [Extern] of each object can be executed upon request. If no default attribute is defined for the object type, the key fields of the object are displayed instead.
You can extend and process the list of objects. The main purpose of this is to make the relevant information available to the agents of the subsequent steps in the workflow as well.

Only object types [Extern] that support the IFFIND interface can be selected. You identify an actual object [Extern] of this type by specifying its key fields [Extern].

- **Type-specific data**
  Type-specific data only applies to work queue and wait step work items.
  
  For work queue work items (type A), the objects and tasks contained in the work queue are listed.
  
  For wait step work items (type E), the number of events expected (information taken from the workflow definition) and the number of events that have already occurred are specified.

- **Execute for testing**
  In order to make it possible to check the runtime system's program execution (work item manager and workflow manager) in debugging mode after execution of an object method, internal communication within the workflow system cannot take place asynchronously. To achieve this, execute the work item for test purposes. Enter /h in the command field first to go to the debugging mode.

- **Monitored work item**
  The work item (type F or W) whose deadlines or completion are monitored by the workflow system is displayed.
  
  This function is only possible and active from missed deadline work items [Extern]. This function displays the work item. Full display functionality, including all navigation and change options, is also available here for this work item.

- **Instance linkage**
  You go to the relevant line of the instance linkage table, in which the expected event (identified by object type and event ID) and the object (identified by object reference) are specified.
  
  This function is only possible and active for those work items that wait for an event that completes them. This may apply to dialog work items (type W) and background work items (type B). This does apply to wait step work items (type E).

- **Container**
  This function displays the content of the task container.

- **Change work item [Seite 1438]**

**Activities**

You can go to the technical view of the work item display by choosing Goto → Technical work item display in the work item display or by having this display variant as a presetting in your personal workflow settings [Seite 1395].
Workflow Log: Standard View

Use
The workflow log [Extern] enables you to format all the information that is generated or collected during the execution of one active business process in one central location. The standard view described here is intended, in particular, for agents and process controllers who want to get an overview of the steps processed so far.

Prerequisites
To use the view described here, you must have set a view with ActiveX in your personal workflow settings [Seite 1395].

Integration
Other options for displaying the workflow log are:
- Workflow Log: Technical View [Seite 1422]
- Graphical Workflow Log [Seite 1424]

Features

Application toolbar functions
The following functions are available on the application toolbar:
- Update
- List with technical details
  This function takes you to the technical view of the workflow log.
- Graphical workflow log
- Personal workflow settings
  You can use this function to maintain your personal workflow settings.

Tab pages
The system shows the various views on the workflow log on different tab pages.

Tab page Workflow Chronicle (𝑖, what was processed when?)
The tab page Workflow Chronicle shows a hierarchical display of all steps in the workflow, which have been processed so far or are due to be processed. If the workflow has a subworkflow structure, the subworkflows are also displayed.
The Details function (𝑖 symbol) lists the following information about each step in the lower part of the screen:
- Who carried out what detailed actions for these work items and with what results.
- When these actions were carried out.
- Which objects were involved.
The Agents function (𝑖 symbol) displays the selected and possible agents of a step.
The Graphic function (𝑖 symbol) displays the graphical workflow log.
Tab page Workflow Agents ( flowed, who processed what?)
The tab page Workflow Agents shows the employees involved in this workflow up to now. The following is displayed for each employee:

- What action was carried out in what step.
- When this action was carried out.
- Which objects were involved.

This view shows how an employee was involved in the execution of a workflow.

Tab page Workflow Objects ( flowed, what was processed?)
The tab page Workflow Objects lists the objects related to the workflow or addressed up to now in the execution of the workflow. These objects include:

- The “leading” object of the workflow.
- Any attachments and objects added in the individual steps of the workflow.

The following is displayed for each object:

- Who carried out what detailed action for what task.
- When this action was carried out.

This view shows what information was generated and processed, and how.

Information at the click of a mouse
You can view all the information provided in the workflow log using the mouse.
You can also go to the work item display [Seite 1411] for each dialog step. You can display address data for agents as well as the contents of work item attachments or the result of actions that have been executed.

Activities
You can access the workflow log from the work item display or the Business Workplace [Seite 1368] via the icon.
Workflow Log: Technical View

Use
The workflow log [Extern] enables you to format all the information that is generated or collected during the execution of one active business process in one central location.
The view described here, in particular the technical view described below, is intended for workflow system administrators.

Integration
Other options for displaying the workflow log are:
- Workflow Log: Standard View [Seite 1420]
- Graphical Workflow Log [Seite 1424]

Prerequisites
To use the view described here, you must have set the view without ActiveX in your personal workflow settings [Seite 1395].
If you have chosen technical view in your settings, the standard view with technical details is displayed (see below). Otherwise, the two workflow log views are identical.

Features
The system uses a two-level, hierarchical list from the ABAP List Viewer [Extern] to display the various workflow log views. You can adapt the appearance of the list to suit your requirements using display variants.
By single-clicking on an entry or a symbol, you can then branch to the workflow container or the work item display [Seite 1411], for example.
The following views are available:

- **Workflow chronicle**
  The first level of the Workflow chronicle view shows all the workflow steps that have already been processed or are currently pending. If the workflow has a subworkflow structure, the subworkflows are also displayed.
  The second level (detail view) shows the following for each step:
  - Who carried out what detailed actions for these work items and with what results.
  - When these actions were carried out.
  - Which objects were involved.
  This view is used to determine what activities were carried out in a workflow and in what order.

- **Workflow agents**
  The first level of the Workflow agents view shows the employees involved in this workflow up to now.
  The second level (detail view) shows the following for each employee:
  - What action was carried out in what step.
  - When this action was carried out.
Workflow Log: Technical View

- Which objects were involved.
  This view shows how an employee was involved in the execution of a workflow.

- Workflow objects
  The first level of the Workflow objects view lists the objects that are related to the workflow or that have been addressed up to now during execution of the workflow. These objects include:
  - The “leading” object of the workflow.
  - Attachments and objects added in the individual workflow steps.
  The second level shows the following for each object:
  - Who carried out what detailed action for what task.
  - When this action was carried out.
  This view shows what information was generated and processed, and how.

- With technical details (technical view)
  The technical view shows technical control information for execution of a workflow, as required by workflow administrators, for instance.
  Based on the workflow chronicle, the technical view shows technical nodes and control structures, and makes additional data available, such as container elements ( ), employee data ( ), and workflow data ( ).
  The status of the work item is also displayed.

- With subworkflow structure
  Here you can choose whether or not to display any subworkflows and their structure.

- With error indicators
  If you activate the function View with error indicators, errors are marked in the log with the symbol. The standard indicator is .

Activities
You can access the workflow log from the workflow inbox of the Business Workplace via the symbol, or using the context menu (click the right-hand mouse button).
You can maintain the individual views (such as the technical view) within the workflow log via Views or Other views.
Graphical Workflow Log

Use
The workflow log [Extern] enables you to format all the information that is generated or collected during the execution of one active business process in one central location.

Integration
Other options for displaying the workflow log are:
- Workflow Log: Technical View [Seite 1422]
- Workflow Log: Standard View [Seite 1420]

Features
The graphical workflow log adds to the textual information. The workflow steps already processed are shown with 🔄 in a graphical representation of the workflow definition. You can see at a glance which “route” a workflow instance has taken and which activities are processed in parallel to your own within a business process. Unlike the text version of the workflow log, the graphical workflow log also shows the subsequent flow of a workflow instance.
The screen of the graphical workflow log is split into the workflow area (left) and the overview area (right).
The following functions are available:
- 🔄 Refresh
  Refreshes the display.
- 🔄 Display node
  If you select an executed step and choose this function, the system goes to the technical work item display.
  If the step has not yet been executed, the relevant activity is displayed.
- 🔄 Align...
  The workflow is centered within the workflow area.
- 🔄 Zoom in
  The size of the workflow is increased within the workflow area.
- 🔄 Zoom out
  The size of the workflow is decreased within the workflow area.
- 🔄 Complete view
  The whole of the workflow is shown within the workflow area.
For other functions, refer to the documentation on the Workflow Builder [Seite 1002].

Activities
You can call the graphical workflow log from within the workflow log [Seite 1420].
Send Mail

Purpose
You want to send a mail to another user in connection with a work item that requires processing. The work item being processed is therefore also to be made accessible to this user for information purposes. The Send mail function enables you to send mails to any recipients with a text which can be entered freely. These mails are automatically linked to your work item in such a way that when the mail is executed by the recipient the work item is displayed.

Process Flow

Procedure as sender of mail
13. In the Business Workplace’s workflow inbox, position the cursor on a work item and choose Other functions → Send mail.
   If you are already in the work item display, choose Work item → Send mail.
14. Enter the text of the mail.
15. Send the mail to any recipients who are available as mail addressees
   The mail text is added to the work item as an attachment. The work item remains in your Business Workplace with the same status.

You can still complete the work item despite the fact that you have sent a mail. Since this makes the mail which has been sent superfluous, the recipient of the mail receives a second mail informing him/her that the (first) mail no longer needs to be dealt with.

Reply ends resubmission
Optional: If you do not want to process the work item until the mail has been replied to, define a resubmission for the work item. To do this, position the cursor on the entry, choose Resubmit and enter a date in the future.
The work item is resubmitted irrespective of this date and appears in your workflow inbox when the reply to the mail is received.

Procedure as recipient of a mail
You receive a mail that can be executed.
17. Read the mail.
18. Execute the mail, if applicable. The work item referred to by the sender in their mail is displayed. For further information on the object to be processed, choose Goto → Object to display the default attribute or execute the default method (generally Display) for the processed object.
   You can execute this work item if you are one of the possible agents of this task.
19. Reply to the mail. To do this, choose Document → Reply or Document → Reply w/reference when the mail is displayed. Then enter your reply and save your entries.
20. Choose Document → Send. On the send screen, the work item to which the mail referred is entered as the “recipient”. Do not change this entry. Send the reply to the proposed recipient.
Result

The reply sent ends a resubmission which may have been defined for the recipient work item. This work item is appended as an (additional) attachment and can be read as such by the original user and taken into account.
Display Agents

Use
This function displays information about the agents of the work item.

Features
The system displays the following agents for the work item:

- Recipient [Extern]
- Possible agents [Extern]
- Excluded agents [Extern]

You can choose a compressed display - containing only the user names (users only function) - or a complete display with additional information on the relationships used by the system to determine the agents (overall view function). You can also display the organizational assignment of each user.
Note that this information is only available for work items of types for which an agent is logical and necessary.

Activities
You can display the agents from the work item display [Seite 1411] by choosing Goto → Agents → ...
Changing Deadlines

Use
You use this function if you want to change the deadlines for a work item at workflow runtime. This dialog box displays all the deadlines of work item processing. Please also refer to Current Dates/Times of a Work Item [Extern].

Features
Depending on whether the work item represents a step in the workflow or a single-step task, the deadlines were either specified when this step was defined in the workflow definition or when the single-step task was started in dialog. The dialog box is split into the following sections:

Deadlines
- Start by (latest start)
- End by (latest end)

Planned deadlines
- Start by (requested start)
- End by (requested end)

A monitored deadline shown in blue is in the past. If  is also displayed, the appropriate action has been initiated. This generally involves informing the deadline recipient.

Actual dates/times
- Created on: Creation date/time of the work item
  (Technically: The work item is created with the status ready or - if the requested start date has not yet been reached - with the status waiting).
- Processed from: Start of processing
  (Technically: Transition of the work item to the status in process).
- Completed on: End of processing or date when set to 'done').
  (Technically: Transition of the work item to the status completed.)

Activities
To execute this function, choose in the work item display [Seite 1411].
Attachment Management

Use
One or more attachments can be assigned to each work item that appears in the Business Workplace's workflow inbox.
Attachments are documents written either with a SAPscript editor (document classes RAW, SCR) or with a PC application (document classes DOC, URL, PPT, XLS, PDF, ...) and then imported. You can enter new documents as attachments or create attachments from existing files.

Features

General
The attachment is automatically
- Added to the work item container
- Added to the container of the superordinate workflow
- Added to the containers of the subsequent work items in the workflow

You can define default documents for the individual document classes. For further information, refer to Default Documents [Extern].
Attachments can be displayed by the recipients of the subsequent steps. But they cannot be changed and, therefore, have a document character.

A superior who is to make a decision on releasing a budget can enter an attachment justifying their decision. The selected agents of the subsequent steps can display this attachment.

If a work item has attachments, this is indicated by a symbol in the Attachments column in the Business Workplace. You can also execute the function for processing an attachment by double-clicking in this column (column header AT). If an attachment already exists, it is displayed.

If a work item has attachments, this is indicated by the symbol in the work item display.

Functions on attachments

| Display | Displays the attachment selected in the dialog box Existing attachments. |
| Create  | Creates a new attachment. If you want to import an existing document from your PC to the attachment, choose Import. |
| Change  | Displays the selection of Changeable attachments. |
| Delete  | Displays the selection of Deletable attachments. |

Activities
You can access attachment maintenance in the Business Workplace by choosing or the relevant context menu (right-hand mouse button).
Reject Execution

Use
You can use this function if you need to reject execution of a work item for business or technical reasons. This function is only available for work items of type W.

- The table entry to be processed already exists or the material whose master data is to be changed is no longer used.

Prerequisites
This function is only available if the property processing rejectable has been selected for the related activity in the workflow definition.

Features
Processing of the work item is terminated with the reject execution function. The subsequent steps defined in the workflow definition are executed.
Do not use this function if you do not want to or cannot process the work item for personal reasons (not responsible, not competent). In this case, replace the reserved work item or forward it.

Activities
You can access this function in the Business Workplace [Seite 1368] by choosing Other functions or the work item display (Work item → Other functions → Reject processing).
Execute Together

Use
You can select various work items and then execute them together. This function is only available for work items of type W.

The work items do not necessarily have to belong to the same workflow.

Prerequisites
The work items must refer to the same single-step task.
For information about other prerequisites for this function, in particular with regard to the underlying method, refer to Creating an "Execute Together" Method [Extern].

Features
Only the first of the selected work items is proposed for execution. The entries you make here also apply to the other work items to be executed together.

Activities
Select the work items that you want to execute together. Then choose Execute together via Other functions... in the Business Workplace [Seite 1368].
Grouping According to Sort Key

Use
Each work item carries the two container elements _WI_Object_ID and _WI_Group_ID in its container. Both elements have been defined to hold an object reference.

- The container element _WI_Object_ID automatically contains the reference to the object to be processed in the work item
- The container element _WI_Group_ID can contain an object reference, which must be assigned to this container element in a binding or via initial value assignment.

The object reference assigned via _WI_Group_ID is generally not identical to _WI_Object_ID, but is derived from the work item execution environment. It is used to group work items that refer to different objects or object types but are nevertheless connected.

Features

Display default attributes in workflow inbox
The default attributes [Extern] of the objects referenced in _WI_Object_ID and _WI_Group_ID are available in the Business Workplace's workflow inbox [Seite 1408] under the column headers Object or Group for grouping, sorting and filtering purposes.

Material master data is processed in several instances of a workflow. The container element _WI_Object_ID in the containers of the individual work items therefore always contains the reference to the object of type BUS1001 (material) to be processed.

The container element _WI_Group_ID is assigned the expression &Material.Labor& (laboratory/drawing office for material as object reference) in the relevant steps of the workflow definition. The default attribute of the laboratory is therefore available as a group.

Default methods
The default method [Extern] of the objects referenced in _WI_Object_ID and _WI_Group_ID is executed by double-clicking in the Object or Group column.
Change Priority

Use
The priority of a work item is a measure of its urgency. It can be used as a sort criterion for organizing the workflow inbox. This function is only available for dialog work items.

Features
End users can display and change the priority of a work item. The priority is between 1 (highest) and 9 (lowest).

The change in priority can also be passed on automatically to the superordinate workflow and then to all work items created subsequently. The Pass on priority to subsequent steps indicator must be selected for this to take effect.

If a higher priority (lower number) is defined in the workflow definition for one of the subsequent steps, it is not changed.

Activities
You can access the Change priority function in the Business Workplace by choosing Other functions...
Create Link

Use
By storing links to work items in [folders](#), you can organize your work effectively using a personal folder hierarchy.

Features
The work item is added to a personal or shared folder as a link. This work item can be displayed from this folder and executed by its recipients. Work items can have an unlimited number of links.

Activities
You can access this function in the [Business Workplace](#) by choosing Environment...
Forward Work Item

Use
Forwarding passes a work item on to another user for execution.

Integration
The range of users to whom a work item can be forwarded is determined from the task definition as follows:

<table>
<thead>
<tr>
<th>Task definition:</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Task is a <code>general task [Extern]</code></td>
<td>Work item may be forwarded to all users.</td>
</tr>
<tr>
<td>The indicator <code>General forwarding allowed</code></td>
<td>Work item may be forwarded to all users.</td>
</tr>
<tr>
<td>The indicator <code>General forwarding not allowed</code></td>
<td>Work item may only be forwarded to the recipients [Extern] of the task.</td>
</tr>
</tbody>
</table>

Features
The user has the following search options to select the new recipient:

- The F4 input help can be used to determine the new recipient by performing a generic search for name components.
- The structure search can be used to determine the new recipient using the graphical display of the organizational plan.

The new recipient does not have to be an actual user. (The work item can also be forwarded to an organizational unit or a job.)

Activities
You can access this function in the Business Workplace [Seite 1368] by choosing .
Change Work Item

Use
The function enables you to change information and data associated with the selected work item.

Features

Changes in the initial screen
In general, the changes that are allowed depend on the type and status of the work item. The following changes are possible:

- **Work item text**
- **Priority**
- **Deadline data**
  
  You can set and change deadline data for the start and end deadlines of the work item. You can only make specifications for deadlines that are still being monitored and are in the future. For example, dialog work items with the *ready* status can no longer be assigned a requested start.

Manual intervention into the processing of work items
If you are an administrator with the appropriate authorization, you can intervene manually into the processing and therefore into the process flow of a workflow. This is particularly helpful for dealing with errors. The following changes are possible:

- **Set to ‘ready’ manually**
  
  The work item status is changed from *waiting* to *ready*.
  
  The work item therefore appears in the workflow inboxes of the selected agents.

- **Complete manually**
  
  The work item status is changed to *completed*.
  
  In the case of dialog work items, this status change (together with the current work item container) is passed on to the workflow system for evaluation. If the object method to be executed is a synchronous method with a result, the possible result values are displayed for selection.

- **Replace manually**
  
  The status of the work item is reset to *ready*.
  
  The work item is therefore displayed again in the workflow inboxes of all selected agents.

- **Restart after error**
  
  Work items and, in particular, workflows with errors can be restarted after the errors have been corrected.
  
  The information written by the workflow system in a log when the error occurred is now taken into account.

- **Delete logically**
  
  The status of the work item is changed to *logically deleted*. 
In the case of dialog work items, this status change is passed on to the workflow system.

**Locking/unlocking execution**
An administrator can lock or unlock the current work item for execution. To do this, you select the appropriate function from the *Edit* menu.

**Deleting deadlines**
To delete a deadline you have changed or entered and reset it to its initial values, choose *Edit → Delete deadline*.

**Changing work item containers**
Choose *Edit → Change container* to go to the editor for changing the work item container. The contents of the container for the relevant work item are displayed. You can change the current, runtime-specific data for the particular work item.
The container may still have elements that do not currently have a value. These elements are hidden as standard. The presence of elements that are not displayed is shown by an indicator. To display these elements, choose *Edit → Show elements*.
To add more lines to container elements defined as multiline, choose *Edit → Additional line*.

**Activities**
You can access this function in the *Business Workplace* by choosing *Other functions*, the relevant context menu or the *technical work item display*. 
Set Work Item to Done

Use
This function is used by the recipient of a work item to confirm explicitly that processing of this work item has been completed. As long as explicit confirmation has not been provided, the work item has executed status and can be executed again or forwarded.

The status of the work item changes from executed $\rightarrow$ completed.

Prerequisites
This function is only possible on a work item if a setting was made in the definition of the associated single-step task stating that the end of processing must be confirmed explicitly.

Activities
You use this Business Workplace function if the status of the work item has been changed to executed by a terminating event.
In general, however, the work item is not set to done via the Business Workplace. This function is usually provided as a dialog box directly after a work item has been processed.
Missed deadline work items [Extern] must always be confirmed explicitly. After execution, they remain in the status in process until they are set to done.
You can access this function in the Business Workplace [Seite 1368] by choosing $\text{Other functions.}$
Workflow Outbox

Use
The workflow outbox lists the workflows started and the work items forwarded and executed by the current user.

Integration
As is the case for the workflow resubmissions [Seite 1443] and the workflow inbox [Seite 1408], the workflow outbox is an integral part of the Business Workplace.

Views in the workflow outbox

Started workflows
This view shows work items for the tasks started by you as a user in dialog or by a triggering event whose event container contains your user name as _Evt_Creator.

Work items executed by me
The work items executed by you are displayed in this view.

Forwarded work items
The work items forwarded by you are displayed in this view.

Workflow Functions
The screen area in which the worklist is displayed has a toolbar containing the workflow functions and functions for configuring the list. For information about these functions, refer to the documentation on the ABAP List Viewer [Extern].
The following functions are available for the different workflow outbox views:

- **Update**
  You use this function to update the workflow outbox.

- **Display work item** (not in Started workflows view)
  This function goes directly to the work item display [Seite 1411]

- **Change agent**
  You use this function to perform an ad-hoc agent assignment [Seite 1097], [Seite 1411]

- **Display workflow log**
  This function displays the workflow log [Seite 1420].

- **Display attachments** (refer to attachment management [Seite 1429])

- **Environment**
  - Display objects in workflow (dialog work items only)
  - Display workflow relationships [Seite 1436]
  - Start Workflow [Seite 1450]
The functions described above can be called using the relevant context menu.

**Activities**

You can access the workflow outbox in the overview tree in the Business Workplace via Workplace → Outbox.
Workflow Resubmissions

Use
The workflow resubmissions [Extern] function can be used to resubmit work items for processing at a later point in time or periodically and to display these work items in a list. Technically, the following applies to work items which appear in your workflow resubmission folder: the work item status is set to waiting, your name is entered as the actual agent, and the requested start date is set to the resubmission date.

Integration
As is the case for the workflow inbox [Seite 1408] and the workflow outbox [Seite 1441], the workflow resubmissions is an integral part of the Business Workplace.

Features
Workflow Functions
The screen area in which the worklist is displayed has a toolbar containing the workflow functions and functions for configuring the list. For information about these functions, refer to the documentation on the ABAP List Viewer [Extern].

The following functions are available in the workflow resubmissions:

- **Update**
  You use this function to update the workflow resubmissions.

- **Display work item**
  This function goes directly to the work item display [Seite 1411].

- **Display workflow log**
  This function displays the workflow log [Seite 1420].

- **Display attachments**
  This function is only available if the work item has attachments. Refer to attachment management [Seite 1429].

- **End resubmission**
  The work item is put back into the workflow inbox. It has the status reserved.

- **Environment**
  - Display objects in workflow (dialog work items only)
  - Display workflow relationships [Seite 1436]
  - Start Workflow [Seite 1450]
  - Create link [Seite 1434]

The functions described above can be called using the relevant context menu.

Activities
You can access the workflow resubmissions in the overview tree in the Business Workplace [Seite 1368] via Workplace → Resubmissions.
You are asked for a date for resubmission of the work item. The work item then disappears automatically from the workflow inbox and is not displayed there again until the date specified.
Work Item Preview

Use
The work item preview is in the bottom right corner of the Business Workplace screen and provides a preview of the properties of a work item. All the information on a work item is displayed if you choose the function *Display work item* in the worklist of the workflow inbox [Seite 1408] or the workflow resubmissions [Seite 1443]. This function goes directly to the work item display [Seite 1411]. A concise view of the workflow log [Seite 1420] is offered in the workflow outbox [Seite 1441] when the started workflows are displayed.

Integration
The work item preview is part of the Business Workplace [Seite 1368].

Features

Workflow functions
The description of the work item is displayed in the work item preview. It is also possible to access objects and attachments of the work item directly from the preview. The concise view of the workflow log enables you to access processed steps (individual work items), their agents and information objects addressed so far. For more information about the concise view of the workflow log, please refer to Workflow Relationships [Seite 1436].

Tips & tricks
"Tips & tricks" are displayed when you position the mouse pointer on the title of a "tips & tricks" entry. Whenever the work item preview is updated, a new title is offered. You can activate/deactivate "tips & tricks" in your personal workflow settings [Seite 1395] (Settings → Workflow settings → Personal settings).

User exit
The work item preview can be adapted to customer-specific requirements using a user exit [Extern].

Activities
The work item preview is activated/updated whenever you select a work item in your worklist.
Workflow Toolbox

Use

*SAP Business Workflow's* Workflow Toolbox enables the user to access workflow functions even during a workflow-driven application transaction. The following functions are now available even during method execution:

- Display work item text
- Display task text
- Availability of attachments and objects as links
- Create/display/change/delete attachments
- Display possible agents
- Display workflow log
- Forward
- Resubmit
- Send mail

The toolbox is easy to activate from the application side in its dialog transactions. The full functionality of the toolbox is encapsulated in class `CL_WAPI_WF_TOOLBOX`.

The workflow WS70000651 *Demo for WI execution with WF toolbox* can be started by any user with transaction SWUI_DEMO, and generates work items in the Business Workplace. The toolbox demonstrates the execution of these work items in its various forms. For further information, refer to *Demo and Test Workflows* [Seite 1499].

Features

**Class CL_WAPI_WF_TOOLBOX**

The class `CL_WAPI_WF_TOOLBOX` encapsulates the full functionality of the toolbox. An instance of this class is created directly before the `CALL SCREEN` of the application `screen`. This instance provides all the `GUI objects` it requires and manages them as well.

**Reusing worklist client functionality**

**Work item preview**

The toolbox uses the `work item preview` [Seite 1445] in the Business Workplace to display work item text, task description and links to attachments and other objects.

**Workflow log**

The first tab page of the ActiveX version of the `workflow log` [Seite 1420] is used again to display the complete workflows already processed.
Various Workflow Toolbox modes

**Toolbar only (embedded)**
If the toolbox is used in its simplest mode C_MODE_INPLACE_WITHOUT_INFO, it requires a custom container of approximately 1 x 4 cm on the screen.
The work item preview and workflow log are displayed in a separate new screen (modeless). A pushbutton is available to the user for this.

**Toolbar and information (embedded)**
If the toolbox is used in mode C_MODE_INPLACE_WITH_INFO, it is displayed in two parts on the screen: The toolbar is displayed above the preview initially. The user can switch between preview and log in the lower part.

**Toolbar and information (modeless)**
If the toolbox is used in mode C_MODE_MODELESS, it is displayed in two parts in a modeless window: The toolbar is displayed above the preview initially. The user can switch between preview and log in the lower part.

**Menu button only (embedded)**
If the toolbox is used in mode C_MODE_INPLACE_BUTTON_ONLY, it requires a custom container of at least 1 x 1 cm on the screen, depending on whether just an icon or an icon with text is to be displayed. The toolbox opens a modeless window to display work item preview and workflow log when the user chooses the relevant menu option.
Reference Documentation

Definition
Complete documentation of the SAP Business Workflow functions.

Purpose
The reference documentation is always used when you are looking for specific information. The purpose of the reference documentation is to be a "reference book" providing complete documentation for SAP Business Workflow.

Structure
The SAP Business Workflow reference documentation consists of four structures:
- **BC - SAP Business Workflow - Navigation [Seite 972]**
  This structure contains a full description of the functions of the SAP Business Workflow development and administration environment.
- **BC - SAP Business Workflow - Tutorials [Seite 1581]**
  This structure contains tutorials for the SAP Business Workflow development and administration environment.
- **BC - SAP Business Workflow - Scenarios in Applications [Seite 1703]**
  This structure contains the documentation for the R/3 application component tasks and workflows supplied by SAP.
- **BC - SAPforms [Seite 1718]**
  This structure contains the documentation for integrating Web forms and electronic forms with the R/3 System.

Integration
There are links in the individual roles of the role documentation to various parts of the reference documentation. The following workflow roles [Seite 94] are defined:
- **Process consultant [Seite 96]**
- **Workflow developer [Seite 220]**
- **Workflow system administrator [Seite 683]**
- **Workflow agent [Seite 805]**
- **Process controller [Seite 884]**

Only the reference documentation contains a description of all functions.
Purpose

The list of individual help topics is based on the overview tree for SAP Business Workflow. You therefore have application-specific access to the various functions.

Process Flow

You commence navigation on the standard screen SAP Easy Access and choose Tools → Business Workflow → Development. You can then go to the following workflow functions:

Definition Tools

- Business Workflow Explorer [Seite 975]
- Wizards
  - Workflow Wizard Explorer [Seite 977]
  - Create "Call Workflow From Message" [Seite 1456]
  - Workflow Wizard Repository [Extern]
- Workflow Builder [Seite 1002]
- Business Object Builder [Seite 1100]
- Workflow Configuration [Seite 1171]
- Tasks/Task Groups [Seite 1172]
- Worklist Client
  - Dynamic Columns for Worklist [Seite 1405]
  - Maintain Views [Seite 1403]
- Standard Roles [Seite 1276]
- Workflow Start Transactions [Seite 1455]
- Organizational Management
  - Organizational Plan [Seite 1708]
  - SAP Org. Objects
- Events [Seite 1321]
  - Event Creation [Seite 1322]
    - Set Up With Wizard [Seite 1324]
    - Change Documents [Seite 1325]
    - Status Management [Seite 1334]
    - Message Control [Seite 1337]
    - Logistics Information System (LIS) [Seite 1338]
Event Linkages

- Type Linkages [Seite 1358]
- Workflow Start Conditions [Seite 1458]

Runtime Tools

- Business Workplace [Seite 1368]
- Start Workflow [Seite 1450]
- Start Workflow (Test Environment) [Seite 1452]
- Workflows For Object [Seite 1436]
- Workflows For Object Type [Seite 1436]

Reporting

- Work Item Analysis [Seite 1484]
- Workload Analysis [Seite 1485]
- Task Analysis [Seite 1488]

Utilities

- Work Item Selection [Seite 1490]
- Diagnosis [Seite 1493]
- Customizing [Seite 1504]
- Consistency Check [Seite 1495]
- Workflow Trace [Seite 1496]
- Events [Seite 1321]

  - Type Linkages [Seite 1358]
  - Instance Linkages [Seite 1354]
  - Simulate Event [Seite 1363]
  - Create Event [Seite 1365]
  - Event Trace [Seite 1366]
  - Event Queue Browser [Seite 1526]

- Workflow RFC Monitor [Seite 1498]

Environment

- Organizational Management [Extern]
- Start Demo Workflows [Seite 1499]
- Start Test Workflows [Seite 1499]
- Demo: Embedded Inbox [Seite 1499]
- Demo: Fill Out Form [Seite 1499]
Administration

- Workflow Runtime [Seite 1500]
- Event Manager [Seite 1500]
- Workflow Definition Time [Seite 1500]
- SAPforms [Seite 1727]
Business Workflow Explorer

Use
You use the Business Workflow Explorer [Extern] to display an overview of all tasks for a selected search range. All the tasks displayed can also be processed from the Business Workflow Explorer.

Prerequisites
You have to define a search range when you first call the Business Workflow Explorer. When the Business Workflow Explorer is called thereafter, the last active search range is used. The Business Workflow Explorer is only available on a 32-bit platform.

Features
In the left-hand screen area, the Business Workflow Explorer displays all the tasks assigned to the selected search range. In the right-hand screen area, the Business Workflow Explorer displays the workflow (with associated tasks) or the task, which you selected in the left-hand screen area.

Activities
You can call the Business Workflow Explorer by choosing Tools → Business Workflow → Development → Definition tools → Business Workflow Explorer.

Task-related functions
The functions available on the tasks displayed are shown in the context menu.

Quick view
With tasks an overview of the most important properties is displayed and with workflows the workflow definition is displayed.

Display or change
A new session is opened to display or change.

Create
You can create new tasks and workflows. If you have selected a workflow you create a new workflow template, and if you have selected a task you create a new standard task.

Workflows of a definition
You go to the work item selection in which the task ID is already entered. Further information on carrying out the selection, refer to Work Item Selection [Seite 1490].

Where-used list
All workflow definitions in which the task is used are displayed.

Choose another search range
This enables you to change the Business Workflow Explorer search range. The search range can be limited according to specific criteria. These criteria are:

- Task groups [Extern]
- Application components
- Your own or all tasks and workflows saved locally
- Tasks and workflows with the SAPphone property
Business Workflow Explorer

- The last ten workflows processed
- Workflows defined as demo workflows or as verification and test workflows.
- Tasks that refer to a particular object type and possibly a particular method

If you define a new search range, it replaces the last active range.

**Search / extended search**
You can use this to search for tasks and workflows. The standard search function only searches for the term in your current search range. The extended search covers all existing tasks and workflows.

**Copy**
You select a task or a workflow in the right-hand tree and choose \( \text{Copy} \). The selected task/workflow is copied to form a new task/workflow. You only see the new task/workflow in the current session of the *Business Workflow Explorer* as well if it is in the search range displayed.

If you copy a task of the type T (customer task) or a workflow of the type WF (workflow task), the copy will be defined as a standard task or workflow template since customer tasks and workflow tasks will not be supported in the future.

**Detailed view**
You can activate a detailed view in the right-hand tree by choosing \( \text{View} \rightarrow \text{Details} \). This displays attributes of the task or workflow. The business object type used is also displayed with tasks.
Workflow Wizard Explorer

Use
You use the Workflow Wizard Explorer [Extern] to:
- Find out about the Workflow Wizards available
- Define a workflow with a Workflow Wizard

Workflows created with a Workflow Wizard are never complete and are only a framework. To create a workflow that meets your requirements you must use the Workflow Builder [Seite 1002] to edit the workflow generated.

Integration
New Workflow Wizards can only be created in the Workflow Wizard Repository [Extern]. To display the technical content of the Workflow Wizard, select it and choose 📜.

Workflow Wizard Explorer as part of the Workflow Builder
The Workflow Wizard Explorer can be used to execute the individual Workflow Wizards from the Workflow Builder. A workflow definition created with a Workflow Wizard is integrated directly into the workflow definition.

Workflow Wizard Explorer as a separate tool
The Workflow Wizard Explorer can be called as a separate tool from which you can start Workflow Wizards.

Features
A Workflow Wizard supports you in the definition of a workflow with a dialog sequence. Before the Workflow Wizard starts its work, you must make certain settings. The dialog box for these settings is displayed automatically when you execute a Workflow Wizard. This dialog box varies depending on whether you call the Workflow Wizard Explorer via the Workflow Builder or as a separate transaction.

Workflow Wizard Explorer when called in the Workflow Builder
The following fields and settings need to be maintained:
- Insert workflow definition only, do not create workflow template
  - The steps created by the Workflow Wizard are inserted into the workflow definition at the selected position in accordance with the positioning rules [Extern].
- Insert workflow definition and create workflow template
  - The steps created by the Workflow Wizard are inserted into the workflow definition at the selected position in accordance with the positioning rules [Extern]. In addition, a new workflow is created, which contains the workflow definition generated.
  - The workflow definition created with the Workflow Wizard can be used again via its task.

Workflow Wizard Explorer when called as a separate tool
The following fields and settings need to be maintained:
- Extended settings: Activate workflow definition
  - If this indicator is set, the workflow definition is activated after generation.
Workflow Wizard Explorer

- Extended settings: Classify task as general task [Extern]
  If this indicator is not set, you have to assign possible agents to the task.

Documentation on Workflow Wizards
You can go to this documentation from every Workflow Wizard. To display the documentation, select in the line of the Workflow Wizard in question.

Activities

Workflow Wizard Explorer in Workflow Builder
To call the Workflow Wizard Explorer in the Workflow Builder, select a step or an outcome and choose Wizards → Workflow Wizard Explorer. The part of the workflow definition created by the Workflow Wizard is inserted into the existing workflow definition at the selected position in accordance with the positioning rules [Extern].

Workflow Wizard Explorer as a separate tool
To call the Workflow Wizard Explorer as a separate tool, choose Tools → Business Workflow → Development → Definition Tools → Wizards → Workflow Wizard Explorer. You have the opportunity to continue any Workflow Wizards whose execution was aborted.

Starting Workflow Wizards
To start a Workflow Wizard, select in the line of the Workflow Wizard in question.
Workflow Wizards for Approval Procedures

Use
This folder contains Workflow Wizards that define workflows for various approval procedures. The approval procedures vary in their complexity.

Deadline monitoring for an approval procedure created must then be defined manually in the Workflow Builder.

All approval procedures have the same evaluation part. For further information, refer to Evaluation Part in All Approval Workflows [Extern].

Integration
The approval task [Extern] you use in the Workflow Wizard must reference the method Approval. The method is inherited from the interface Approval [Seite 1112]. Each approval object type should support the interface Approval and implement the method Approval.

For an extended approval, you use an approval task that references the method XApproval. Implement this method from the interface Extended Approval [Seite 1113].

Features
The following Workflow Wizards define approval procedures:

<table>
<thead>
<tr>
<th>Workflow Wizard</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Hierarchical Approval Workflow [Seite 986]</td>
<td>This Workflow Wizard maps the basic form of an approval procedure. Several recipients specified in the definition decide one after the other whether the approval object should be approved or rejected.</td>
</tr>
<tr>
<td>Dynamic Hierarchical Approval Workflow [Seite 980]</td>
<td>This Workflow Wizard implements a hierarchical approval procedure for a particular object type and with a particular approval task. The levels and the recipients are not determined until runtime.</td>
</tr>
<tr>
<td>Extended Hierarchical Approval Workflow [Seite 982]</td>
<td>This Workflow Wizard allows not only rejection or approval on every level, but also proposed rejection or approval.</td>
</tr>
<tr>
<td>Release Subworkflow for FI (FIPP, BSEG) [Seite 984]</td>
<td>The Workflow Wizard is specially adapted for use as a subworkflow in the workflows FIPP_FRAME* and BSEG_FRAME*. These workflows from the Financial Accounting area are distinguished by the fact that the actual approval procedures are implemented as subworkflows that are determined dynamically at runtime. The Workflow Wizard helps to provide these subworkflows.</td>
</tr>
<tr>
<td>Parallel Approval Workflow [Seite 988]</td>
<td>This Workflow Wizard creates an approval procedure in which the recipients decide to approve or reject the approval object at the same time. An approval often does not require agreement from all recipients. Agreement from a particular number of recipients specified in the definition often suffices (agreement of n from m recipients).</td>
</tr>
</tbody>
</table>
Dynamic Hierarchical Approval Workflow

Use
This Workflow Wizard creates a dynamic, hierarchical approval workflow. During execution of the Workflow Wizard you determine the type of the object to be approved and the approval task [Extern] to be used for the approval.
You do not determine the extent of the approval procedure and the recipients approving at each level until the workflow is executed. When you do this you can assign a user, a position [Extern], a job [Extern], an organizational unit [Extern] or a work center at each level. Once a recipient does not approve the object, the recipients higher in the hierarchy do not receive the object for approval.

Prerequisites
The object type of the objects to be approved must have an approval method and should also have a method for displaying the objects, which is defined as the default method. The Workflow Wizard can insert this default method into the approval step as a secondary method [Extern]. All users who are to be able to carry out approvals in this workflow must be possible agents [Extern] of the approval task.

Features
The workflow created by the Workflow Wizard has three parts:
4. The users who are to approve the object are established.
5. Each approver gets the opportunity to approve in turn in a loop. If someone rejects, the loop is ended and no further approvals are sought.
6. The approval status is evaluated and the applicant has an opportunity to revise the object if it was rejected.
The workflow initiator establishes the approvers from the possible agents of the approval task. When the Workflow Wizard is executed you can specify whether the workflow initiator can be one of the approvers as well. When you choose the approvers you must ensure that the users who approve the object first are lowest in the hierarchy.
All approvers receive the object for approval in their respective Business Workplace in a loop. Once an approver does not approve the object, execution of the loop is aborted. If the object is approved, the approval status is changed.
The approval status is checked in the evaluation part of the workflow. Only once all approvals have been made is the approval result set to approved. The workflow initiator receives an e-mail stating that the object has been approved. Otherwise, the workflow initiator receives a work item in their Business Workplace. They must decide whether the object is to be revised or deleted. The approval result is set accordingly.

Activities
After execution of the Workflow Wizard you still have to add steps to the workflow, in which the object that is to be approved is created or established. This object must be available in the container element ApprovalObject.
When the Workflow Wizard is executed, the following container elements are created in the workflow container:

<table>
<thead>
<tr>
<th>Description (technical name)</th>
<th>Explanation</th>
</tr>
</thead>
<tbody>
<tr>
<td>Approval object (ApprovalObject)</td>
<td>Reference to the object to be approved. The object to be approved and the approval task are determined when the Workflow Wizard is executed.</td>
</tr>
<tr>
<td>List of all possible agents (AllAgents)</td>
<td>This element is no longer required.</td>
</tr>
<tr>
<td>List of selected agents</td>
<td>The responsible agents [Extern] of the approval task, who are selected</td>
</tr>
</tbody>
</table>
## Dynamic Hierarchical Approval Workflow

<table>
<thead>
<tr>
<th>(ApprovalAgents)</th>
<th>by the workflow initiator from the possible agents of the approval task, are stored in this multiline container element.</th>
</tr>
</thead>
<tbody>
<tr>
<td>Approval status (ApprovalState)</td>
<td>The status is initialized at the number of approvers in a background step, and decreased by one with each approval. If the status is zero on evaluation, the object was approved. If it is greater than zero, the object was rejected.</td>
</tr>
<tr>
<td>Table index (Index)</td>
<td>This container element is used for index access to the table of selected agents.</td>
</tr>
<tr>
<td>Current agent (CurrentAgent)</td>
<td>The user who processes the work item is saved in this container element.</td>
</tr>
<tr>
<td>List of excluded agents (ExcludedAgents)</td>
<td>The list of agents contains all users, positions [Extern], jobs [Extern], organizational units [Extern] and work centers that have been involved in the approval process. This list is used to exclude users who have already granted an approval from approving the same object again. You can also include the workflow initiator in this list. This prevents applicants from being able to approve their own requests.</td>
</tr>
<tr>
<td>Rejection indicator (RejectedFlag)</td>
<td>This indicator is set in the event of a rejection.</td>
</tr>
<tr>
<td>Approval result (Result)</td>
<td>This field contains the result of the approval procedure after the evaluation part has been executed: A for approved, D for delete, E for edit</td>
</tr>
</tbody>
</table>
Extended Hierarchical Approval Workflow

Use
Several users are involved at each approval level in an extended, hierarchical, approval procedure. Only some of them are authorized to actually approve or reject the object. The other users are only authorized to propose that the object be approved or rejected.

Each approval level can only be left when the object has actually been approved or rejected. To be proposed for approval or rejection is not sufficient to leave the approval level. A user who has granted an approval, or proposed an approval or rejection, is excluded from further processing in the same approval procedure.

If the object is approved on an approval level, the next approval level is called. You define the number of approval levels when executing the Workflow Wizard.

Once one user does not approve the object, no further approval levels are initiated.

You can specify whether the same approval task is to be used for all approval levels when the Workflow Wizard is executed.

Prerequisites
The object type of the objects to be approved must have an extended approval method and should also have a method for displaying the objects, which is defined as the default method. The Workflow Wizard can insert this default method into the approval step as a secondary method. All users who are to be able to carry out approvals in this workflow must be possible agents of the approval task.

Features
With the Workflow Wizard for extended hierarchical approvals, you can create approval procedures of any nesting depth.

In the first step of any approval procedure, the approval status (container element ApprovalState) is first initialized at one.

This initialization is followed by the approval steps nested in each other, all of which have at least four possible results (approval, rejection, proposed approval, proposed rejection). In each approval step, the approval method of the object to be approved is executed and, optionally, its default method as a secondary method.

In the event of approval, the approval status is increased by one and the next approval level initiated.

In the event of rejection, the workflow goes immediately to its evaluation part. If the approval procedure is terminated in this way, the approval status is always less than or equal to the number of approval levels.

If the approval or rejection is only proposed by an agent due to their restricted authorization, the approval procedure remains at this approval level. The agent is included in the list of excluded agents. A work item is generated again, and the object is submitted for approval again.

For further information, refer to Evaluation Part of All Approval Workflows.

Enhancements to the workflow definition
The workflow definition created by the Workflow Wizard can only be used practically as part of a superordinate workflow definition. Steps still need to be added to it in which the object that is to be approved is created or established.

A reference to this object must be stored in the container element ApprovalObject in the workflow container.

Activities
An extended hierarchical approval procedure created by a wizard also requires additional container elements in the workflow container:
## Extended Hierarchical Approval Workflow

<table>
<thead>
<tr>
<th>Data type</th>
<th>Description (technical name)</th>
<th>Explanation</th>
</tr>
</thead>
<tbody>
<tr>
<td>Object reference</td>
<td>Approval object (ApprovalObject)</td>
<td>Reference to the object to be approved. The object to be approved and the approval task must be determined at definition time.</td>
</tr>
<tr>
<td></td>
<td>Address object (AddressObjects)</td>
<td>Reference to the address objects of those who are informed of the approval.</td>
</tr>
<tr>
<td>ABAP Dictionary reference</td>
<td>Approval status (ApprovalState)</td>
<td>Status of the approval procedure. The status is initialized at one at the start, and increased by one with each approval. If the status is greater than the number of approval levels on evaluation, the object was approved.</td>
</tr>
<tr>
<td></td>
<td>Current agent (CurrentAgent)</td>
<td>The user who processes the work item is saved in this container element.</td>
</tr>
<tr>
<td></td>
<td>List of excluded agents (ExcludedAgents)</td>
<td>The list of agents contains all users, positions [Extern], jobs [Extern], organizational units [Extern] and work centers that have been involved in the approval process. This list is used to exclude agents who have already granted an approval from approving the same object again. You can also include the workflow initiator in this list. This prevents applicants from being able to approve their own requests.</td>
</tr>
<tr>
<td></td>
<td>Decision indicator (DecisionFlag)</td>
<td>This indicator is set when a decision is made (approval or rejection).</td>
</tr>
<tr>
<td></td>
<td>Rejection indicator (RejectedFlag)</td>
<td>This indicator is set in the event of a rejection.</td>
</tr>
<tr>
<td></td>
<td>Approval result (Result)</td>
<td>This container element holds the result of the approval procedure after the workflow. In the case of detailed evaluation, there are the following values: A for approved D for delete E for edit In the case of the minimum evaluation, there are the following values: 0 for approved 4 for not approved</td>
</tr>
</tbody>
</table>
Release Subworkflow for FI (FIPP, BSEG)

Use
The workflow definition created is specially adapted for use as a subworkflow in the workflows FIPP_FRAME* and BSEG_FRAME*.
These workflows from the Financial Accounting area are distinguished by the fact that the actual approval procedures are implemented as subworkflows that are determined dynamically at runtime. The Workflow Wizard helps to provide these subworkflows.
In a hierarchical approval workflow, the type of object to be approved, the approval task [Extern], the levels and the specific agents for each level must be specified at definition time.
The approval procedure is otherwise such that every agent can reject the object to be approved with the consequence that the agents following know nothing of the approval procedure started, because it has already broken down at a lower level in the hierarchy.

Features
In the first step of every release procedure, the release status (element Level) is first initialized at one.
This initialization is followed by the approval steps nested in each other, all of which have at least two possible results (approved or rejected). The approval method of the object to be approved is executed and, optionally, its default method as a secondary method. This execution takes place asynchronously in a new session without influencing the execution of the workflow.
In the event of approval, the approval status is increased by one and the next approval step initiated.
In the event of rejection, nothing happens and the workflow goes immediately to its evaluation part. If the approval procedure is terminated in this way, the approval status is always less than the number of levels.

Enhancements to the workflow definition
The workflow definition created by the Workflow Wizard can only be used practically as part of a superordinate workflow definition. Steps still need to be added to it in which the object that is to be approved is created or established.
A reference to this object must be stored in the container element ApprovalObject in the workflow container.

Activities
A hierarchical approval procedure created by a wizard requires a few additional elements in the workflow container:

<table>
<thead>
<tr>
<th>Data type</th>
<th>Short description (name)</th>
<th>Explanation</th>
</tr>
</thead>
<tbody>
<tr>
<td>Object reference</td>
<td>Approval object (ApprovalObject)</td>
<td>Reference to the object to be approved.</td>
</tr>
<tr>
<td>ABAP Dictionary reference</td>
<td>Release status (Level)</td>
<td>Status of the approval procedure. The status is initialized at one at the start, and increased by one with each approval. If the status is equal to zero on evaluation, the object was approved. If it is greater than zero, the object was rejected.</td>
</tr>
<tr>
<td></td>
<td>Current agent (CurrentAgent)</td>
<td>The user who processes the work item is saved in this container element.</td>
</tr>
<tr>
<td></td>
<td>List of excluded agents (Activity_Excluded_Agents)</td>
<td>The list of agents contains all users, positions [Extern], jobs [Extern], organizational units [Extern] and work centers which are involved in the approval process.</td>
</tr>
</tbody>
</table>
This list is used to exclude agents who have already granted an approval from approving the same object again. You can also include the workflow initiator in this list. Thus, you prevent applicants from being able to approve their own requests.

<table>
<thead>
<tr>
<th>Release result</th>
<th>This field contains the result of the approval procedure after the workflow: X for approved y for not approved</th>
</tr>
</thead>
</table>
Hierarchical Approval Workflow

Use
In a hierarchical approval procedure, one agent must agree before the next agent can decide. Each agent has the option of rejecting the request therefore terminating the entire procedure. Since the workflow can be terminated by any agent, it is very important that the hierarchy be respected (lower positions first, higher positions last), because it would not make sense for a managing director's decision to be nullified by a head of department, for example. This logical sequence cannot be checked by the wizard, so must be defined correctly by the user.

An administrator needs a new PC and enters a purchase order starting a workflow for approval.

First their head of department must agree, and then a designated agent from the IT department must grant their approval. Lastly an employee from Controlling must grant their approval to finally release the purchase order.

Another possibility is that the approvers are on the same hierarchical level but can belong to different departments. The basic principle is otherwise the same.

First a Financial Accounting clerk (from several possible) must agree and then a Controlling administrator (from several possible). It may also make sense, within the double verification principle, for two or more agents from the same department to carry out an approval.

In a hierarchical approval workflow, the type of object to be approved, the approval task, the levels and the specific agents for each level must be specified at definition time.

Prerequisites
The object types that are to use an automatically created approval procedure must have an approval method. It is best if these object types support the interface Approval, which incorporates a method for approval and whose interface is ideal for the Workflow Wizard.

The object type should also always have a method for displaying objects, which should be entered as default method, because the Workflow Wizard integrates the default method of the object type to be approved into the approval step as a secondary method during instantiation.

Features
The Workflow Wizard for hierarchical approvals can be used to create approval procedures with any nesting level.

In the first step of every approval procedure, the approval status (container element ApprovalState) is first initialized with the number of hierarchical levels, that is the number of approvals to be granted. This initialization is followed by the approval steps nested in each other, all of which have at least two possible results (approved or rejected). The approval method of the object to be approved is executed and, optionally, its default method as a secondary method. This execution takes place asynchronously in a new session without influencing the execution of the workflow.

In the event of approval, the approval status is decreased by one and the next approval step initiated. Once all agents have agreed, this is therefore zero.

In the event of rejection, nothing happens and the workflow goes immediately to its evaluation part. If the approval procedure is terminated in this way, the approval status is always greater than zero.
Hierarchical Approval Workflow

For further information, refer to Evaluation Part of All Approval Workflows [Extern].

Extensions to the workflow definition

The workflow definition created by the Workflow Wizard can only be used practically as part of a superordinate workflow definition. Steps still need to be added to it in which the object that is to be approved is created or established.

A reference to this object must be stored in the container element ApprovalObject in the workflow container.

Activities

A hierarchical approval procedure created by a wizard requires certain additional elements in the workflow container, which are created by the workflow system:

<table>
<thead>
<tr>
<th>Data type</th>
<th>Description (technical name)</th>
<th>Explanation</th>
</tr>
</thead>
<tbody>
<tr>
<td>Object reference</td>
<td>Approval object (ApprovalObject)</td>
<td>Reference to the object to be approved. The object to be approved and the approval task must be determined at definition time.</td>
</tr>
<tr>
<td>List of addresses</td>
<td>(AddressObjects)</td>
<td>Reference to the address objects of those who are informed of the approval.</td>
</tr>
<tr>
<td>ABAP Dictionary</td>
<td>Approval status (ApprovalState)</td>
<td>Status of the approval procedure. The status is initialized at the start at the number of approval levels, and decreased by one with each approval. If the status is zero on evaluation, the object was approved. If it is greater than zero, the object was rejected.</td>
</tr>
<tr>
<td>Current agent</td>
<td>(CurrentAgent)</td>
<td>The user who processes the work item is saved in this container element.</td>
</tr>
<tr>
<td>List of excluded agents</td>
<td>(ExcludedAgents)</td>
<td>The list of agents contains all users, positions [Extern], jobs [Extern], organizational units [Extern] and work centers that have been involved in the approval process. This list is used to exclude agents who have already granted an approval from approving the same object again. You can also include the workflow initiator in this list. This prevents applicants from being able to approve their own requests.</td>
</tr>
<tr>
<td>Approval result</td>
<td>(Result)</td>
<td>This field contains the result of the approval procedure after the workflow: A for approved D for delete E for edit</td>
</tr>
</tbody>
</table>
Parallel Approval Workflow

Use
This workflow implements majority decisions of the form \( n \) from \( m \). A total of \( m \) users receive a work item with the activity for approval at the same time. They can also agree or reject at the same time. If the necessary number of \( n \) agreements required for an approval is reached, the object is regarded as having been approved. The object is regarded as rejected when the number of users rejecting it renders it impossible for the necessary number of agreements to be reached.

A 2 from 4 approval procedure is implemented for a purchase order. A total of 4 recipients receive a work item for approval at the same time.

If 2 recipients agree, the purchase order is approved. Once 3 have rejected, the purchase order is rejected.

With this variant \( n \) (necessary agreements) from \( m \) (total number of recipients) must agree. To implement this, the Workflow Wizard creates a workflow with parallel processing with \( m \) branches in which the approvals are carried out.

The approval procedure is terminated once approval has been granted in \( n \) branches, or once the number of branches in which the object has been rejected renders it impossible for the number of necessary agreements to be reached.

For a parallel approval workflow, the type of the object to be approved, the approval task [Extern] and the recipients for each branch must also be specified at definition time.

Prerequisites
The object types that are to use an automatically created approval procedure must have an approval method. It is best if these object types support the interface Approval [Seite 1112], which incorporates a method for approval and whose interface is ideal for the Workflow Wizard. The object type should always, however, have a method for displaying objects, which should be entered as default method, because the Workflow Wizard integrates the default method of the object type to be approved into the approval step as a secondary method during instantiation.

Features
In the first steps, the two elements in the workflow container are initialized with the number of necessary approvals (container element Approvals) or the number of necessary rejections (container element Rejections). The workflow definition then contains a fork that has a branch for each agent who is to receive the approval task. Each approval task can have Approved or Rejected as a result.

In the event of approval, the approval counter (Approvals) is decreased by one and checked to see whether it has reached zero.

In the event of rejection, the rejection counter (Rejections) is decreased by one and checked to see whether it has reached zero.

If one of the check routines ascertains that the approval or rejection counter is less than or equal to zero, the fork is terminated and the evaluation step is implemented.

For further information, refer to Evaluation Part of All Approval Workflows [Extern].
### Enhancements to the workflow definition

The workflow definition created by the Workflow Wizard can only be used practically as part of a superordinate workflow definition. Steps still need to be added to it in which the object that is to be approved is created or established.

A reference to this object must be stored in the container element `ApprovalObject` in the workflow container.

### Activities

The system adds the following container elements to the workflow container:

<table>
<thead>
<tr>
<th>Data type</th>
<th>Description (technical name)</th>
<th>Explanation</th>
</tr>
</thead>
<tbody>
<tr>
<td>Object reference</td>
<td>Approval object (ApprovalObject)</td>
<td>Reference to the object to be approved. The object to be approved and the approval task must be determined at definition time.</td>
</tr>
<tr>
<td>Address object</td>
<td>AddressObjects (AddressObjects)</td>
<td>Reference to the address objects of those who are informed of the approval.</td>
</tr>
<tr>
<td>ABAP Dictionary reference</td>
<td>Number of approvals (Approvals)</td>
<td>This container element counts the agreements. If the necessary number of agreements is reached, the workflow is set to <code>approved</code>.</td>
</tr>
<tr>
<td></td>
<td>Number of rejections (Rejections)</td>
<td>This container element counts the rejections. If the necessary number of rejections is reached, the workflow is set to <code>rejected</code>.</td>
</tr>
<tr>
<td></td>
<td>Exit indicator (Exit)</td>
<td>This indicator exits the fork if it is set to X.</td>
</tr>
<tr>
<td></td>
<td>List of (excluded) agents (Agents)</td>
<td>The list of agents contains all users, positions [Extern], jobs [Extern], organizational units [Extern] and work centers that have been involved in the approval process. This list is used to exclude agents who have already granted an approval from approving the same object again. You can also include the workflow initiator in this list. This prevents applicants from being able to approve their own requests.</td>
</tr>
<tr>
<td></td>
<td>Approval result (Result)</td>
<td>This field contains the result of the approval procedure after the workflow: A for <code>approved</code> D for <code>delete</code> E for <code>edit</code></td>
</tr>
</tbody>
</table>
Workflow Wizards for Circulars

Use
This folder contains Workflow Wizards which define workflows for the forwarding of any documents (circulars).
This enables you to ensure that the circular reaches all recipients and does not remain for too long with the individual recipients.

Features
All workflow definitions for circulars have the following properties in common:

- The object type of the circular is determined, for example:
  - *Office document* (object type SOFM)
  - *Parked document* (object type FIPP)
  - *Notification of absence* (object type FORMABSENCE)
  - *Material* (Object type BUS2001)

- The processing task (and therefore the method) is determined, with which the circular is displayed or processed.

The workflow definitions for circulars can be differentiated as follows:

- Static or dynamic
  - With the static variant, all recipients of the circular are determined in the workflow definition. This specification then applies to all workflows which refer to this definition.
  - With the dynamic variant, the recipients of the circular are not specified individually for the respective current workflow until runtime.

- Parallel or sequential
  - With the parallel variant, all recipients can access the circular at the same time.
  - With the sequential variant, the circular is presented to the recipients for display purposes in sequence. Only when one recipient has finished, does the next recipient get the opportunity to access the circular. With the help of the Workflow Wizard, you can define deadline monitoring so that the circular is forwarded after a specific period of time automatically or the workflow initiator is notified.

The following combinations result:

<table>
<thead>
<tr>
<th></th>
<th>Static</th>
<th>Dynamic</th>
</tr>
</thead>
<tbody>
<tr>
<td>Parallel</td>
<td>Parallel Circular [Seite 997]</td>
<td>Dynamic Parallel Circular [Seite 995]</td>
</tr>
<tr>
<td>Sequential</td>
<td>Circular [Seite 991]</td>
<td>Dynamic Circular [Seite 993]</td>
</tr>
</tbody>
</table>
Circular

Use
A user can send a specific circular (for example a Microsoft Word document or any business document) sequentially to several defined users in the R/3 System, who are to view the circular and make comments if applicable.

In workflows defined with this wizard, you specify all the recipients of the circular in the workflow definition.

If you do not want to define the recipients of the circular until runtime, you must use one of the Workflow Wizards for dynamic circulars.

The workflow definition created with the Workflow Wizard contains the recipient of the circular as a defined "distribution list". This should only contain positions [Extern], but it can also contain users or organizational units [Extern].

The user who starts the workflow (workflow initiator) passes the object for circulation at runtime in an initial value assignment or in an initial workflow step.

Prerequisites
The workflow definition created by the Workflow Wizard can only be executed without postprocessing if the method used by the processing task for displaying or changing the circular does not use parameters but rather only the object reference. Otherwise you must revise the binding definition.

Features
The current recipient receives a work item in their Business Workplace. If they execute the work item, the circular is displayed or can be changed.

You can define a latest end for this work item, after which the workflow initiator is either notified by means of an e-mail or the workflow with the next step is continued.

The recipients of the circular can add attachments and notes using the standard functions in the Business Workplace [Seite 1368] which are available both to the recipients of the subsequent steps and to the workflow initiator.

The number of recipients cannot be changed at runtime. Any changes that need to be made to the agents can be made using the organizational plan (changes to the staffing assignments).

The workflow initiator receives the circular at the end as the final recipient.

Enhancements to the workflow definition
You must amend the workflow definition created by the Workflow Wizard by adding steps, in which the circular is created or determined.

Activities
The system adds the following container elements to the workflow container:

<table>
<thead>
<tr>
<th>Data type</th>
<th>Description (ID)</th>
<th>Explanation</th>
</tr>
</thead>
<tbody>
<tr>
<td>Object reference</td>
<td>Object name from object type definition</td>
<td>Object reference to the circular.</td>
</tr>
<tr>
<td>ABAP Dictionary reference</td>
<td>List of agents (CircularAgents)</td>
<td>Multiline container element in which all recipients are stored.</td>
</tr>
</tbody>
</table>
Dynamic Circular

Use
A user can send a specific circular (for example a Microsoft Word document or any business document) sequentially to a number of users not yet defined at definition time in the R/3 System, who are to view the circular and make comments if applicable.
A dialog box is displayed in which the user who starts the workflow (workflow initiator) can determine the recipient of the circular.
The workflow initiator passes the object for circulation at runtime in an initial value assignment or in an initial workflow step.

Prerequisites
The workflow definition created by the Workflow Wizard can only be executed without postprocessing if the method used by the processing task for displaying or changing the circular does not use parameters but rather only the object reference. Otherwise you must revise the binding definition.

Features
The list of all users authorized to change/display the circular from an organizational point of view is displayed to the workflow initiator. This organizational authorization is derived from the assignment of the possible agents [Extern] for the processing task. The workflow initiator selects the recipient to whom the circular is to be sent. The workflow initiator cannot specify an order.

The object is distributed to the agents using the task Selection of agent for subsequent task (TS30200146) supplied by SAP. This task is classified in automatic Customizing as a general task [Extern].

The current recipient receives a work item in their Business Workplace. If they execute the work item, the circular is displayed or can be changed. For this, a loop (UNTIL) [Extern] is defined in the workflow definition, in which the next recipient is read from the table of all recipients (CircularAgents).
You can define a latest end for this work item, after which the workflow initiator is either notified by means of an e-mail or the workflow with the next step is continued.
The recipients of the circular can add attachments and notes using the standard functions in the Business Workplace [Seite 1368] which are available both to the recipients of the subsequent steps and to the workflow initiator.

The number of recipients cannot be changed at runtime. Any changes that need to be made to the agents can be made using the organizational plan (changes to the staffing assignments).

The condition section of the UNTIL loop checks whether additional recipients are contained in the table. The workflow initiator receives the circular at the end as the final recipient.

Enhancements to the workflow definition
You must amend the workflow definition created by the Workflow Wizard by adding steps, in which the circular is created or determined.

Activities
The system adds the following container elements to the workflow container:
<table>
<thead>
<tr>
<th>Data type</th>
<th>Short description (ID)</th>
<th>Explanation</th>
</tr>
</thead>
<tbody>
<tr>
<td>Object reference</td>
<td><em>Object name</em> from object type definition</td>
<td>Object reference to the circular.</td>
</tr>
<tr>
<td>ABAP Dictionary reference</td>
<td>List of all possible agents (<em>AllAgents</em>)</td>
<td>This container element is no longer required.</td>
</tr>
<tr>
<td></td>
<td>List of agents</td>
<td>Multiline container element in which all recipients are stored.</td>
</tr>
<tr>
<td></td>
<td>Recipient</td>
<td>The user who processes the work item is saved in this container element.</td>
</tr>
<tr>
<td></td>
<td>Index</td>
<td>Index for access to the list of recipients (<em>CircularAgents</em>)</td>
</tr>
</tbody>
</table>
Dynamic Parallel Circular

Use
A user can send a specific circular (for example a Microsoft Word document or any business document) at the same time to a number of users not yet defined at definition time in the R/3 System, who are to view the circular and make comments if applicable.
A dialog box is displayed in which the user who starts the workflow (workflow initiator) can determine the recipient of the circular.
The workflow initiator passes the object for circulation at runtime in an initial value assignment or in an initial workflow step.

Prerequisites
The workflow definition created by the Workflow Wizard can only be executed without postprocessing if the method used by the processing task for displaying or changing the circular does not use parameters but rather only the object reference. Otherwise you must revise the binding definition.

Features
The list of all users authorized to change/display the circular from an organizational point of view is displayed to the workflow initiator. This organizational authorization is derived from the assignment of the possible agents [Extern] for the processing task. The workflow initiator selects the recipient to whom the circular is to be sent. The workflow initiator cannot specify an order.

The object is distributed to the agents using the task Selection of agent for subsequent task (TS30200146) supplied by SAP. This task is classified in automatic Customizing as a general task [Extern].
All recipients receive a work item in their Business Workplace at the same time. If you execute the work item, the circular is displayed or can be changed. The Table-Driven Dynamic Parallel Processing [Extern] is used which contains all the recipients in the table CircularAgents.
You can define a latest end for this work item, after which the workflow initiator is either notified by means of an e-mail or the workflow with the next step is continued.
The recipients of the circular can add attachments and notes using the standard functions in the Business Workplace [Seite 1368] which are available both to the recipients of the subsequent steps and to the workflow initiator.

The number of recipients cannot be changed at runtime. Any changes that need to be made to the agents can be made using the organizational plan (changes to the staffing assignments).
The condition section of the UNTIL loop checks whether additional recipients are contained in the table. The workflow initiator receives the circular at the end as the final recipient.

Enhancements to the workflow definition
You must amend the workflow definition created by the Workflow Wizard by adding steps, in which the circular is created or determined.

Activities
The system adds the following container elements to the workflow container:
<table>
<thead>
<tr>
<th>Data type</th>
<th>Short description (ID)</th>
<th>Explanation</th>
</tr>
</thead>
<tbody>
<tr>
<td>Object reference</td>
<td>Object name from object type definition</td>
<td>Object reference to the circular.</td>
</tr>
<tr>
<td>ABAP Dictionary</td>
<td>List of all possible agents (AllAgents)</td>
<td>This container element is no longer required.</td>
</tr>
<tr>
<td>reference</td>
<td>List of agents (CircularAgents)</td>
<td>Multiline container element in which all recipients are stored.</td>
</tr>
<tr>
<td></td>
<td>Recipient (Current_Agent)</td>
<td>The user who processes the work item is saved in this container element.</td>
</tr>
<tr>
<td></td>
<td>Index (Index)</td>
<td>Index for access to the list of recipients (CircularAgents)</td>
</tr>
</tbody>
</table>
Parallel Circular

Use

A user can send a specific circular (for example a Microsoft Word document or any business document) at the same time to several defined users in the R/3 System, who are to view the circular and make comments if applicable.

In workflows defined with this wizard, you specify all the recipients of the circular in the workflow definition.

If you do not want to define the addressees of the circular until runtime, you must use one of the Workflow Wizards for dynamic circulars.

A workflow definition created with this Workflow Wizard contains the readers of the circular as a defined "distribution list". It is better if this only contains positions, but it can also contain users or organizational units.

The workflow initiator transfers the object for circulation at runtime in an initial value assignment or in an initial workflow step, and leaves the rest to the workflow.

Prerequisites

The workflow definition created by the Workflow Wizard can only be executed without postprocessing if the method used by the processing task for displaying or changing the circular does not use parameters but rather only the object reference. Otherwise you must revise the binding definition.

Features

All recipients receive a work item in their Business Workplace at the same time. If you execute the work item, the circular is displayed or can be changed. For this, a fork [Extern] is used.

You can define a latest end for this work item, after which the workflow initiator is either notified by means of an e-mail or the workflow with the next step is continued.

The recipients of the circular can add attachments and notes using the standard functions in the Business Workplace [Seite 1368] which are available both to the recipients of the subsequent steps and to the workflow initiator.

The number of recipients cannot be changed at runtime. Any changes that need to be made to the agents can be made using the organizational plan (changes to the staffing assignments).

The workflow initiator receives the circular at the end as the final recipient.

Enhancements to the workflow definition

You must amend the workflow definition created by the Workflow Wizard by adding steps, in which the circular is created or determined.

Activities

The system adds the following container elements to the workflow container:

<table>
<thead>
<tr>
<th>Data type</th>
<th>Short description (ID)</th>
<th>Explanation</th>
</tr>
</thead>
<tbody>
<tr>
<td>Object reference</td>
<td><em>Object name</em> from object type definition</td>
<td>Object reference to the circular.</td>
</tr>
<tr>
<td>ABAP Dictionary reference</td>
<td>List of agents (CircularAgents)</td>
<td>Multiline container element in which all recipients are stored.</td>
</tr>
</tbody>
</table>
Other Workflow Wizards

Use
This folder contains Workflow Wizards which help to simplify workflow modeling.

Features
The following Workflow Wizards are available for simplified workflow modeling:

- [Creation of Customizing workflows](Seite 1000)
- [Simplified workflow definition](Seite 1001)
Creation of Customizing Workflows

Use
Customizing activities usually require a number of views or view clusters to be maintained in a particular order. So that view maintenance does not have to be called for each view or view cluster, all views/view clusters that are to be maintained together in one context should be processed within one workflow definition.

Features
To create the workflow definition, the Workflow Wizard asks for the views to be maintained. In the workflow definition, an object reference is first created from the view name to an object of type VIEWDATA for each view to be processed. This object reference is stored in the workflow container in a container element with the same name as the view. A step for maintaining this view is then created. The individual maintenance dialogs are processed sequentially. The Customizing navigator provides information between the individual maintenance dialogs on the views that have already been maintained or are still to be maintained.
Simplified Workflow Definition

Use
In many cases, workflow definitions must be created in which only a single object is processed in some form in several sequential or parallel steps. The possible operations on this object are available in the form of methods and the single-step tasks derived from them.

Prerequisites
To use an object type in a simplified workflow definition without encountering problems, you should make sure during definition of the object types that the methods only require the object reference and not additional method parameters, since these cannot always be created using automatic binding.

Features
The Workflow Wizard first requests the object type on whose object the operations of the workflow definition are to be performed. The system automatically creates a container element with an object reference to the requested type in the workflow container.
Then you choose the tasks that are to be executed in workflow. Agent selection can be performed. The system caters for all appropriate object references of the tasks in the binding. All other input and output parameters required and container elements in the workflow container are created by the system.

You must check the generated workflow definition in the Workflow Builder.
The tasks are arranged in sequential steps as standard. It is possible to group individual steps in a fork. The forks cannot be nested in each other. Other modeling elements, such as loops, are not supported.

Activities
All tasks of the selected object type are displayed in the left-hand part of the screen. Using the pushbuttons Add and Add all, the selected tasks can be transferred to the field on the right-hand side.
Cut and Paste can be used to define the sequence in which the single-step tasks are to be executed.
Several tasks can be grouped together in a fork with Select block. Identical numbers then indicate this association.
The pushbuttons Agent assignment and Task details take you to task processing.
Workflow Builder

Use
This is the main SAP Business Workflow tool for creating, displaying and processing workflow definitions [Extern]. You can test workflow definitions and generate operable versions.

Integration
The Workflow Builder provides a graphical view of the workflow definition. The alphanumeric workflow editor provides a textual view of the workflow definition in a tree display, as an alternative to the Workflow Builder.

Prerequisites
Each workflow definition has one multistep task [Seite 1194]. While the definition of the multistep task regulates the external relationship (call interface, triggering events), the workflow definition describes the actual implementation of a business scenario as a sequence of connected steps.

Features
The steps [Seite 1005], outcomes [Extern] and triggering events [Extern] of the workflow definition are displayed graphically with their interconnections. You can choose the type of display in the Workflow Builder options [Seite 1016].

The Workflow Builder screen is divided into the following areas.

- **Workflow**
  Here you can insert new steps into the workflow definition and process existing ones.
  Double-clicking on a step calls the associated step definition.
Overview

The overview graphic is displayed here. The part of the workflow graphic displayed in the screen area *Workflow* is marked with a green rectangle. Changing the size or position of the rectangle changes the display in the screen area *Workflow*.

Step types

All step types that you can use in the workflow are displayed here if you are in change mode. To insert a new step into the workflow, you click on the step type and insert it into the workflow definition.

Tasks

This screen area (which you can deactivate) can be used to display tasks in order to use them in the workflow definition. A displayed task can be inserted into your workflow definition as an activity by double-clicking. The binding between the task container and the workflow container as well as the other container elements required is generated automatically by the workflow system as far as possible.

Objects

The objects of the workflow are displayed here depending on the display of the workflow definition. The container elements of the workflow container and the document templates of the workflow are also displayed. You can change an object or create new ones directly from the list. A where-used list is available for document templates and container elements.

Messages

All messages generated in where-used lists and workflow tests are displayed here. Double-clicking on a message takes you to the relevant step definition.

For further information, refer to *Workflow Builder Functions*.
Calling the Workflow Builder

Procedure

You can call the Workflow Builder in two ways:

- **Direct start**: Choose **Tools → Business Workflow → Development → Definition tools → Workflow Builder**
- **Start from task**: From a screen for processing or displaying a multistep task

**Direct start**

This loads the workflow definition you last saved. If you want to switch between change mode and display mode, choose 📊. To load another version of the workflow definition or a new workflow definition, you have the following options:

- If the workflow definition is displayed in the task area in the folder *Workflows last processed*, the version number of the workflow definition is also specified in parentheses. To load it, select the multistep task and choose *Load workflow definition* in the context menu.

- If the multistep task is displayed in a different folder in your task area, you can select the version that you want to load in the dialog box *Open another workflow definition*. Select the multistep task and choose *Load workflow definition* in the context menu.

- If you want to load another workflow definition, select 📊. In the dialog box *Open another workflow definition*, you can select the workflow and version that you want to load.

- If you want to create a new workflow definition, choose **Workflow → New**. You make specifications for the multistep task when the workflow definition is first saved.

**Start from task**

If you start the Workflow Builder from a screen for processing or displaying a multistep task, choose **Workflow Builder**. The active version of the workflow definition is loaded. If there is no active version, the version with the highest version number is loaded.

To call another version of the workflow definition, choose **Goto → Workflow Builder → Choose version**. This takes you to a dialog box in which you can choose a version of the workflow definition. For further information, refer to [Versions of a Workflow Definition](Seite_1019).

**Result**

The workflow definition is displayed and can be processed as applicable. If the Workflow Builder is being called to process a newly-created, initial workflow definition for the first time, the following representation is displayed:

- The start of the workflow definition is indicated by 🔄 **Start workflow**. If the new workflow definition was called from the definition of a multistep task in which *triggering events [Extern]* are defined, these are represented by the symbol 🔄 and their description.

- The end of the workflow definition is indicated by 🔄 **Workflow terminated**.

- The area in which the new workflow definition can be inserted is indicated by an undefined step with an outcome. Steps are represented by symbols. The name of an outcome is represented in the standard view on the arrow.
### Step Types and Their Symbols

<table>
<thead>
<tr>
<th>Symbol</th>
<th>Step type</th>
<th>Runtime function</th>
</tr>
</thead>
<tbody>
<tr>
<td>🔄</td>
<td>Process control</td>
<td>The functions <em>Cancel work item</em> or <em>Set work item to obsolete</em> are applied to other work items of the current workflow.</td>
</tr>
<tr>
<td>🔄</td>
<td>Activity</td>
<td>Execution of a task. This task can be a single-step task or a multistep task.</td>
</tr>
<tr>
<td>🔄</td>
<td>Condition</td>
<td>One of two branches defined in the workflow definition is processed. The system makes the decision based on defined conditions. When doing this, the system takes account of information from the context of the workflow or the application objects processed.</td>
</tr>
<tr>
<td>🔄</td>
<td>User decision</td>
<td>The process flow of the current workflow is controlled on the basis of a decision made by the current agent.</td>
</tr>
<tr>
<td>🔄</td>
<td>Document from template</td>
<td>A document is created from a document template.</td>
</tr>
<tr>
<td>🔄</td>
<td>Container operation</td>
<td>Elementary arithmetic operations or value assignments are applied to individual elements of the workflow container.</td>
</tr>
<tr>
<td>🔄</td>
<td>Event creator</td>
<td>An event is created.</td>
</tr>
<tr>
<td>🔄</td>
<td>Multiple condition</td>
<td>One of several branches defined in the workflow definition is processed. To do this, the system checks the value of a container element in the workflow container.</td>
</tr>
<tr>
<td>🔄</td>
<td>Fork</td>
<td>The processing that follows takes place in parallel branches. You can define how many branches have to be processed for the fork to be successfully completed or define an end condition in the condition editor.</td>
</tr>
<tr>
<td>🔄</td>
<td>Undefined step</td>
<td>No function at runtime. Undefined steps always have an outcome.</td>
</tr>
<tr>
<td>🔄</td>
<td>Loop (UNTIL)</td>
<td>A sequence of steps is processed repeatedly until the defined termination condition occurs.</td>
</tr>
<tr>
<td>🔄</td>
<td>Wait for event</td>
<td>The system waits for a specific event. The work item is only completed if the expected event occurs.</td>
</tr>
<tr>
<td>🔄</td>
<td>Loop (WHILE)</td>
<td>A sequence of steps is processed repeatedly as long as the defined comparisons apply. The system leaves the loop when the basis of comparison does not agree with any of the comparison values.</td>
</tr>
</tbody>
</table>
Workflow Builder Functions

Use
You need these functions for the following activities:

- Adding, reassigning or removing steps in a workflow definition
- Processing Steps
- Checking, testing or printing the workflow definition
- Processing another workflow definition
- Processing objects in the object area

Features
If not otherwise specified, you will find these entries in the context menu of a selected node.

Block operations: Cut, copy, paste, delete
The functions always refer to the complete block in which the selected node is located, including all subordinate blocks in the hierarchy. Therefore, always consider the scope of a block. Loops and forks, in particular, can cover large sections of your workflow definition. If necessary, choose Graphic → Blocks → Show to see which steps are in the block.

⚠️ You can only change or delete an outcome in the step definition of the preceding step. If you choose Change for an outcome, only the name is changed.

Functions for changing, creating and displaying a step
If you choose Change or Display, you branch to the step definition.
If you want to change the step type of the step, you must delete the old step and insert a new step of the desired step type. To change an undefined step into another step type, select the undefined step by double-clicking and choose the new step type.
For further information, refer to Creating Steps [Seite 1008].

Inserting Multistep Tasks
The workflow definition of the chosen multistep task is inserted into the current workflow definition at the selected point. Triggering events of the inserted workflow are not transferred. Choose Edit → Insert multistep task.

Undoing changes
You can undo up to ten changes. Select ⬇️ and in the list the action up to which you want to undo changes.

Restoring changes
You can restore changes that have been undone. Select ⬇️.

Where-used list
The Workflow Builder has two where-used lists:

- You can generate a where-used list for container elements of the workflow container and for document templates. To do this, you select the container element or the document template in the object area and choose Where-used list in the context menu. All nodes in which the
object is used are listed in the message area. From here you can branch to the relevant step definition by double-clicking on a usage.

- You can create a where-used list for tasks listed in the task area. To do this, you select the task and choose Where-used list in the context menu. All workflow definitions in which the task is used within a step definition are displayed.

**Processing objects in the object area**

You can display the object definition for all objects displayed in the object area. If you are in change mode, you can process all objects except the workflow system variables in the workflow container. For further information on processing container elements, refer to Definition of Workflow Container [Seite 1213]. When you create a new document template, the application last used for a document template is started. You can use the function Change document class to start a different application. For further information, refer to Creating Document Templates [Seite 1047].

**Checking the workflow definition**

You select to check the workflow definition. All problems recognized are classified as errors or warnings and are output in the message area. You can process the node in which the error occurs if you double-click on the message. Workflow definitions that contain errors cannot be activated. For further information, refer to Versions of a Workflow Definition [Seite 1019]. You can also start the workflow definition test from the display mode. The active version of the workflow definition is used for the test.

**Testing the workflow definition**

You select to execute the workflow definition on a test basis. This classifies the multistep task temporarily as a general task and saves and activates the workflow definition. If no error occurred, the multistep task is started in dialog. For further information, refer to Start Workflow (Test Environment) [Seite 1452].

**Printing the workflow definition**

For more information, refer to printing workflow definition [Seite 1011].

**Processing another workflow definition**

In the Workflow Builder, you can go directly into the processing of another workflow definition. If the workflow definition that you want to process is displayed in the task area, select it and choose Load workflow definition in the context menu. If it is not displayed, select . You can then choose the desired workflow definition.
Create Step

Use
You create a step in a workflow definition in order to execute tasks or to control the workflow, for example. A step is always created in a **block [Extern]**. This block contains at least the created step and its defined outcomes.

Procedure
You create a new step in the workflow definition by either changing an undefined step or creating a new step between two existing nodes.

**Changing undefined steps**
1. Open the dialog box *Choose Step/Operator* by double-clicking on the undefined step.
2. Choose a step type.

**Creating new steps**
3. Select a step or an outcome in the workflow area. Consider the **positioning rules [Extern]** for new steps.
4. Choose *Create* in the context menu. The dialog box *Choose Step/Operator* opens.
5. Choose a step type.

**Inserting tasks from the task area**
You can insert a task displayed in the task area into the workflow definition as an activity.
3. Select a step or an outcome in the workflow area. Consider the **positioning rules [Extern]** for new steps.
4. In the task area, select the task that you want to insert.
5. In the context menu, choose *Include as an activity*.

If you have selected a workflow definition (multistep task), you can also choose *Include workflow definition*. The steps of the selected workflow definition are then inserted.

**Result**
If you have created a new step, a step of the chosen step type is inserted at the chosen position/ replaces the undefined step. You go directly to the step definition. Undefined steps are also inserted in the workflow definition for some step types.
For further information, refer to **Step Types and Their Symbols [Seite 1005]**.
If you have inserted a task from the task area, an activity which references the chosen task is inserted in the workflow definition.

The binding is created automatically by the system and you can adapt it to suit your requirements.
Special Functions in the Workflow Builder

Use
These functions are for processing a workflow definition in the Workflow Builder. The way the functions described are used depends on the options set [Seite 1016].

Features

Drag and drop
You can use drag and drop to create a new step. To do this, you select the desired step type in the step type area. This moves the cursor to the center of the work area and changes its form. You then choose a step or outcome [Extern], which inserts the new step in accordance with the positioning rules [Extern].

Double-click
Double-clicking on the various steps and outcomes in the workflow area has different effects:

<table>
<thead>
<tr>
<th>Double-click on</th>
<th>Effect</th>
</tr>
</thead>
<tbody>
<tr>
<td>Undefined step or its</td>
<td>A new step is inserted</td>
</tr>
<tr>
<td>outcome</td>
<td></td>
</tr>
<tr>
<td>Any other type of step</td>
<td>The step definition is opened.</td>
</tr>
<tr>
<td>Outcome</td>
<td>The name of the outcome can be edited and, depending on the</td>
</tr>
<tr>
<td></td>
<td>outcome type, additional information about the outcome is displayed.</td>
</tr>
</tbody>
</table>

If you have selected a step or an outcome in the workflow area, double-clicking on a task in the task area will transfer it into the workflow definition as an activity in accordance with the positioning rules [Extern].

Showing blocks
The individual blocks [Extern] that make up the workflow definition can be displayed using various gray scales. To do this, choose Graphic → Blocks → Show.

Nested blocks are represented with darker gray scales. Blocks on the same level have the same gray scale. The blocks each have a heading derived from the name of their first step and displayed in a different color in the upper-left part of the block.

Aligning
This function can be used to reformat the display in the workflow area. To do this, you select Lock.
Inserting Multistep Tasks

Prerequisites
You can insert a workflow definition into the workflow definition currently being processed. You insert a workflow definition (as with creating steps) according to the rules of block orientation.

Procedure
2. Select a step or an outcome. Consider the positioning rules [Extern].
The remaining procedure depends on whether the workflow definition to be inserted is displayed in the task area.

Variant A: Workflow definition displayed in task area:
4. Select the workflow definition to be inserted in the task area.
5. In the context menu, choose Include workflow definition.

Variant B: Workflow definition not displayed in task area:
5. Choose Edit → Insert multistep task. The dialog box Insert workflow definition in workflow definition is opened.
6. Choose the multistep task whose workflow definition you want to insert.
7. Choose the version of this workflow definition.

If the workflow definition to be inserted has container elements in the workflow container, which exist under the same name (but maybe a different type) in the workflow definition currently being processed, you are given the opportunity to rename these container elements. If you rename the container element, all the references throughout the workflow definition are also renamed.

Result
The workflow definition inserted has now become an indistinguishable part of the original workflow definition.

If you want to insert a workflow definition that is to remain recognizable as an independent unit, you must insert an activity [Seite 1026] with reference to a multistep task into your workflow definition.
Printing Workflow Definitions

Printing structures
Select \( \text{\textbullet} \) to print the graphical representation of a workflow definition. You can specify the size and arrangement of the printed workflow definition in a dialog box.

If you select \( \text{\textbullet} \) in the alphanumeric workflow editor, the textual structure of the workflow definition is printed. This corresponds to the display in the object area.

Printing details
Choose Workflow → Print → Details to print a list of all the steps involved in the workflow definition with their most important properties. A dialog box WF: Print workflow definition opens, in which you define the scope of the list.
Select \( \text{\textbullet} \) to print the list directly, or \( \text{\textbullet} \) to display the list on the screen first.
Condition Editor

Use
The subsequent flow of a workflow depends at various points on a condition. The evaluation of a condition always returns either the result *true* or the result *false*. You formulate the condition with the condition editor.

<table>
<thead>
<tr>
<th>Step type that uses the condition editor</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Condition</strong> [Seite 1061]</td>
<td>The result of the evaluation decides which of two modeled branches the workflow processes.</td>
</tr>
<tr>
<td><strong>Loop (UNTIL)</strong> [Seite 1087]</td>
<td>The result of the evaluation decides whether the loop is processed again or not.</td>
</tr>
<tr>
<td><strong>Fork</strong> [Seite 1089]</td>
<td>The result of the evaluation decides whether parallel processing is ended, even if the number of necessary branches is not yet reached.</td>
</tr>
</tbody>
</table>

Prerequisites
You are editing a step type that requires a condition. You start the condition editor by double-clicking in the condition table.

Features
There are two modes for working with the condition editor, which differ only in the display and the way you select expressions and operators. To change modes, you select the button. The following modes are available for display:

- **Direct text entry**
  You can select the expressions and operators for constructing the condition using the F4 input help.

- **Mouse-oriented**
  The screen of the mouse-oriented condition editor is divided into the following areas:
The current condition is displayed and can be edited here. Every condition has the general form:

\[ [<\text{Not}>] \ <\text{Expression}> \ <\text{Operator}> \ <\text{Expression}> \ [<\text{And/Or}>\ <\text{Not}> \ <\text{Expression}> \ <\text{Operator}> \ <\text{Expression}> \ ...] \]

**Expression**

Here you will find the following input options for expressions.

- System fields
- Container elements of the workflow container
  
  You can also access the fields of the referenced structure or the attributes of the referenced object via a container element.

**Operators**

The operators available are displayed here. When an operator is chosen, it is inserted into the current line of the condition and replaces an operator that is already there.

<table>
<thead>
<tr>
<th>Operator</th>
<th>Meaning</th>
<th>Remarks</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Equals</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Does not equal</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Less than</td>
<td></td>
</tr>
<tr>
<td>Condition Editor</td>
<td></td>
<td></td>
</tr>
<tr>
<td>------------------</td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>≤</strong></td>
<td>Less than or equal to</td>
<td></td>
</tr>
<tr>
<td><strong>&gt;</strong></td>
<td>Greater than</td>
<td></td>
</tr>
<tr>
<td><strong>≥</strong></td>
<td>Greater than or equal to</td>
<td></td>
</tr>
<tr>
<td><strong>⟨⟩</strong></td>
<td>Contains pattern</td>
<td>ordinary characters and wildcard characters can be entered in pattern c2. &quot;*&quot; denotes a character string and &quot;+&quot; denotes a character.</td>
</tr>
<tr>
<td><strong>⟨⟩</strong></td>
<td>Contains no pattern</td>
<td>c1 does not match c2</td>
</tr>
<tr>
<td><strong>CE</strong></td>
<td>Table contains element</td>
<td>c1 contains c2</td>
</tr>
<tr>
<td><strong>NE</strong></td>
<td>Table does not contain element</td>
<td>c1 does not contain c2</td>
</tr>
<tr>
<td><strong>EX</strong></td>
<td>Object/element exists</td>
<td>Element exists in container or referenced element exists - expression 2 is not required</td>
</tr>
<tr>
<td><strong>NX</strong></td>
<td>Object/element does not exist</td>
<td>Element does not exist in container or referenced element does not exist - expression 2 is not required</td>
</tr>
</tbody>
</table>

**Logical operators**

<table>
<thead>
<tr>
<th>Operator</th>
<th>Meaning</th>
<th>Remarks</th>
</tr>
</thead>
<tbody>
<tr>
<td>AND</td>
<td>And</td>
<td></td>
</tr>
<tr>
<td>OR</td>
<td>Or</td>
<td></td>
</tr>
<tr>
<td>NOT</td>
<td>Not</td>
<td></td>
</tr>
</tbody>
</table>

**NOT** is stronger than **AND**, and **AND** is stronger than **OR**.

A condition of the form **A OR B AND C OR D** is equivalent to **A OR (B AND C) OR D**

**Constants**

You can enter a constant in this input field for expression 2. The format of the constant entered depends on expression 1.

**Parentheses**

You can structure the condition by using parentheses and activate the display of the contents of the parentheses selectively.

The mouse-oriented mode requires a 32-bit platform.
Activities

Editing conditions
The description below applies to operation in mouse-oriented mode.
The expression you can enter or change is selected using 📷. To create or edit a condition, you first select
the relevant expression field in the condition by double-clicking and then define the content.
Usually at least one of the two expressions refers to a container element of the workflow container.
If you choose an operator or a logical operator, it is transferred automatically into the line in which the
selection is located. If you delete a condition line, the logical operators AND and OR of the previous line
are also deleted. If you select a NOT operator that already exists in a line again, it is deleted.

Using parentheses and comments
To insert parentheses into an existing condition, you select the condition lines around which parentheses
are to be inserted and choose 📷. You can change the comment for parentheses. Note that this text is not translated.

Expanding or collapsing parentheses
You can display or hide the contents of all or individual pairs of parentheses. The position of the cursor has
the following effects:
- In the input field outside all parentheses: All parentheses are affected
- Inside parentheses or in a line with parentheses: One pair of parentheses is affected
To hide the content of parentheses you select 📷 and to display it 📷.

Syntax check
To check the condition for syntax errors, you select 📷.

Testing conditions
You can test your condition at definition time.

Test data
You create the test data required as follows:
- Use real data from a work item container. To do this, choose Retrieve test data for work item
  in the context menu.
- Enter the values directly. To do this, choose Enter test data directly in the context menu.

Testing
Choose Evaluate condition with test data in the context menu.
After the test you can use Test analysis to see how the test result arose, and Display test messages
to display the test messages again.
Workflow Definition Extras

Features

The following functions are located in the *Extras* menu in the Workflow Builder:

- **Workflows of this definition**
  
  The system calls the work item selection [Seite 1490]. The current multistep task is preset. Define the search period.

- **Business Workplace** [Seite 1368]

- **Business Object Builder** [Seite 1100]

- **Create event** [Seite 1365]

- **Technical information**

  The technical information contains the table entries with the definition and runtime structures on which the workflow definition is based.

  If you contact SAP about any errors that may have occurred, you should have this information to hand.

- **Expand subworkflows**

  Subworkflows used in the workflow definition are inserted directly into the workflow definition. If there are naming conflicts between the container elements of the existing workflow container and the container elements to be inserted, the system informs you and gives you the opportunity to rename these container elements.

- **Translation**

  Here you can translate all language-dependent texts that appear in the workflow definition. This applies to:

  - Step and outcome names
  - Decision texts and titles
  - Container element names

- **Ad hoc functions**

  You can use this to change the workflow type to *ad hoc*. For further information, refer to *Ad Hoc Workflow* [Seite 1095].

- **Special functions**

  These enable you to correct the container structure or block structure if the system points out problems.

- **Call workflow configuration**

  This enables you to display the configuration of the workflow definition that is currently loaded. For further information, refer to *Workflow Configuration* [Seite 1171].

Effects of setting options

The tab page *General* is used to specify whether the tips and tricks are to be displayed when the Workflow Builder is started.
The tab page *Graphic* is used to determine which view is used for the display:

- **Without event-driven process chains**
  
  All steps, triggering events and the start and end of the workflow are displayed with symbols. Outcomes are displayed in text on the connecting lines.

- **Classic event-driven process chains (ClassicEPCs)**
  
  All triggering events, outcomes, steps and operators are displayed. Steps are displayed as green rectangles, triggering events and outcomes as red hexagons and operators as circles. The node number and the node type are displayed with the step name in the graphical symbols. The step type area is not displayed.

- **Event-driven process chains (EPCs)**
  
  All triggering events, outcomes, steps and operators are displayed. Steps are displayed as green rectangles, triggering events and outcomes as red hexagons and operators as circles. The step name is displayed in the graphical symbols.
Definition of Workflows

Purpose
You define workflows:
- If you want to model business processes that include more than one step or more than one agent, and the workflow definitions of the workflow supplied by SAP are not sufficient.
- If you want to extend or modify the workflow supplied by SAP.

The process description below assumes that you are creating a completely new workflow definition. Logically, it applies if you are processing or displaying an existing workflow definition.

If you only want to change the responsible agents [Extern] and deadlines for a SAP workflow definition, then you can extend the workflow definition [Seite 1171].

Prerequisites
Before defining a workflow, you must do the following:
- Analyze the content of the process to be modeled
- Check whether the objects involved and the methods they require are defined

The tasks you want to use in the steps of the workflow definition do not necessarily have to be defined beforehand. You can also define these tasks from the workflow definition.

Process Flow
4. Start the Workflow Builder. You can start with processing of the workflow definition and create the required container element of the workflow container in the container area. When saving for the first time you determine the abbreviation and the name.

   For further information, refer to Calling the Workflow Builder [Seite 1004].

5. You check, save and activate to conclude your work.

   For further information, refer to Workflow Builder Functions [Seite 1006].

6. If you require the triggering events [Extern], switch to processing of the relevant Multistep task [Extern]. Here you define this including the binding definition.

   For further information, refer to Definition of a Multistep Task [Seite 1194].
Versions of a Workflow Definition

Use
The system administers several versions of one workflow definition within a validity period for a multistep task. One of the versions of a workflow definition is its active version.

The import and export parameters of the workflow container are not subject to any versioning.

To check which version of the workflow definition you are processing, select in the Workflow Builder. The version number is displayed on the tab page Control. An overview of all versions of the workflow definition can be found on the tab page Overview. The status of the workflow definition is displayed in the header.

A workflow always refers to the version of the workflow definition active at the time it is started.

Features

Calling the Workflow Builder
When the Workflow Builder is called, the active version is displayed as standard. If there is no active version, the version with the highest version number is displayed. For further information, refer to Calling the Workflow Builder [Seite 1004].

Saving and activating workflow definitions
When a workflow definition is saved, the old version is overwritten automatically. After being saved, the workflow definition has the status revised.

To set the workflow definition to status active, you select . The workflow definition is first subjected to a syntax check. Any errors and warnings discovered are displayed in the message area of the Workflow Builder. Only if errors are found, is the workflow definition not activated. If no errors are found (no red traffic lights in the message area), the workflow definition is saved and activated.

If you save and activate a workflow definition, current workflows that refer to this version of the workflow definition may no longer be executable.

Generating a runtime version of a workflow definition
You can generate a runtime version without activating the workflow definition. To do this, you choose Workflow → Generate runtime version.

New version of a workflow definition
To generate a new version, you choose Workflow → Generate version. The workflow definition is set to status new, saved.

In order not to increase the memory space requirements for workflow definitions excessively, you should only generate a new version under the following circumstances:
If you have made incompatible changes
If there are productive workflows running that refer to the current version.
As long as you are still in the test and development phase, you do not usually have to create new versions.

### Transporting a workflow definition

If a workflow definition is transported into another system, only the active version is transported. In the target system, an existing workflow definition with the same version number is overwritten if there are no current workflows for it. Otherwise, the transported workflow definition is saved with a free version number. The transported workflow definition becomes the active workflow definition in the target system.
Basic Data of a Workflow Definition

Use
You can process the basic data of the workflow definition by selecting in the Workflow Builder. General information about the workflow definition is also displayed here, and you can create and process triggering events, start conditions and start transactions.

Features
Tab page Start
In the area Workflow start by triggering events..., you can define and remove triggering events for the workflow, as well as create and process start conditions for the triggering events. To define a triggering event, enter the relevant object type and the event into the appropriate fields in a line. You can use the F4 input help. You can activate a defined triggering event and process the binding from the event container to the workflow container.

You can process the properties of a triggering event by selecting it and choosing .

You can create start conditions for each triggering event defined. The workflow then only starts if the triggering event has occurred and one of the start conditions defined for it is fulfilled. Use the condition editor to define the start condition. For further information, refer to Binding Definitions from the Event Container and Start Conditions for a Workflow.

In the area ...and/or start transaction(s), you can create a start transaction for the workflow. This start transaction is executed if you start the workflow manually. There are input fields in the start transaction, in which you can assign data to the import parameters of the interface. You can process the layout of the start transaction generated in the Screen Painter.

Tab page Control
You can make settings for the workflow definition here. If you set the indicator Advance with dialog, you can set the indicator individually for each step definition.
If you do not specify anything for maximum node number or maximum subworkflow level, the Customizing settings apply. In the field Performance optimization, you specify what data is to be logged by the system. The more data you have logged, the worse the system performance is.

If you choose Only log errors, the workflow log does not contain any attachments, objects, forwarding information or information in the area Details.

Tab page Responsibilities
Here you define the system administrator for this workflow. If no entry is made, the entry for the party responsible in the definition of the relevant multistep task is evaluated first, then the relevant SAP Business Workflow Customizing entry if necessary.
You can also define a standard recipient for all missed deadlines. This party is notified if the following apply:

- The workflow definition is used as a subworkflow.
- A monitored deadline with notification is activated in the step definition, but no recipient is entered.

For further information, refer to Defining Responsibilities.
Tab page Notification
Here you define the message recipient for completion of the workflow. This party is notified if the workflow is used as a subworkflow and there is no recipient specified in the task definition. Upon completion of the workflow, the text displayed here is sent as an e-mail.
For further information, refer to Defining Responsibilities [Extern].

Tab page Change data
All entry, change and activation data of the workflow definition is displayed here.

Tab page Technical information
Here you will find general data about the workflow and a version overview for the workflow definition. The version number of the version being processed in the original development system is displayed in the field Original version. If a workflow is transported into another system, in which workflows are already running for this definition, the version numbers are reconciled.
The indicator Locked against instantiation is maintained in the classification of the multistep task.
Using Wizards

Use
You use wizards when you want to simplify the definition of certain steps within a workflow definition.
The number of wizards supplied is increased on a regular basis. Therefore, look in the system to see which wizards are currently available. The wizards are called in the Workflow Builder under Wizards. You can also call the wizard Include "Send mail" from the step type area.

You usually have to process the parts of your workflow definition created by the wizard yourself afterwards. The appropriate explanatory text in the wizard will point this out.

Integration
The wizards only ever create parts of an overall workflow definition.
You can also call the Workflow Wizard Explorer from the Workflow Builder. You can use the workflow wizards available there to extend your workflow definition.
Maintenance of Activities

Purpose
You define or process an activity in the following cases:
- If you want to add a new activity to a workflow definition.
- If you want to change an existing activity with regard to its agent assignment or its deadline monitoring.
- If you want to analyze a workflow definition.

In the workflow definition an activity is represented with the symbol [Extern].

Prerequisites
One of the following prerequisites must be fulfilled when you create a new activity:
- You know the task (single-step or multistep) that you want to reference in the activity.
- You know an object type and one of its methods that has the required functionality.

During definition of an activity, you can find a defined single-step task via the method. If there are no appropriate single-step tasks, you can define one from the workflow definition.

Process Flow
The maintenance of an activity is spread across several tab pages. You can process the properties of the activity here, for example:
- Task selection
- Data for deadline monitoring [Extern]
- Outcomes
  - Responsible agents [Extern]

The order of the tab pages defines the recommended sequence of processing. You should start the definition of a new activity on the tab page Control, since certain entries are made automatically on other tab pages when the task is entered.

On the tab page Control, you can make all the entries required to define an executable activity in a sequence.

Result
The system inserts a step of the type Activity into the workflow definition. To ensure the consistency of the workflow definition, all active outcomes are inserted with the actual step. For each outcome, the system adds a branch in the workflow definition and puts an undefined step into this branch. To ensure the consistency of the block structure, all branches created are brought together before the next step of the workflow definition.

At runtime, an activity is represented with a work item of one of the following types:
- Dialog work item [Extern] if the activity references a single-step task with dialog.
- Background work item [Extern] if the activity references a single-step task processed by the system without dialog.
Maintenance of Activities

- [Workflow work item [Extern]] if the activity represents a multistep task.
Maintenance of Tab Page Control

Use
With the specifications you make on this tab page, you define:

- Which task is executed in this activity.
- Which properties this step has.
- Who is responsible for processing the activity.

When you define a new activity, you must make entries about the task. This supplements entries on the tab page Outcomes.

Activities
Process the following settings:

Task
Define the task to be executed. An existing task can be selected via:

- Parts of names.
- The application component assigned.
- The method used.

Select if no suitable task exists yet.

If you enter another task in an existing step definition, the outcomes of the original single-step task are not assigned automatically to the outcomes of the newly entered single-step task. You are asked to assign the selected outcomes of the new single-step task to the outcomes of the single-step task originally entered. Superfluous outcomes and the corresponding branches of the Workflow definition are deleted.

The system creates new branches in the workflow definition for additional or unassigned outcomes of the new single-step task.

The properties and descriptive text of the selected task are displayed. However, you can only maintain these within the task definition.

Step properties
You can define the following step properties:

- The task is to be determined with an expression [Extern].

You can determine which task is referenced in this activity dynamically. To do this, you define a task in the activity as usual (basic task). Select and specify the container element containing the alternative task. This task must be identical to the basic task in all interfaces. If there is no result or an error is found when the expression is resolved, the basic task is used. If you use task TS70008069 in a preceding step, the task can be evaluated easily. To do this, you must define a task group containing all tasks that could be executed in this activity. You pass this task group to the method used in task TS70008069. This method provides an expression that contains the selected task and can be used in this activity.

- Processing can be rejected
If the indicator is set, processing can be rejected in the Business Workplace. The workflow definition is extended by the addition of a branch in which you must model the reaction to this rejection.

- Work items for this step appear in the workflow log [Seite 1420].
- The following step is to be executed immediately by the same recipient [Extern]. (For further information, refer to Advancing with Immediate Dialog [Seite 1453].)

**Binding**

You can process the binding definition between task container and workflow container. When you enter a task for the first time, the system proposes the following:

- New container elements for the workflow container.
- A binding definition.

You decide whether you want to accept the proposal in full or in part. To process the binding later, select Binding. For further information, refer to Rule-Based Proposal for Binding Definitions [Seite 1221].

Even if you reject the proposal at first, you can define the binding automatically at a later date. To do this, select.

**Agents**

You process the responsible [Extern] agents and the excluded [Extern] agents for this step if you are referencing a dialog task. These specifications only apply to this step and complement the specifications on organizational responsibility for the task. For further information, refer to Defining Responsibilities [Extern].
Maintenance of Tab Page Outcomes

Use

The specifications you make on this tab page determine the outcomes of the step for which you want to model a reaction. If reactions to missed deadlines are modeled, the outcomes defined for the relevant missed deadline are only displayed here.

Features

The system determines the outcomes that are displayed on this tab page from the following specifications:

- Possible values of the result [Extern] of the method used in the task.
- Terminating events [Extern] of the task.
- Exceptions [Extern] for the method used in the task.
- Step definition.

You decide which outcomes you want to take into account in the workflow when modeling. The only outcomes that have to be taken into account whenever they are displayed on this tab page are the outcomes for missed deadlines Task executed synchronously and Processing rejected.

If you have not included an outcome that is required during execution into the definition, the associated workflow work item is forced into status error.

All possible outcomes are listed in the following table:

<table>
<thead>
<tr>
<th>Outcome</th>
<th>This outcome exists if ...</th>
<th>Notes and comments</th>
</tr>
</thead>
<tbody>
<tr>
<td>Terminating event of task</td>
<td>The task was defined with terminating events.</td>
<td>If the underlying method is an asynchronous method, you must select at least one event as an outcome.</td>
</tr>
<tr>
<td>Possible value of result of object method</td>
<td>The synchronous object method is defined with a result, for which fixed values or a check table are maintained.</td>
<td>If you deactivate all the values of the result, the system activates the outcome Task executed synchronously.</td>
</tr>
<tr>
<td>Object method exception</td>
<td>The object method is defined with exceptions.</td>
<td>Exceptions in the Workflow Definition [Extern]</td>
</tr>
</tbody>
</table>
| Task executed synchronously                  | • The activity refers to a synchronous object method without result.  
  • The activity refers to a synchronous object method with result, but no result is selected. | If this outcome is displayed, it is always active.       |
| Processing rejected                          | The indicator Processing can be rejected is set.               | If processing of the relevant work item is rejected at runtime (via Reject execution in the Business Workplace), the steps defined after this outcome are executed. |
| Processing obsolete                          | The work item can be set to obsolete via a step of the type.   | The steps defined after this outcome are executed.      |
Maintenance of Tab Page Outcomes

<table>
<thead>
<tr>
<th>Process control</th>
<th>This outcome should only be activated if the modeled reaction to a missed deadline contains a step that sets the work item for this step to obsolete.</th>
</tr>
</thead>
<tbody>
<tr>
<td>Requested end</td>
<td>The relevant deadline monitoring is activated and a modeled reaction required.</td>
</tr>
<tr>
<td>Latest end</td>
<td></td>
</tr>
<tr>
<td>Latest start</td>
<td>Within these branches, you can model a process control step that sets the work item of the step to obsolete. However, you cannot deactivate these outcomes.</td>
</tr>
</tbody>
</table>

At runtime, the system enters an event receiver linkage only for those terminating events that you select in the definition of an activity and therefore include in the workflow definition.

No other events terminate the activity, even if they are entered as terminating events for the single-step task.

**Activities**

You activate the outcomes that you want to actually transfer into the workflow definition. Select the relevant line. Active outcomes are marked with a green dot and have a modeling branch in the workflow definition.

If you deactivate an active outcome, all steps already modeled in this branch are deleted.

For further information about an outcome, select it and choose Detail in the context menu.
Maintenance of Tab Page Notification

Use
The specifications you make on this tab page define the notification agent for completion. This entry is optional.
These notification agents receive a notification of completion when the relevant work item is set to status completed.

Activities
Enter a message recipient for completion. For further information, refer to Defining Responsibilities.
If no recipient is entered, no notification of completion is sent. The notification of completion contains the completion text that can only be processed in the task definition, and a reference to the work item display of the completed work item.
Maintenance of Deadline Tab Pages

Use
You can define all the deadlines for this step with the specifications you make on these tab pages. All these entries are optional.

Features
You can set the following deadlines on the corresponding tab pages:

- Requested start [Extern]
- Latest start [Extern]
- Requested end [Extern]
- Latest end [Extern]

You can define reactions to a missed deadline for the last three deadline types. The system offers the following possible reactions:

- Notification
- Free modeling

Activated deadlines are marked with in the tab page index.

You define deadlines with respect to a reference date/time. The system offers the following reference date/times:

- The creation date/time of the work item.
- The creation date/time of the workflow to which the monitored work item belongs.
- A date in the form of an expression, which is derived from the context of the application during execution of the workflow.

Activities
You activate monitoring of the relevant deadline by selecting a reference date/time for the deadline.

If you choose expression, you have to define the reference date/time by specifying expressions [Extern] for the date and time yourself. Use the F4 input help for entering expressions.

The value referenced using the expression must be of data type D for the date and data type T for the time. If you specify a date but no time, the system sets the time to 00:00:01 (requested and latest start) or 23:59:59 (requested and latest end).

Process the deadline by entering a duration and an appropriate unit. Negative durations only apply if you define the reference date/time yourself via an expression.

The reaction to a deadline being reached depends on the type of deadline:

**Requested start**
If the work item is created before the requested start, it is assigned the status waiting. When the requested start is reached, the system sets the work item to status ready. If the work item is created after the requested start, it is assigned the status ready immediately.

**Latest end, latest start, requested end**
You can either notify someone or model a reaction in the workflow definition for when the deadline is reached.
- Display text (standard escalation)
  
  If you want to notify someone, you must specify a recipient [Extern] on the tab page Display text. The message text is displayed in the display area. It can only be changed in the task definition.

- Modeled
  
  If you want to model your own reaction, enter a name for the new outcome on the tab page Modeled. A reaction branch is added to the workflow definition, which is processed if the deadline is missed.

  If a deadline is missed, the original step is not yet completed. The steps following this "missed deadline event" do not end the original activity.

  For further information, refer to Modeled Deadline Monitoring or Standard Escalation [Extern].
Maintenance of Tab Page Methods

Use

In addition to the task's method, you can define the following methods:

- **Before method [Extern]**: Executed before execution of the method.
- **Secondary method [Extern]**: Executed at the same time as the method.
- **After method [Extern]**: Executed after execution of the method.

Prerequisites

The methods and their object types are defined in the *Business Object Repository*. The objects on which the methods are executed must be referenced in a container element in the workflow container.

⚠️ If you define one of these methods for a user decision, you must delete the entry in the field *Function module* on the tab page *Work item display*.

Features

Calling before and after methods

When executing a work item with before or after methods, the system first calls the before methods synchronously in any order and processes them. Then the system starts the task method and secondary methods in different sessions. Secondary methods are only for display. Any changes carried out in them have no (direct) effect on the workflow. Finally, the system calls the after methods synchronously in any order and processes them.

The after methods being called is not dependent on the result of the task method. They are called even if the task method terminated with an error.

The after methods being called is not dependent on whether or not the task method is terminated with an event. They are called immediately after the synchronous part of the method.

With work items that are set to status *completed* by a terminating event without method execution, the before, secondary and after methods are not called. Before methods cannot create the object on which the task method is to run. However, after methods can run on a object created by the task method.

Data transfer into and out of the methods

There is no separate binding between task container and method container for these methods. The methods are given the method container of the task method. If no binding is defined between the task and the method, the system passes the task container. Secondary methods cannot return data to the workflow nor influence the process control. Before and after methods can evaluate and manipulate the method container in order to influence the subsequent method or steps.
Error handling
As with secondary methods, errors in before and after methods have no influence on the execution of a work item. The processing logic of the work item depends solely on the task method. The system records errors in the before and after methods as warnings in the workflow log.

Activities
To define an object method as a before, an after or a secondary method, select (in the column Object) a container element in the workflow container, which contains the object to be processed. Then select the object method required in the column Method.
Maintenance of Tab Page Work Item Display

Use
On the tab page *Work item display*, you can individualize the work item display with an additional tab page, and the work item preview with graphics or text.

Features

Step-specific tab page for work item display
An individual tab page is added to the work item display. This tab page is displayed in addition to the three tab pages *Basic data*, *Activities* and *Available objects* as the first tab page of the work item display. This additional tab page only appears when work items that represent this step are displayed. This tab page can be used, for example, to:
- Display important information from the current process, which does not appear in the work item display as standard.
- Provide functions that the end user has to execute often.
- Create a customer-specific 'look and feel'.

Adapting the work item preview in the Business Workplace
A user exit is available for individualizing the work item preview. For further information, refer to the User Exit for Work Item Preview [Extern].

A user decision can be executed in the work item preview as standard. The function module required for this is entered automatically.

If secondary methods are to be executed or methods before or after the work item execution in a user decision, delete the entry in the field *Function module*.

Activities

Step-specific tab page for work item display
Enter the program and the number of the screen into the relevant input fields. For further information, refer to Programming a Step-Specific Tab Page [Extern].

Adapting the work item preview in the Business Workplace
Enter the function module that you want to use in the input field *Function module*. You can create a function module using a template by selecting 📝.
Maintenance of Tab Page Other

Use
On the tab page Other, you can change the priority and repeated execution of the work item.

Features

Step priority
Here you define what priority the work items created for this step have. Dialog work items of priority 1 are displayed in the Business Workplace as express work items.

Table-Driven Dynamic Parallel Processing
You can process this step several times in parallel. The multiline container element entered here controls this parallel processing. For further information, refer to Table-Driven Dynamic Parallel Processing [Extern].

Repeat counter for work items with temporary errors
Maximum number of attempts made by the system to restart a background work item [Extern] with temporary errors. If all of the attempts are unsuccessful, the work item status is set to:

- **Error** (if the temporary exception is not modeled)
- **Completed** (if the temporary exception is modeled)

This function is only available for steps that refer to a single-step task without dialog ("background task"). If the repeat counter has the value 0, the maximum number of retries is determined from the settings in Customizing [Extern].

For further information, refer to Error Handling for Background Work Items [Seite 1388].
Maintenance of a Web Activity

Purpose
You use this step type to send an XML document from the workflow to another system. The recipient can process the XML document according to their requirements. This XML document can start a workflow in another system for example.
If the receiver is also a SAP System, a service [Extern] is available, which can be used to start a Business Workflow automatically.
In the workflow definition such a step type is represented with the symbol [Ex].

Prerequisites
The SAP System must be appropriately configured for a workflow containing this step type to be executed. For further information, refer to Defining the Web Server [Seite 1513].

Process Flow
The maintenance is spread across several tab pages. You can process properties on them, such as:
- Call options [Seite 1038]
- Deadline monitoring [Extern]
The order of the tab pages defines the recommended sequence of processing. To create an executable Web activity, you only have to process the tab page Call.

Result
The system inserts a step of the type Web activity into the workflow definition. To ensure the consistency of the workflow definition, all active outcomes are inserted with the actual step. For each outcome, the system adds a branch in the workflow definition and puts an undefined step into this branch. To ensure the consistency of the block structure, all branches created are brought together before the next step of the workflow definition.
Maintenance of Tab Page Call

Use
With the specifications you make on this tab page, you define:

- Which interface task is used.
- The address to which the XML document is sent.
- Which transfer format is used.

Activities
Process the following settings:

Reference workflow
Specify a workflow whose interface is in accordance with the interface of the workflow to be called. For all parameters of the interface, a container element with the same name is created in the workflow container if none exists as yet. The system defines the binding required automatically.

If the XML document is sent to an SAP System and a Business Workflow is to be started there, the interface of the reference workflow must be identical to that of the workflow to be started. In particular, the same names must be used.

You should only use single-line parameters in the interface, which use the ABAP Dictionary or the object type XML_DOC as data type reference.

Step name
Give the step a name. This name is used to label the step within your workflow.

Transfer format
Select the transfer format in which the XML document is to be sent. The following transfer formats are currently supported:

- WF-XML (draft version) - number 01
  The system creates an XML document in accordance with the transfer format WF-XML of the Workflow Management Coalition. All import parameters of the interface are transferred into this XML document. For further information on creating a WF-XML document, refer to Creation of a WF-XML Document [Seite 1093].

- Send XML data without envelope - number 02
  You should only use this transfer format if the interface only contains one import parameter with the data type reference to the object type XML_DOC. If the interface contains an export parameter with the data type reference to the object type XML_DOC and the http reply contains an XML document, the XML document is written into the export parameter.

Address (URL)
Enter the address to which the XML document is to be sent in the form of a URL. You can also enter an expression that contains a URL.

If the XML document is sent to an SAP System and a Business Workflow is to be started there, you can use the wizard to define the URL. To do this, select . The URL has the following structure:
In <protocol> you specify the transfer format used in the form of a two-digit number. In <localkey> you specify the ID of the Business Workflow to be started.

When the other SAP System is called, a logon is performed using a predefined user name. This user must be a possible agent [Extern] of the Business Workflow to be started.

You want to start workflow WS00000000 and use transfer format 01. The SAP System on which the workflow is to be started has Web server WEBSERVER01. The entry required in Address (URL) is:

http://webserver01/scripts/wgate/wf_handler/!?~protocol=01&~localkey=WS00000000

Wait for feedback

If you set this indicator, the associated work item is not completed during execution until a reply is received in the form of an XML document. If you chose transfer format 01, the system's required reply URL is included into the XML document. The XML reply document must contain the operation ProcessInstanceStateChanged and set the process status to closed.completed or closed.abnormalCompleted for the workflow to be continued.

Only set this indicator when using transfer format 01.
Maintenance of Tab Page Control

Use
This tab page is used to define the step properties and display the description of the reference workflow.

Activities
You can define whether the execution of the step is logged in the workflow log. The description of the reference workflow is displayed. Click to go directly to the definition of the reference workflow and process the description.
Maintenance of Tab Page Outcomes

Use
The specifications you make on this tab page determine the outcomes of the step for which you want to model a reaction. If reactions to missed deadlines are modeled, the outcomes defined for the relevant missed deadline are only displayed here.

Features
On this tab page you can only activate the outcome Processing obsolete. All other outcomes are determined by the system and activated automatically.

All possible outcomes are listed in the following table:

<table>
<thead>
<tr>
<th>Outcome</th>
<th>This outcome exists if ...</th>
<th>Notes and comments</th>
</tr>
</thead>
<tbody>
<tr>
<td>Step executed</td>
<td>Always</td>
<td>You cannot deactivate this outcome.</td>
</tr>
<tr>
<td>Processing obsolete</td>
<td>The work item can be set to obsolete via a step of the type process control.</td>
<td>The steps defined after this outcome are executed. This outcome should only be activated if the modeled reaction to a missed deadline contains a step that sets the work item for this step to obsolete.</td>
</tr>
<tr>
<td>Requested end</td>
<td>The relevant deadline monitoring is activated and a modeled reaction required.</td>
<td>Within these branches, you can model a process control step that sets the work item of the step to obsolete. You cannot deactivate these outcomes.</td>
</tr>
<tr>
<td>Latest end</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Latest start</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Maintenance of Tab Page Notification

Use
The specifications you make on this tab page define the notification agent for completion. This entry is optional. These notification agents receive a notification of completion when the relevant work item is set to status completed.

Activities
Enter a message recipient for completion. For further information, refer to Defining Responsibilities. If no recipient is entered, no notification of completion is sent. The notification of completion contains the completion text that can only be processed in the task definition, and a reference to the work item display of the completed work item.
Maintenance of Deadline Tab Pages

Use
You can define all the deadlines for this step with the specifications you make on these tab pages. All these entries are optional.

Features
You can set the following deadlines on the corresponding tab pages:
- Requested start
- Latest start
- Requested end
- Latest end

You can define reactions to a missed deadline for the last three deadline types. The system offers the following possible reactions:
- Notification
- Free modeling

Activated deadlines are marked with in the tab page index.

You define deadlines with respect to a reference date/time. The system offers the following reference date/times:
- The creation date/time of the work item.
- The creation date/time of the workflow to which the monitored work item belongs.
- A date in the form of an expression, which is derived from the context of the application during execution of the workflow.

Activities
You activate monitoring of the relevant deadline by selecting a reference date/time for the deadline.
If you choose expression, you have to define the reference date/time by specifying expressions for the date and time yourself. Use the F4 input help for entering expressions.

The value referenced using the expression must be of data type D for the date and data type T for the time. If you specify a date but no time, the system sets the time to 00:00:01 (requested and latest start) or 23:59:59 (requested and latest end).

Process the deadline by entering a duration and an appropriate unit. Negative durations only apply if you define the reference date/time yourself via an expression.

The reaction to a deadline being reached depends on the type of deadline:

Requested start
If the work item is created before the requested start, it is assigned the status waiting. When the requested start is reached, the system sets the work item to status ready. If the work item is created after the requested start, it is assigned the status ready immediately.

Latest end, latest start, requested end
You can either notify someone or model a reaction in the workflow definition for when the deadline is reached.
• Display text (standard escalation)
  
  If you want to notify someone, you must specify a recipient [Extern] on the tab page Display text. The message text is displayed in the display area. It can only be changed in the task definition.

• Modeled
  
  If you want to model your own reaction, enter a name for the new outcome on the tab page Modeled. A reaction branch is added to the workflow definition, which is processed if the deadline is missed.

  If a deadline is missed, the original step is not yet completed. The steps following this "missed deadline event" do not end the original activity.

  For further information, refer to Modeled Deadline Monitoring or Standard Escalation [Extern].
Maintenance of Documents from a Template

Purpose
You use this step type to create workflow documents of various PC applications and process them within the workflow. The work item recipient creates the document based on the document template which can also contain container elements of the workflow container.

In the workflow definition such a step type is represented with the symbol 🆉.

Prerequisites
The PC application for which you want to create a document must be installed on your PC.

Process Flow
The maintenance is spread across several tab pages. You can process the properties here, for example:

- Template selection
- Responsible agents [Extern]
- Deadline monitoring [Extern]

The order of the tab pages defines the recommended sequence of processing.

When a document from template is created, the system enters the SAP document task Create document from template (TS70008298) into the field Task on the tab page Control as standard. You can change this entry at any time.

You should only create your own document task if:

- You want to restrict the possible agents of the task.
- You want to display a different text in the work item preview of the Business Workplace.

The generated document is passed to a container element of the workflow container, from which it is available to other steps.

Result
The system inserts a step of the type document from template into the workflow definition. To ensure the consistency of the workflow definition, all active outcomes are inserted with the actual step. For each outcome, the system adds a branch in the workflow definition and puts an undefined step into this branch.

To ensure the consistency of the block structure, all branches created are brought together before the next step of the workflow definition.

At runtime, this step is represented by a dialog work item [Extern] in the Business Workplaces [Seite 1368] of its recipients [Extern].

When a recipient chooses the work item created for this step for processing in the Business Workplace, the relevant PC application is opened with the chosen template. If several or no templates are defined in the step definition, the recipient can choose the desired template in a dialog box.
Maintenance of Tab Page Document Templates

Use
With the specifications you make on this tab page, you define:

- Which document templates are offered to the user.
- The container element of the workflow container, in which the document is stored.
- Who the responsible agent [Extern] of the step is.

Activities
Make the following settings:

Document template
You must enter the document templates that should relate to the document to be created. If you enter more than one template, the user can choose which template is to be used when the step is executed. If you do not enter any templates, the user is offered all possible templates for selection. If the template required does not exist, select [ ] For further information, refer to Creating Document Templates [Seite 1047].

Element for document
You must choose a container element in the workflow container for the new document. Use the F4 input help. If you enter a container element that does not yet exist, it is created automatically.

Agent determination
You process the responsible [Extern] agents and the excluded [Extern] agents for the document from template. These specifications only apply to this step and complement the specifications on organizational responsibility. For further information, refer to Defining Responsibilities [Extern].

If you only make entries on this tab page, a step of the type Document from template on the basis of the standard task supplied by SAP is transferred into the workflow definition.
Creating Document Templates

Use
Document templates determine the appearance of PC documents created in the workflow. The document created is based on one of the templates available, and can contain container elements of the workflow container or system fields as variables.

Prerequisites
The PC application on which the template is to be based must be installed locally on your PC. In order to display the Document template folder in the object area of the Workflow Builder, the workflow definition must have been saved at least once.

Procedure
1. Open the PC application by double-clicking on the Document templates folder in the object area or select in the step definition of document generation.
   
   The PC application which you last used to create a template is opened in the Workflow Builder. The system fields and the container elements of the workflow container are offered for selection in the object area. They can be transferred into the template by double-clicking. All relationships created between container elements/system fields and the document template are displayed in the task area.

   It is not possible to insert system fields and container elements into all types of document template.

2. If you want to create a document template of a different class, select Change document class and choose the new type of document template.

3. Select to assign a name for the document template.

4. Create the document template in the usual way in your PC application. You can use all the functions of the PC application.

5. Insert container elements and/or system fields from the object area into your template by double-clicking. These fields are replaced with the content of the container elements/system fields at runtime.

6. Select to save your document template.

   You can only save the document template from your PC application as a local copy on your PC.
Maintenance of Tab Page Control

Use
The specifications you make on this tab page enable you to create a document from a template.

Activities
Make the following settings:

Task
A document task must be entered in this field. You should only use another document task if you want to restrict the selection of possible agents [Extern] or display another text in the work item preview. Select to create a new document task on the basis of the standard task.

Step properties
You can define the following step properties:

- The task is to be determined with expression [Extern].
  You can determine which document task is used dynamically. To do this, select and specify the container element that references the alternative task. This task must be a decision task and should have been developed from a copy of the standard document task.

- Processing can be rejected
  If the indicator is set, processing can be rejected in the Business Workplace. The workflow definition is extended by the addition of a branch in which you must model the reaction to this rejection.

- Work items for this step appear in the workflow log [Seite 1420].

- The following step is to be executed immediately by the same recipient [Extern] (for further information, refer to Advancing with Immediate Dialog [Seite 1453]).

Binding
Process the binding between task container and workflow container. This is necessary if the agent is to create a document that is based on a template as an attachment, and if text variables are used in the task description. To do this, select Binding (existing). For further information, refer to Binding Definition for Document from Template [Seite 1230].
Maintenance of Tab Page Outcomes

Use
The specifications you make on this tab page determine the outcomes of the step for which you want to model a reaction. If reactions to missed deadlines are modeled, the outcomes defined for the relevant missed deadline are only displayed here.

Features
On this tab page you can only activate the outcomes *Processing obsolete* and *Document could not be created*. All other outcomes are determined by the system and activated automatically.

All possible outcomes are listed in the following table:

<table>
<thead>
<tr>
<th>Outcome</th>
<th>This outcome exists if ...</th>
<th>Notes and comments</th>
</tr>
</thead>
<tbody>
<tr>
<td>Document could not created</td>
<td>Always</td>
<td>Defined exception for the method of the standard document task.</td>
</tr>
<tr>
<td>Task executed synchronously</td>
<td>Always</td>
<td>You cannot deactivate this outcome.</td>
</tr>
<tr>
<td>Processing rejected</td>
<td>The indicator <em>Processing can be rejected</em> is set.</td>
<td>If processing of the relevant work item is rejected at runtime (via <em>Reject execution</em> in the Business Workplace), the subsequent steps defined after this exception are executed.</td>
</tr>
<tr>
<td>Processing obsolete</td>
<td>The work item can be set to obsolete via a step of the type <em>process control</em>.</td>
<td>The subsequent steps defined after this exception are executed. This outcome should only be activated if the modeled reaction to a missed deadline contains a step that sets the work item for this step to obsolete.</td>
</tr>
<tr>
<td>Requested end</td>
<td>The relevant deadline monitoring is activated and a modeled reaction required.</td>
<td>Within these branches, you can model a <em>process control</em> step that sets the work item of the step to obsolete. You cannot deactivate this outcome.</td>
</tr>
<tr>
<td>Latest end</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Latest start</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Activities
You activate the outcomes that you want to actually transfer into the workflow definition. Select 📐 in the relevant line.

Active outcomes are marked with 📐 and have a modeling branch in the workflow definition.

⚠️
If you deactivate an active outcome, the whole of the associated branch is deleted.

For further information about an outcome, select it and choose *Detail* in the context menu.
Maintenance of Tab Page Notification

Use
The specifications you make on this tab page define the notification agent for completion. This entry is optional. These notification agents receive a notification of completion when the relevant work item is set to status completed.

Activities
Enter a message recipient for completion. For further information, refer to Defining Responsibilities. If no recipient is entered, no notification of completion is sent. The notification of completion contains the completion text that can only be processed in the task definition, and a reference to the work item display of the completed work item.
Maintenance of Deadline Tab Pages

Use
You can define all the deadlines for this step with the specifications you make on these tab pages. All these entries are optional.

Features
You can set the following deadlines on the corresponding tab pages:

- Requested start [Extern]
- Latest start [Extern]
- Requested end [Extern]
- Latest end [Extern]

You can define reactions to a missed deadline for the last three deadline types. The system offers the following possible reactions:

- Notification
- Free modeling

Activated deadlines are marked with a symbol in the tab page index.

You define deadlines with respect to a reference date/time. The system offers the following reference date/times:

- The creation date/time of the work item.
- The creation date/time of the workflow to which the monitored work item belongs.
- A date in the form of an expression, which is derived from the context of the application during execution of the workflow.

Activities
You activate monitoring of the relevant deadline by selecting a reference date/time for the deadline.

If you choose expression, you have to define the reference date/time by specifying expressions [Extern] for the date and time yourself. Use the F4 input help for entering expressions.

The value referenced using the expression must be of data type D for the date and data type T for the time. If you specify a date but no time, the system sets the time to 00:00:01 (requested and latest start) or 23:59:59 (requested and latest end).

Process the deadline by entering a duration and an appropriate unit. Negative durations only apply if you define the reference date/time yourself via an expression.

The reaction to a deadline being reached depends on the type of deadline:

**Requested start**
If the work item is created before the requested start, it is assigned the status waiting. When the requested start is reached, the system sets the work item to status ready. If the work item is created after the requested start, it is assigned the status ready immediately.

**Latest end, latest start, requested end**
You can either notify someone or model a reaction in the workflow definition for when the deadline is reached.
• Display text (standard escalation)
  
  If you want to notify someone, you must specify a recipient [Extern] on the tab page Display text. The message text is displayed in the display area. It can only be changed in the task definition.

• Modeled
  
  If you want to model your own reaction, enter a name for the new outcome on the tab page Modeled. A reaction branch is added to the workflow definition, which is processed if the deadline is missed.

  If a deadline is missed, the original step is not yet completed. The steps following this "missed deadline event" do not end the original activity.

  For further information, refer to Modeled Deadline Monitoring or Standard Escalation [Extern].
Maintenance of Tab Page Methods

Use
In addition to the task's method, you can define the following methods:

- **Before method [Extern]**: Executed before execution of the method.
- **Secondary method [Extern]**: Executed at the same time as the method.
- **After method [Extern]**: Executed after execution of the method.

Prerequisites
The methods and their object types are defined in the *Business Object Repository*. The objects on which the methods are executed must be referenced in a container element in the workflow container.

⚠️ If you define one of these methods for a user decision, you must delete the entry in the field *Function module* on the tab page *Work item display*.

Features

Calling before and after methods
When executing a work item with before or after methods, the system first calls the before methods synchronously in any order and processes them. Then the system starts the task method and secondary methods in different sessions. Secondary methods are only for display. Any changes carried out in them have no (direct) effect on the workflow. Finally, the system calls the after methods synchronously in any order and processes them.

The after methods being called is not dependent on the result of the task method. They are called even if the task method terminated with an error.

The after methods being called is not dependent on whether or not the task method is terminated with an event. They are called immediately after the synchronous part of the method.

With work items that are set to status *completed* by a terminating event without method execution, the before, secondary and after methods are not called. Before methods cannot create the object on which the task method is to run. However, after methods can run on a object created by the task method.

Data transfer into and out of the methods
There is no separate binding between task container and method container for these methods. The methods are given the method container of the task method. If no binding is defined between the task and the method, the system passes the task container. Secondary methods cannot return data to the workflow nor influence the process control. Before and after methods can evaluate and manipulate the method container in order to influence the subsequent method or steps.
**Error handling**
As with secondary methods, errors in before and after methods have no influence on the execution of a work item. The processing logic of the work item depends solely on the task method. The system records errors in the before and after methods as warnings in the workflow log.

**Activities**
To define an object method as a before, an after or a secondary method, select (in the column Object) a container element in the workflow container, which contains the object to be processed. Then select the object method required in the column Method.
Maintenance of Tab Page Work Item Display

Use
On the tab page *Work item display*, you can individualize the work item display with an additional tab page, and the work item preview with graphics or text.

Features

Step-specific tab page for work item display
An individual tab page is added to the work item display. This tab page is displayed in addition to the three tab pages *Basic data*, *Activities* and *Available objects* as the first tab page of the work item display. This additional tab page only appears when work items that represent this step are displayed. This tab page can be used, for example, to:
- Display important information from the current process, which does not appear in the work item display as standard.
- Provide functions that the end user has to execute often.
- Create a customer-specific 'look and feel'.

Adapting the work item preview in the Business Workplace
A user exit is available for individualizing the work item preview. For further information, refer to the *User Exit for Work Item Preview* [Extern].

A user decision can be executed in the work item preview as standard. The function module required for this is entered automatically.

If secondary methods are to be executed or methods before or after the work item execution in a user decision, delete the entry in the field *Function module*.

Activities

Step-specific tab page for work item display
Enter the program and the number of the screen into the relevant input fields. For further information, refer to *Programming a Step-Specific Tab Page* [Extern].

Adapting the work item preview in the Business Workplace
Enter the function module that you want to use in the input field *Function module*. You can create a function module using a template by selecting 📁.
**Maintenance of Tab Page Other**

**Use**
On the tab page *Other*, you can change work item priority.

**Features**

**Step priority**
Here you define what priority the work items created for this step have. [Dialog work items](#) of priority 1 are displayed in the Business Workplace as express work items.
Maintenance of Mail Transmission

Purpose
You use this step type to send a mail to different recipients. You define the text and subject of the mail when defining the step. You can use variables in the text, which are filled from the task container at runtime.
This step type replaces the wizard Include "Send mail".
In the workflow definition such a step type is represented with the symbol 📧.

Process Flow
The step definition is spread across several tab pages. To define an executable step, you only have to complete the tab page Mail. You enter the recipients and the text of the mail on this tab page. During step definition, the system requests an abbreviation and a name for a new task, which the system creates and assigns to the step.

Result
The system creates a task automatically, which sends the mail. The text to be sent is stored with the task. The system creates the variables used within the step definition in the task container and defines a binding from the workflow container to the task container for them.
Maintenance of Tab Page Mail

Use
With the specifications you make on this tab page, you define:

- Who will be a recipient of the mail.
- The text of the mail.
- The subject of the mail.

Features
With the specifications you make on this tab page, you create an executable step: When you create a new step, a task is created automatically, which sends the mail. You must give this task an abbreviation and a name for it to be created. The system requests this data in a dialog box, which appears when the step is left or earlier.

Activities
Process the following settings:

Recipient type/Recipient
First choose whether you want to specify the recipient of the mail as an e-mail address or as an organizational object.
If the workflow initiator is to be the recipient of the mail, select the relevant entry in the field *Recipient type*. The system enters the expression required automatically.
If you chose *Organizational object* as recipient type, you can enter an organizational object. You must also specify the type of the organizational object (such as position, user). As an alternative, you can choose *Expression* as the type and specify an expression that can contain one or more organizational objects.
If you choose *E-mail address* as recipient type, you can either enter a fixed e-mail address directly or specify an expression that can contain one or more e-mail addresses.

The expression can contain either just e-mail addresses or just organizational objects.

Use the F4 input help to specify an expression. You can select container elements from the workflow container to define the expression. The system creates a container element in the task container and the associated binding automatically.

Send express
Set the indicator if the mail is to be sent as an express mail within the SAP R/3 System. When e-mail addresses are used, the relevant indicator is sent to *urgent* for the mail.

Subject/Mail text
Enter the text to be sent into the relevant input fields. You can use expressions in the mail text and in the subject. Select * mediated* to enter an expression. You can select container elements from the workflow container to define the expression. The system creates a container element in the task container and the associated binding automatically.
When you create a new mail step the subject entered is used as the step name automatically. You can change the step name on the tab page Control.
Maintenance of Tab Page Control

Use
The technical specifications for sending a mail are located on this tab page. You do not have to make any changes on this tab page as a rule.

Activities
The following specifications apply:

Task
The system enters a newly created mail task in this field automatically. This is always a synchronous background task to which no agents have to be assigned.

Binding
This binding for this task is generated automatically. You do not usually have to process this yourself.

Step name
The subject of the mail is used as the step name as standard. You can change the name here.

Outcome name
This step always has an outcome. The name of this outcome is Mail sent as standard. You can change the name here.

Not in workflow log
Set this indicator if work items for this step are not to appear in the workflow log. [Seite 1420].
Maintenance of Conditions

Use
You use a condition [Extern] in a workflow definition if:

- At execution time, only one of two possible alternatives can be processed during the workflow, from a business point of view.
- The workflow system can make a decision based on the contents of the workflow container without user interaction.

In the workflow definition a condition is represented with the symbol 🔄.

Features
A condition is a branching in your workflow definition. The workflow system evaluates the condition comparing elements of the workflow container with constants, system fields or other fields. You define conditions in the condition editor, in which you can also define complex conditions. The evaluation of a condition can result in the values true or false.

Activities
You maintain conditions on the screen Create step: Condition. Here you have to assign a step name, define the condition [Seite 1012] and determine names for the outcomes true and false. These names are used to label the two outcomes of the condition in the workflow definition.

The currently valid condition is displayed on this screen. To change this or create a new condition, you must go to the condition editor by double-clicking.

Result
The system inserts a step of the type condition into the workflow definition. For both outcomes, the system adds a branch in the workflow definition and puts an undefined step into this branch. To ensure the consistency of the block structure, all branches created are brought together before the next step of the workflow definition.
Maintenance of Multiple Conditions

Use

You use a multiple condition [Extern] in a workflow definition if:

- At execution time, only one of several possible alternatives can be processed during the workflow, from a business point of view.
- The workflow system can make a decision based on the contents of the workflow container.
- The decision can be formulated based on a comparison between an expression from workflow container and a finite amount of comparison values (constants, expressions).

You also have the option of going to an Others branch if none of the conditions agree with the value of the basis of comparison.

In the workflow definition a multiple condition is represented with the symbol 📦.

Features

For the decision, the workflow system checks the value of a basis of comparison against the defined comparison values.

The basis of comparison can be a container element of the workflow container or a system field, and the comparison value can be a constant, a container element of the workflow container or a system field.

You cannot use multiline container elements.

Activities

You define the basis of comparison as an expression [Extern] with reference to the workflow container.

You enter the comparison values into the table as expressions and assign a name with which the outcome is displayed in the workflow definition. For comparison values that are not taken into account, you can include a branch into the workflow definition for Other values. To do this, you assign a name in the relevant input field.

If the basis of comparison has a value that does not agree with any comparison value and if there is no branch Other values, the workflow assumes the status error.

For each outcome, the system adds a branch in the workflow definition and puts an undefined step into this branch. To ensure the consistency of the block structure, all branches created are brought together before the next step of the workflow definition.
Maintenance of Event Creator

Use

Use an event creator if you want to publish an event from a workflow. This event can be used as follows:

- To start other workflows or tasks.
  
  For this, the event must be entered as a triggering event [Extern] for the relevant workflow or task.

- As a terminating event [Extern] of a task.

- For internal communication and synchronization.
  
  To react to an event in a workflow, you use a wait step [Extern].

In the workflow definition an event creator is represented with the symbol '★'.

Features

At runtime the system creates an event in accordance with your specifications. The system caters automatically for the workflow system variables in the event container [Extern] as well.

If the event container contains other, non-standard elements, you must define a binding from the workflow container to the event container [Seite 1223].

Activities

You give a name to the step and to the outcome. The other fields are as follows:

- Container element
  
  This container element of the workflow container must contain an object reference. The referenced object is the object to which the event refers.

  If the event is created at runtime, the system writes this object reference into the container element _Evt_Object in the event container.

- Object type
  
  This system automatically puts the type of the object referenced in the container element into this field.

- Event
  
  This event is created by the system at execution time of the workflow. The event must be defined for the object type in the Business Object Repository.

Select '★' to see whether all the receivers you expected were started. For further information, refer to Event Simulation [Seite 1363].
Maintenance of Wait Steps

Purpose
You use a wait step in a workflow definition:

- To suspend the entire execution of the workflow until a defined event has occurred.

  A workflow is waiting for requested documents to be provided for a particular applicant activity.
  Once the requested documents are received in the company, the transaction used for inbound processing and archiving creates an event which the wait step has been waiting for as a receiver.

- To wait for an event in parallel processing branches, which renders processing in the other branches superfluous.

  The event waited for should not be created in the other branches of the fork.

In the workflow definition a wait step is represented with the symbol.

Process Flow
The maintenance of a wait step is spread across several tab pages. You can process the properties of the wait step here, for example:

- Event to be waited for.
- Number of events to be waited for.
- Missed deadlines
- Outcomes

The order of the tab pages defines the recommended sequence of processing.

On the tab page Control, you can make all the entries required to define an executable wait step.

Result
The system inserts a step of the type wait step into the workflow definition. To ensure the consistency of the workflow definition, all active outcomes are inserted with the actual step. For each outcome, the system adds a branch in the workflow definition and puts an undefined step into this branch. To ensure the consistency of the block structure, all branches created are brought together before the next step of the workflow definition.

At runtime, a wait step is represented by a wait step work item.
Wait step work items are not displayed in the Business Workplace but can be found using the work item selection [Seite 1490].
Maintenance of Tab Page Control

Use
With the specifications you make on this tab page, you define the event you want the workflow to wait for.

Activities
You must describe the expected event uniquely. Specify the following:

Container element
In this step, the workflow waits for an event created by a specific object. As this is not yet known at definition time, you specify a container element of the workflow container, which will contain an object reference at runtime. With this information, the workflow system enters the instance linkage [Extern].

The entries in the tables for event receiver linkage (linkage tables [Extern]) are made exclusively by the workflow system itself.

Object type
Here the workflow system automatically enters the type of the object referenced in the container element.

Event
Choose the event to be waited for in this step. The event must be defined for the object type in the Business Object Repository.

Number of events
You enter how many times the same event must occur for execution of the workflow to be continued.

Binding
Process the binding definition from the event container [Seite 1227] to the workflow container if you want to use information supplied with the event in the workflow. To do this, select Binding.

In general, this binding definition is not required.

Name
Enter the ID of the outcome of the wait step, with which processing is to be continued after the event occurs.
Maintenance of Tab Page Outcomes

Use
The specifications you make on this tab page determine the outcomes of the step for which you want to model a reaction. The outcome entered on the tab page Control is only displayed here. If reactions to missed deadlines are modeled, the outcomes defined for the relevant missed deadline are only displayed here.

Features
On this tab page you can only activate the outcome Processing obsolete. All other outcomes are determined by the system and activated automatically.
All possible outcomes are listed in the following table:

<table>
<thead>
<tr>
<th>Outcome</th>
<th>This outcome exists if ...</th>
<th>Notes and comments</th>
</tr>
</thead>
<tbody>
<tr>
<td>Wait step completed</td>
<td>Always</td>
<td>You cannot deactivate this outcome.</td>
</tr>
<tr>
<td>Processing obsolete</td>
<td>The work item can be set to obsolete via a step of the type process control.</td>
<td>The subsequent steps defined after this exception are executed. This outcome should only be activated if the modeled reaction to a missed deadline contains a step that sets the work item for this step to obsolete.</td>
</tr>
<tr>
<td>Requested end</td>
<td>The relevant deadline monitoring is activated and a modeled reaction required.</td>
<td>Within these branches, you can model a process control step that sets the work item of the step to obsolete. You cannot deactivate these outcomes.</td>
</tr>
<tr>
<td>Latest end</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Latest start</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Activities
You activate the outcomes that you want to actually transfer into the workflow definition. Select the relevant line.
Active outcomes are marked with and have a modeling branch in the workflow definition.

⚠️ If you deactivate an active outcome, the whole of the associated branch is deleted.

For further information about an outcome, select it and choose Detail in the context menu.
Maintenance of Tab Page Notification

Use
The specifications you make on this tab page define the notification agent for completion. This entry is optional. These notification agents receive a notification of completion [Extern] when the relevant work item is set to status completed.

Activities
Enter a message recipient for completion. For further information, refer to Defining Responsibilities [Extern]. If no recipient is entered, no notification of completion is sent. The notification of completion contains the completion text that can only be processed [Seite 1201] in the task definition, and a reference to the work item display of the completed work item.
Maintenance of Deadline Tab Pages

Use
You can define all the deadlines for this step with the specifications you make on these tab pages. All these entries are optional.

Features
You can set the following deadlines on the corresponding tab pages:
- Requested start [Extern]
- Latest start [Extern]
- Requested end [Extern]
- Latest end [Extern]
You can define reactions to a missed deadline for the last three deadline types. The system offers the following possible reactions:
- Notification
- Free modeling
Activated deadlines are marked with 🎉 in the tab page index.
You define deadlines with respect to a reference date/time. The system offers the following reference date/times:
- The creation date/time of the work item.
- The creation date/time of the workflow to which the monitored work item belongs.
- A date in the form of an expression, which is derived from the context of the application during execution of the workflow.

Activities
You activate monitoring of the relevant deadline by selecting a reference date/time for the deadline. If you choose expression, you have to define the reference date/time by specifying expressions [Extern] for the date and time yourself. Use the F4 input help for entering expressions.

The value referenced using the expression must be of data type D for the date and data type T for the time. If you specify a date but no time, the system sets the time to 00:00:01 (requested and latest start) or 23:59:59 (requested and latest end).

Process the deadline by entering a duration and an appropriate unit. Negative durations only apply if you define the reference date/time yourself via an expression.
The reaction to a deadline being reached depends on the type of deadline:

Requested start
If the work item is created before the requested start, it is assigned the status waiting. When the requested start is reached, the system sets the work item to status ready. If the work item is created after the requested start, it is assigned the status ready immediately.

Latest end, latest start, requested end
You can either notify someone or model a reaction in the workflow definition for when the deadline is reached.
• Display text (standard escalation)

If you want to notify someone, you must specify a recipient [Extern] on the tab page Display text. The message text is displayed in the display area. It can only be changed in the task definition.

• Modeled

If you want to model your own reaction, enter a name for the new outcome on the tab page Modeled. A reaction branch is added to the workflow definition, which is processed if the deadline is missed.

If a deadline is missed, the original step is not yet completed. The steps following this "missed deadline event" do not end the original activity.

For further information, refer to Modeled Deadline Monitoring or Standard Escalation [Extern].
Maintenance of Process Control

Use
You use this step type to complete or cancel other work items of the current workflow or the workflow itself at runtime. In the workflow definition a process control step is represented with the symbol 🛡.

Features
You determine the type of process control in the field Function. This step always has an outcome.

Process control influences other work items

Cancel work item
The workflow system forces another work item of the same workflow into the status logically deleted. This completes this other work item. Defined subsequent steps of this work item are not executed. To indicate the work item to be canceled, you specify the node number of a step from the current workflow definition in the field Workflow step.

This function can only be used if the process control step and the step to be canceled are located in a fork. Note that the other work item is only forced into the status logically deleted, if it has the status in process when the process control step is executed.

Set work item to obsolete
The workflow system forces another work item of the same workflow into the status completed and continues the processing of the workflow in the branch processing obsolete. This other work item must therefore support the exception processing obsolete. To indicate the work item for which processing is to be set to obsolete, you specify the node number of a step from the current workflow definition in the field Workflow step.

You can use this function to define modeled deadline monitoring. The work item whose deadline has been missed is forced into status completed. For further information on this type of modeling, refer to Modeled Deadline Monitoring or Standard Escalation [Extern].

You should not use this function in forks to force work items in other branches of the fork into status completed.

Process control influences the workflow itself

Terminate workflow
The workflow system terminates the current workflow. If there are any incomplete work items, for example in parallel branches of this workflow, they are forced into status logically deleted. The workflow is in its defined end status. If the terminated workflow was a subworkflow of a superordinate workflow, the system executes the binding between the container of the subworkflow and the container of the superordinate workflow in accordance with the definition, and continues the superordinate workflow.
Cancel workflow
The workflow system cancels further execution of the current workflow. The workflow work item is forced into status *logically deleted*. If there are any incomplete work items, for example in parallel branches of this workflow, they are also forced into status *logically deleted*.
If the canceled workflow was a subworkflow of a superordinate workflow, the superordinate workflow is not continued.

Activities
You select the function to be performed in this step and enter the node number of the workflow step to be cancelled if applicable. Assign a name for the step outcome.
Maintenance of Container Operations

Use
You use a container operation to change a container element of the workflow container at runtime. In the workflow definition a container operation is represented with the symbol 📃.

Features
A container operation has one outcome. The following operation types are available:

Arithmetic operation
In an arithmetic operation, the result of a simple calculation is assigned to a container element of the workflow container. The arithmetic operation requires two expressions.

Assign/Append
In assigning or appending, the content of a container element or expression is assigned to a container element of the workflow container. The assignment only requires an expression. The second expression and the operator are not required.

Assign
The expression is assigned to the container element. This content of the container element is deleted first. The assignment is possible both for single-line and for multiline container elements.

When assigning to multiline container elements, the following apply:
- `<MultilineElement>` =
  The multiline element is initialized.
- `<MultilineElement>` = `<MultilineExpression>`
  The content of `<MultilineElement>` is deleted.
  The content of `<MultilineExpression>` is assigned to `<MultilineElement>`.
- `<MultilineElement>` = `<Single-LineExpression>`
  The content of `<MultilineElement>` is deleted.
  The content of `<Single-LineExpression>` is entered into the first line of the multiline container element.

Append
The expression is appended to the container element. This extends the content of the multiline container element. Appending is only possible for multiline container elements.

When assigning to multiline container elements, the following apply:
- `<MultilineElement>` <- `<MultilineExpression>`
  The content of `<MultilineExpression>` is appended to the content of `<MultilineElement>`.
- `<MultilineElement>` <- `<Single-LineExpression>`
  The content of `<Single-LineExpression>` is appended to the multiline container element as the last line.
Maintenance of User Decisions

Purpose
In a user decision [Extern], the task description is displayed to the current agent together with the possible decision options during execution. You can use a user decision in the following situations:

- Only one of several possible alternatives can be processed in the workflow from a business point of view at execution time.
- A decision about the progression of the workflow must be made in dialog with a user.
- An instruction (= user decision with only one decision option) is required to continue the workflow.
- For approval and release steps, possibly in connection with a secondary method (the latter is possible without the application having to cater for these functions).

In the workflow definition a user decision is represented with the symbol 

Prerequisites
The user decision refers to a decision task [Extern]. In order for an individual text to be displayed in the dialog box for the user decision, you must create your own decision task. You specify the individual text as task description in the task definition. For further information, refer to Creating Decision Tasks [Seite 1078].

If you do not think an individual text is necessary, you do not generally need your own decision task. In this case, you use the task generic decision task (TS00008267).

Process Flow
The maintenance of a user decision is spread across several tab pages. You can process its properties here, for example:

- Decision options
- Responsible agents [Extern]
- Deadline monitoring [Extern]

The order of the tab pages defines the recommended sequence of processing.

On the tab page Decision, you can make all the entries required to define an executable user decision.

When a user decision is created, the system enters the SAP decision task generic decision task (TS00008267) into the relevant field on the tab page Control as standard. You can change this entry at any time.

You can change this setting for the decision task in the Workflow Builder Customizing [Extern].

If you use a decision task you defined in a user decision and have text variables for substitution, you must provide the relevant elements in the task container with values via a binding from the workflow container. For further information, refer to Binding Definition in Decision Tasks [Seite 1231].
Secondary methods with user decision

If you use the user decision as a step for approval or release of an object, you can display the object in question as a secondary method, hence supplementing the functions of the user decision. For further information, refer to Maintenance of Tab Page Methods [Seite 1083].

Result of user decision

If you want to store the result of a user decision in an element of the workflow container in order to use it again in a later step (for example, loop or condition), you must define a binding from the task container to the workflow container. For further information, refer to Binding Definition in Decision Tasks [Seite 1231].

Result

The system inserts a step of the type user decision into the workflow definition. To ensure the consistency of the workflow definition, all active outcomes are inserted with the actual step, whereby every decision alternative is represented by an outcome. For each outcome, the system adds a branch in the workflow definition and puts an undefined step into this branch. To ensure the consistency of the block structure, all branches created are brought together before the next step of the workflow definition.

At runtime, this user decision is represented by a dialog work item [Extern] in the Business Workplaces of its recipients [Extern].

When a recipient chooses the work item created for this step for processing in the Business Workplace, a screen is displayed with the description text and the defined decision texts and one of the alternatives offered can be selected.

The recipient can also display or create attachments and objects on this screen.
Maintenance of Tab Page Decision

Use
With the specifications you make on this tab page, you define:

- The decision text used when the user decision appears to the recipient at runtime.
- The decision options offered to the recipient for selection.
- Who the responsible agent of the decision is.

Activities
You make the following specifications to define a decision:

Decision title
You can include up to four system fields or expressions from the workflow container in the title as parameters. To do this, you mark the points at which the parameters are to be inserted with placeholders and enter the expressions in the fields Parameter1 to Parameter4.

At runtime, the system replaces the placeholders with the current values.

Decision options
For each option, specify a name and a decision text that is displayed to the recipient. All options defined here are outcomes of the step and are transferred with their names to the tab page Outcomes.

Agent determination
Define the responsible agents and the excluded agents. These only apply to this workflow definition. For further information, refer to Defining Responsibilities.

If you only make entries on this tab page, a user decision on the basis of the decision task supplied by SAP is transferred into the workflow definition.
Maintenance of Tab Page Control

Use
With the specifications you make on this tab page, you define:

- Which decision task is used.
- How the user decision is presented.

Activities
Make the following settings:

Task
A decision task [Extern] must be entered in this field. You should only use another decision task if you want to restrict the selection of possible agents [Extern] or use another decision text. Select to create a new decision task. For further information, refer to Creating Decision Tasks [Seite 1078]. The properties and task description of the selected task are displayed. The task description is displayed to the decision-maker as a decision text at runtime.

Step properties
You can define the following step properties:

- Work items for this step appear in the workflow log [Seite 1420].
- The following step is to be executed immediately by the same recipient [Extern] (for further information, refer to Advancing with Immediate Dialog [Seite 1453])

Binding
Process the binding between task container and workflow container. This is necessary if the agent is to create a document that is based on a template as an attachment, and if text variables are used in the task description. To do this, select Binding (existing). For further information, refer to Binding Definition in a Decision Task [Seite 1231].

Template for attachments
You can determine a container element in the workflow container that contains a template for an attachment that is to be created. If an attachment is created during the execution of the step, the user receives the template and can add to it.
Creating Decision Tasks

Use
A decision task is only required for steps of the type *user decision*. You should only create a new decision task if:

- You want to restrict the possible agents.
- You require another decision text.

A decision task must always refer to the object type DECISION with the method *Process*.

Procedure
Always create a new decision task as a copy of the standard decision task supplied by SAP. This is task *generic decision task* (TS00008267).

1. Go into the step definition of a decision task. A decision task is referenced there in the field *Task*. If this is task TS00008267 or a copy of this task, select ![select task].

2. Specify a new abbreviation and new name for the task and assign it to a development class.

3. Process the relevant text types on the tab page *Description*. The task description is displayed to the recipient when the user decision is executed. For further information, refer to *Maintaining Task Texts [Seite 1201]*.

4. Add the elements that you inserted into the description text as text variables (or object references) to the task container as *mandatory import elements*. To do this, select ![select container].

   In order for the text variables to be filled with values at runtime, you must define the binding from the workflow container to the task container in the definition of the user decision.

5. Process the *possible agents [Extern]* of your decision task. Choose *Additional data → Agent assignment → Maintain*. 
Maintenance of Tab Page Outcomes

Use

The specifications you make on this tab page determine the outcomes of the step for which you want to model a reaction. The outcomes entered on the tab page Decision for the decision options are only displayed here. If reactions to missed deadlines are modeled, the outcomes defined for the relevant missed deadline are only displayed here.

Features

On this tab page you can only activate the outcome Processing obsolete. All other outcomes are determined by the system and activated automatically.

All possible outcomes are listed in the following table:

<table>
<thead>
<tr>
<th>Outcome</th>
<th>This outcome exists if ...</th>
<th>Notes and comments</th>
</tr>
</thead>
<tbody>
<tr>
<td>Decision option</td>
<td>You created options on the tab page Decision.</td>
<td>These outcomes cannot be deactivated. To remove them, you must delete the decision option.</td>
</tr>
<tr>
<td>Processing obsolete</td>
<td>The work item can be set to obsolete via a step of the type process control.</td>
<td>The steps defined after this outcome are executed. This outcome should only be activated if the modeled reaction to a missed deadline contains a step that sets the work item for this step to obsolete.</td>
</tr>
<tr>
<td>Requested end</td>
<td>The relevant deadline monitoring is activated and a modeled reaction required.</td>
<td>Within these branches, you can model a process control step that sets the work item of the step to obsolete. You cannot deactivate these outcomes.</td>
</tr>
</tbody>
</table>

Activities

You activate the outcomes that you want to actually transfer into the workflow definition. Select 📐 in the relevant line.

Active outcomes are marked with 🌿 and have a modeling branch in the workflow definition.

⚠️ If you deactivate an active outcome, the whole of the associated branch is deleted.

For further information about an outcome, select it and choose Detail in the context menu.
Maintenance of Tab Page Notification

Use
The specifications you make on this tab page define the *notification agent for completion*. This entry is optional.
These notification agents receive a *notification of completion* when the relevant work item is set to status *completed*.

Activities
Enter a message recipient for completion. For further information, refer to [Defining Responsibilities](Extern). If no recipient is entered, no notification of completion is sent. The notification of completion contains the *completion text* that can only be processed in the task definition, and a reference to the work item display of the completed work item.
Maintenance of Deadline Tab Pages

Use
You can define all the deadlines for this step with the specifications you make on these tab pages. All these entries are optional.

Features
You can set the following deadlines on the corresponding tab pages:

- **Requested start** [Extern]
- **Latest start** [Extern]
- **Requested end** [Extern]
- **Latest end** [Extern]

You can define reactions to a missed deadline for the last three deadline types. The system offers the following possible reactions:

- Notification
- Free modeling

Activated deadlines are marked with 🕒 in the tab page index.

You define deadlines with respect to a reference date/time. The system offers the following reference date/times:

- The creation date/time of the work item.
- The creation date/time of the workflow to which the monitored work item belongs.
- A date in the form of an expression, which is derived from the context of the application during execution of the workflow.

Activities
You activate monitoring of the relevant deadline by selecting a reference date/time for the deadline. If you choose *expression*, you have to define the reference date/time by specifying expressions [Extern] for the date and time yourself. Use the F4 input help for entering expressions.

The value referenced using the expression must be of data type D for the *date* and data type T for the *time*. If you specify a date but no time, the system sets the time to 00:00:01 (requested and latest start) or 23:59:59 (requested and latest end).

Process the deadline by entering a duration and an appropriate unit. Negative durations only apply if you define the reference date/time yourself via an expression.

The reaction to a deadline being reached depends on the type of deadline:

**Requested start**
If the work item is created before the requested start, it is assigned the status *waiting*. When the requested start is reached, the system sets the work item to status *ready*. If the work item is created after the requested start, it is assigned the status *ready* immediately.

**Latest end, latest start, requested end**
You can either notify someone or model a reaction in the workflow definition for when the deadline is reached.
• Display text (standard escalation)

If you want to notify someone, you must specify a recipient [Extern] on the tab page Display text. The message text is displayed in the display area. It can only be changed in the task definition.

• Modeled

If you want to model your own reaction, enter a name for the new outcome on the tab page Modeled. A reaction branch is added to the workflow definition, which is processed if the deadline is missed.

If a deadline is missed, the original step is not yet completed. The steps following this "missed deadline event" do not end the original activity.

For further information, refer to Modeled Deadline Monitoring or Standard Escalation [Extern].
Maintenance of Tab Page Methods

Use
In addition to the task's method, you can define the following methods:

- **Before method [Extern]**: Executed before execution of the method.
- **Secondary method [Extern]**: Executed at the same time as the method.
- **After method [Extern]**: Executed after execution of the method.

Prerequisites
The methods and their object types are defined in the *Business Object Repository*. The objects on which the methods are executed must be referenced in a container element in the workflow container.

> If you define one of these methods for a user decision, you must delete the entry in the field *Function module* on the tab page *Work item display*.

Features

Calling before and after methods
When executing a work item with before or after methods, the system first calls the before methods synchronously in any order and processes them. Then the system starts the task method and secondary methods in different sessions. Secondary methods are only for display. Any changes carried out in them have no (direct) effect on the workflow. Finally, the system calls the after methods synchronously in any order and processes them.

> The after methods being called is not dependent on the result of the task method. They are called even if the task method terminated with an error.

> The after methods being called is not dependent on whether or not the task method is terminated with an event. They are called immediately after the synchronous part of the method.

With work items that are set to status *completed* by a terminating event without method execution, the before, secondary and after methods are not called.

Before methods cannot create the object on which the task method is to run. However, after methods can run on a object created by the task method.

Data transfer into and out of the methods
There is no separate binding between task container and method container for these methods. The methods are given the method container of the task method. If no binding is defined between the task and the method, the system passes the task container.

Secondary methods cannot return data to the workflow nor influence the process control.

Before and after methods can evaluate and manipulate the method container in order to influence the subsequent method or steps.
Error handling
As with secondary methods, errors in before and after methods have no influence on the execution of a work item. The processing logic of the work item depends solely on the task method. The system records errors in the before and after methods as warnings in the workflow log.

Activities
To define an object method as a before, an after or a secondary method, select (in the column Object) a container element in the workflow container, which contains the object to be processed. Then select the object method required in the column Method.
Maintenance of Tab Page Work Item Display

Use

On the tab page *Work item display*, you can individualize the work item display with an additional tab page, and the work item preview with graphics or text.

Features

**Step-specific tab page for work item display**

An individual tab page is added to the work item display. This tab page is displayed in addition to the three tab pages *Basic data*, *Activities* and *Available objects* as the first tab page of the work item display. This additional tab page only appears when work items that represent this step are displayed. This tab page can be used, for example, to:

- Display important information from the current process, which does not appear in the work item display as standard.
- Provide functions that the end user has to execute often.
- Create a customer-specific 'look and feel'.

**Adapting the work item preview in the Business Workplace**

A user exit is available for individualizing the work item preview. For further information, refer to the User Exit for Work Item Preview [Extern].

A user decision can be executed in the work item preview as standard. The function module required for this is entered automatically.

![](warning)

If secondary methods are to be executed or methods before or after the work item execution in a user decision, delete the entry in the field *Function module*.

Activities

**Step-specific tab page for work item display**

Enter the program and the number of the screen into the relevant input fields. For further information, refer to Programming a Step-Specific Tab Page [Extern].

**Adapting the work item preview in the Business Workplace**

Enter the function module that you want to use in the input field *Function module*. You can create a function module using a template by selecting 📦.
Maintenance of Tab Page Other

Use
On the tab page Other, you can change the priority and repeated execution of the work item.

Features

Step priority
Here you define what priority the work items created for this step have. Dialog work items of priority 1 are displayed in the Business Workplace as express work items.

Table-Driven Dynamic Parallel Processing
You can process this step several times in parallel. The multiline container element entered here controls this parallel processing. For further information, refer to Table-Driven Dynamic Parallel Processing [Extern].

Repeat counter for work items with temporary errors
Maximum number of attempts made by the system to restart a background work item [Extern] with temporary errors. If all of the attempts are unsuccessful, the work item status is set to:

- **Error** (if the temporary exception is not modeled)
- **Completed** (if the temporary exception is modeled)

This function is only available for steps that refer to a single-step task without dialog (“background task”). If the repeat counter has the value 0, the maximum number of retries is determined from the settings in Customizing [Extern].

For further information, refer to Error Handling for Background Work Items [Seite 1388].
Maintenance of UNTIL Loops

Use
You use an UNTIL loop [Extern] in a workflow definition if:

- At execution time with one of two possible alternatives the workflow is to “jump back” to process sections of the workflow definition again.
- The workflow system can make a decision based on the contents of the workflow container.

In the workflow definition an UNTIL loop is represented with the symbol 🔄.

Features
An UNTIL loop consists of a sequence of steps processed by the workflow system until a termination condition is recognized as true by the workflow system. In an UNTIL loop, the steps are processed at least once before the loop can be terminated.

Outcome of UNTIL Loops
The evaluation of a condition can return the values true or false. You assign an event name for each of the two values. These names are used to label the two outcomes of the loop in the workflow definition. In an UNTIL loop, the return jump always occurs when the condition evaluation returns the value false. The workflow execution is continued if the condition evaluation returns the value true.

Activities
To define the loop you must assign names for the step and enter the outcomes. Use the condition editor to enter the termination condition. For further information, refer to Condition Editor [Seite 1012]. The system creates a loop containing an undefined step in the workflow definition. You must then define the loop content.
Maintenance of WHILE Loops

Use
You use a **WHILE loop** in a workflow definition if:

- At execution time, only **one of several** possible alternatives can be processed in the workflow from a business point of view, and then the comparison is to be performed again.
- The **workflow system** can make a decision based on the contents of the workflow container.
- The decision can be formulated based on a comparison between an expression from workflow container and a finite amount of comparison values (constants, expressions).

In the workflow definition a WHILE loop is represented with the symbol 🔄.

Features
At runtime, the workflow system compares the value of an expression of the workflow container with the defined comparison values. If the system cannot establish any agreement, the workflow is continued in the **Other values** branch.

If the value of the expression of the workflow container agrees with one of the comparison values, the steps modeled in the relevant branch are processed. Then the return jump occurs and the system runs the loop check again.

Activities
Assign a step name and define the basis of comparison as an expression of the workflow container. Enter a name for each comparison value. These names identify the various processing branches in the workflow definition.

The branch **Other values** is created automatically by the system even if you do not assign a name. This ensures that the workflow definition is executable.

Make sure that in each branch of the loop you implement steps that change the basis of comparison in an appropriate way.
Maintenance of Forks

Use
You use a fork in a workflow definition when the business process can be continued by several users at the same time. You can also configure the fork in such a manner that not all branches have to be processed.

In the workflow definition, the start of a fork is shown with the symbol and the end with .

Integration
In addition to this explicitly-modeled parallelism, there is also table-driven, dynamic parallel processing and the work queue. For further information, please refer to Implementation Options for Parallel Processing [Extern].

Features
You can create any number of branches in a fork. All branches of the fork flow into a join operator .

When one branch of a fork reaches this join operator at runtime, the end conditions are checked.

The system checks whether the number of branches processed agrees with the number of branches required as specified in the definition. The system then checks whether a condition produces the result true.

If one of the end conditions is fulfilled, any existing work items of the fork are set to status logically deleted, and the workflow is continued after the join operator.

The individual branches of a fork should be functionally independent.

Activities
To define a fork, you specify the number of parallel branches required with the step name. The fork, consisting of the start operator, the branches each with an undefined step [Extern] and the join operator, is inserted into the workflow definition.

To define the termination of the fork, you enter the number of branches required and the condition. For further information on defining the condition, please refer to Condition Editor [Seite 1012].

If you make no entry for the number of branches required, the system takes the number of branches.

You can transfer an existing modeled block into a fork as a new branch as follows:

3. Copy or cut the block.
4. Select and choose Paste block in the context menu.
Top-Down Modeling

Use
Top-down modeling allows you to create workflow definitions taking into account block orientation without knowledge of the individual tasks used. At an early modeling stage, it is therefore possible to create incomplete but consistent workflow definitions. It may well be appropriate to define the flow logic initially for complex workflows for example. Top-down modeling is also useful for presentation purposes.

Prerequisites
To model a workflow "top-down", start the Workflow Builder [Seite 1002] and select ☐ in the application toolbar.

Features
Top-down modeling
You can use all the step types available when modeling. The following top-down task can be used when describing an activity in the workflow definition:

<table>
<thead>
<tr>
<th>Task</th>
<th>TS30100074</th>
</tr>
</thead>
<tbody>
<tr>
<td>Abbreviation</td>
<td>WF_MODTASK</td>
</tr>
<tr>
<td>Name</td>
<td>Modeling task for TOP/DOWN approach</td>
</tr>
</tbody>
</table>

You should overwrite the proposed step name with a name that describes the significance or function of this step in your scenario.

The top-down task references an object method with a result [Extern]. The result can assume six values, which are offered in the workflow definition as outcomes [Extern]. You can then choose the number of outcomes required for your scenario and name them. You must deactivate possible outcomes that you do not require. The top-down task can be incorporated in a workflow definition more than once.

It is also possible to maintain the agent assignment to the individual steps.

Productive use
To use the "top-down" modeled workflow productively, you must overwrite the task ID TS30100074 in the definition of the activity with the new ID. The definition of the task entered is read and the binding definition between the workflow container and task container generated if applicable. The outcomes of the activity can be described in the normal way. For further information, refer to Maintenance of Tab Page Outcomes [Seite 1028] When you return from the definition of the activity to the display of the workflow definition, superfluous outcomes of the top-down task are deleted. New branches are created in the workflow definition for additional or unassigned outcomes of the task entered.
WebFlow

Purpose
WebFlow provides functions that enable SAP Business Workflows to be executed across the Internet. WebFlow makes it possible to:

- Send XML documents to other systems from a Business Workflow
- Start a Business Workflow on another SAP System from your SAP System
- React to the results of a Business Workflow executed in another system
- Start a Business Workflow when an appropriate XML document is received

Prerequisites
A Web server and an Internet Transaction Server (ITS) must be configured for the SAP System for the transmission and reception of the XML documents to be successful. For further information, refer to Defining the Web Server [Seite 1513].

Process Flow

Overview of the process flow of a Web activity

You can either send XML documents you have created yourself or have WebFlow create a Wf-XML document [Extern] from your data. For further information, refer to Creation of a Wf-XML Document [Seite 1093].

To send XML or Wf-XML documents, you use the step type Web activity [Extern] in a workflow. For further information, refer to Maintenance of a Web Activity [Seite 1037].

The transfer protocols http and https are used for the transmission. A transmission of this type is not completed until the other system has sent back an http reply.

A special service [Extern] must be active for a Business Workflow to be started by an inbound Wf-XML document in a SAP System. This service processes the inbound Wf-XML document and writes the data...
contained into the import parameters of the Business Workflow to be started. The Business Workflow is then started and executed.

The following are possible scenarios:

**Sending an XML document to another system**
You can send an XML document that you created yourself from your SAP System A to system B using a WebFlow. The Web activity sends your XML document to the recipient you specified. If system B’s http reply contains an XML document, it can be stored in the workflow container. The Business Workflow in system A is then continued.

**Sending a Wf-XML document to another system**
You can have the WebFlow create the XML document to be sent. A Wf-XML document is then sent to system B, which contains your data and the http reply is evaluated by the WebFlow. The Business Workflow in system A is then continued.

**Sending a Wf-XML document to another system with feedback**
This scenario is the same as the scenario without feedback up to reception of the http reply. The Web activity waits for a Wf-XML reply document. The Web activity writes the data in it into the workflow container and the Business Workflow in system A is continued.

**Sending a Wf-XML document to another SAP system and starting a Business Workflow**
If you want to start a Business Workflow in SAP System B using a WebFlow, have the system create a Wf-XML document for you from your data. You can use a wizard to generate the URL to which the Wf-XML document is sent. To ensure that the Business Workflow to be started in system B receives all the data required, you must know the interface [Extern] of the Business Workflow to be called in full. The http reply is evaluated by the WebFlow and the Business Workflow in System A is then continued.

**Sending a Wf-XML document to another SAP system and starting a Business Workflow with feedback**
This scenario is the same as the scenario without feedback up to reception of the http reply. Once the Business Workflow in system B is terminated, the export parameter data of the interface is sent back to the calling Business Workflow in system A as a Wf-XML reply document. The Web activity writes this data into the workflow container and the Business Workflow in system A is continued.

**Starting a Business Workflow with an inbound Wf-XML document without reply**
When a Wf-XML document arrives, the WebFlow tries to start the Business Workflow named in the document. The data contained in the Wf-XML document is written into the relevant import parameters of the interface. The WebFlow sends an http reply back.

**Starting a Business Workflow with an inbound Wf-XML document with reply**
This scenario is the same as the previous scenario without reply. Once the Business Workflow has been terminated the WebFlow creates a Wf-XML reply document. The export parameter data of the interface is written into this, and the document is sent back to the calling system.
Creation of a Wf-XML Document

Use
The data exchange with other systems is performed with an XML document in a WebFlow. The workflow system can generate an XML document automatically for the data to be transferred, which is in accordance with the Wf-XML transfer format of the Workflow Management Coalition in the beta version of 11 January 2000 with the following modification:
The operation `CreateProcessInstance` is used to start a Business Workflow.

Features
The following Wf-XML documents are created automatically:

**Wf-XML documents created by a WebFlow**

<table>
<thead>
<tr>
<th>SAP System A</th>
<th>Transmission with http or https</th>
<th>SAP System B</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**Wf-XML document (1)**
The import parameters from the task container are used in creation. These are derived from the interface [Extern] of the reference workflow of the Web activity [Extern]. The import parameter data of the task container is written into the Wf-XML document. The following content is transferred depending on the data type reference [Extern]:

<table>
<thead>
<tr>
<th>Data type reference</th>
<th>Content transferred</th>
</tr>
</thead>
<tbody>
<tr>
<td>ABAP Dictionary</td>
<td>All data</td>
</tr>
<tr>
<td>Object reference to XML_DOC</td>
<td>Referenced XML document itself is transferred in full</td>
</tr>
<tr>
<td>Other object reference</td>
<td>Not supported</td>
</tr>
</tbody>
</table>

Multiline import parameters are not supported.
Within the Wf-XML document the data from a container element is stored in separate areas denoted using `<container_element_ID> data </container_element_ID>`. The assignment of the data to the individual areas is made on the basis of name parity. When an SAP System receives a Wf-XML document that is to start or continue a workflow, the data from the individual sections of the Wf-XML document is assigned to container elements in the workflow container on the basis of name parity as well. For an example of a generated Wf-XML document (1), refer to number 1 in Examples of Created Wf-XML Documents [Extern].

If the indicator Wait for feedback is set within the Web activity, the URL to which the Wf-XML reply document (3) from system B is to be sent is written into the Wf-XML document (1) under `<ObserverKey>`. This ensures that the Wf-XML reply document (3) can be evaluated by the Web activity. The Web server [Seite 1513] must be defined for the URL to be created.

http reply (2)

If SAP System B receives a Wf-XML document, it sends an http reply (2) in the form of an XML document. This confirms the start of the Business Workflow. For an example of an http reply (2), refer to number 2 Examples of Created Wf-XML Documents [Extern].

Wf-XML reply document (3)

If system A is waiting for feedback, SAP System B sends a Wf-XML reply document (3) after the Business Workflow is completed. The export parameters of the workflow are written in the relevant areas of the Wf-XML reply document (3) as with the import parameters when the call is made. For an example of a generated Wf-XML reply document (3), refer to number 3 Examples of Created Wf-XML Documents [Extern].

Activities

The system creates the Wf-XML document automatically if you chose transfer format 01 in the Web activity. You will find the XML documents created in the work item display of the Web work item [Extern].

If a Wf-XML document was to start a Business Workflow in your system but this has failed, you can find and display the Wf-XML document using the report Displaying XML Documents [Seite 1512].
Ad Hoc Workflow

Purpose
There are two types of ad hoc workflow:

Ad hoc agent assignment
A workflow of this type enables agents to be assigned to workflow steps when the workflow is started. During workflow execution the agent assignment can be changed by authorized users at any time for steps not yet completed.

Ad hoc definition
Once you have selected a work item in the Business Workplace, you can start workflows that use the application object of the work item. If the workflow required does not yet exist, you can create it ad hoc.

Prerequisites
To execute an ad hoc definition effectively, tasks groups must be assigned to the application object types. For further information, refer to Defining Task Groups for Ad Hoc Workflows [Seite 1099].

Process Flow

Ad hoc agent assignment
You can convert an existing workflow into an ad hoc workflow in the Workflow Builder. No responsible agents should be entered in the step definitions for all steps to which ad hoc users are to be assigned during execution. Container elements are created automatically by the system in the workflow container and the relevant task containers during conversion into an ad hoc workflow. An expression [Extern] is assigned to the input fields for responsible agents.
For further information, refer to Defining Ad Hoc Agent Assignment [Seite 1096].
If this ad hoc workflow is started by a user, this workflow initiator defines the recipients [Extern] who are to execute the individual steps. Throughout the execution of the workflow the workflow initiator can change this assignment of recipients for steps not yet completed at any time.
For further information, refer to Executing Workflows with Ad Hoc Agent Assignment [Seite 1097].

Ad hoc definition
If you select a work item in the Business Workplace, you can choose Environment → Start Workflows. This displays a dialog box for starting workflows, in which only the workflows that work with the application object of the selected work item are offered. Here you can define a workflow ad hoc if the workflow you require does not exist yet. The Workflow Builder (with restricted functionality) can be used to define the ad hoc workflow. Tasks are available in the tray My workflows and tasks, which you can use in the workflow definition.
For further information, refer to Defining Workflows Ad Hoc [Seite 1098].
Defining Ad Hoc Agent Assignment

Use
You can define ad hoc agent assignment for a workflow. This enables the user who starts the workflow to assign a recipient to all relevant steps at the start of execution. This user can also change the assignment of recipients for steps that have not yet been completed at any time during execution of the workflow.

Prerequisite
You have defined the workflow for which you want to enable ad hoc agent assignment in the Workflow Builder.

Procedure
You are in the Workflow Builder. The workflow is opened in change mode.
4. Identify all the steps for which you want to enable ad hoc agent assignment.
5. Delete the responsible agents [Extern] in the respective step definitions.
6. Choose Extras → Ad hoc functions → Enable ad hoc agent assignment.

The system adds container elements required for the ad hoc agent assignment to the workflow container. An expression [Extern] is assigned automatically to the input fields for the responsible agents of a step as well.

If you want to deactivate ad hoc agent assignment for particular steps, replace the expression for the responsible agents with an entry. If you want to integrate steps added subsequently into the ad hoc agent assignment, carry out step 3 again.

Result
Your workflow has been changed making ad hoc agent assignment possible. Test this by choosing Extras → Ad hoc functions → Test ad hoc agent assignment.
Executing Workflows with Ad Hoc Agent Assignment

Use
The ad hoc agent assignment is performed at the start of the workflow. The user who started the workflow can perform an ad hoc agent assignment for steps that have not yet been completed at any time while a workflow is running.

Procedure

Ad hoc agent assignment at the start of a workflow
6. Select the workflow to be started.
   If this is an ad hoc workflow, input fields are displayed for agents in the lower part of the preview area.
7. Assign the relevant users to the individual steps.

Changing the ad hoc agent assignment while a workflow is running

Only the user who started the workflow can change the assignment during execution.
4. Open the workflow outbox in your Business Workplace.
5. Select the workflow for which you want to change the ad hoc agent assignment.
6. Select to change the agent assignment.
   This displays the dialog box Ad hoc change of workflow agents. If a step has already been completed, the actual agent [Extern] is displayed. For steps that have not yet been completed, the assigned recipients are displayed.
8. Change the agent assignment.

Result
The workflow is executed with the new agent assignment. If you change the agent assignment for a step that is already being executed, the work item is forwarded from the old to the new actual agent.
Defining Workflows Ad Hoc

Use
You can define a workflow ad hoc to be executed with the application object of a selected work item.

Prerequisites
A task group must be defined to make tasks and workflows with which you can define a workflow ad hoc available. This task group must contain all tasks and workflows that you can use for the definition of the workflow. For further information, refer to Defining Task Groups for Ad Hoc Workflows [Seite 1099].

Procedure
6. To define a workflow ad hoc you must select a work item in the Business Workplace, which has an application object, and choose Environment → Start Workflow.

   This displays the dialog box Start Workflow. All workflows that can be started immediately are listed on the left.

7. Select Create workflow to define a workflow that uses the application object.

   This displays the Workflow Builder for creating an ad hoc workflow. The task group defined for the application object type is displayed in the tray My workflows and tasks.

8. Define the workflow.

   Use the tasks available in the tray My workflows and tasks. In the workflow definition select the position at which the task is to be inserted. Then select the task in the tray and Choose in the context menu. In the step type area there is a number of step types from the Workflow Builder available, which you can use in your workflow. For further information about the step types available and their uses, refer to Step Types and Their Symbols [Seite 1005].

9. Save your ad hoc workflow. Give your workflow an abbreviation and a name when doing so.

   You should always save the workflow as a local object. The workflow system administrator can assign the workflow to a development class at a later date as well.

10. Exit the Workflow Builder.

    This displays the dialog box Start Workflow again.

Result
Your newly created workflow is now in the table of startable workflows and is already selected. You can now start it directly.
Defining Task Groups for Ad Hoc Workflows

Use
If you want to define a workflow for an application object of a work item ad hoc, an associated task group must be defined. Only certain tasks that already exist are to be used as activities in the workflow definition. These tasks must all be based on the object type of the application object. You collect these tasks in a task group and assign this to the object type.

A workflow is to be defined ad hoc for a work item that has a notification of absence (object type FORMABSENCE) as application object. The following are tasks that deal with a notification of absence in your system:

- Approve notification of absence
- Process notification of absence
- Display notification of absence
- Delete notification of absence

Users who are allowed to create workflows ad hoc are only to use the tasks Process notification of absence and Display notification of absence. You therefore create a task group containing only the tasks Process notification of absence and Display notification of absence, and assign this task group to the object type FORMABSENCE.

Procedure
To create a task group containing appropriate tasks for a particular object type, proceed as follows:

5. Determine the tasks and workflows that refer to the object type in question. You can use the Business Workflow Explorer for this.

6. Create a new task group. For further information, refer to Definition of a Task Group.

7. Transfer all the tasks and workflows that the user who is to create workflows ad hoc is to be able to use in their workflows into the task group.

8. Assign this task group to the object type. To do this, go to the Workflow Builder and choose Extras -> Ad hoc functions -> Link object and task group.

Result
When a workflow is created ad hoc, the task group appropriate to the object type appears in the tray My workflows and tasks in the Workflow Builder.
Business Object Builder

Use
You can use the Business Object Builder [Extern] to create, display or change object types [Extern].

Integration
If you want to display or change a defined object type in the Business Object Builder, you must know its ID, name or description.
If you do not know the object type precisely or if you are not sure whether there is an appropriate object type, you can find an object type via its position in the component hierarchy or via its relationships to other object types. In this case, use the Business Object Repository Browser [Seite 1169].
Runtime objects [Extern] are based on object types.

Features
Various functions such as check, test, generate or where-used list are available. You can also create subtypes [Extern] of an existing object type or determine a delegation type [Extern].
Maintenance of Object Types

Use
The maintenance of object types [Extern] encompasses the functions of the Business Object Builder for creating, displaying and changing object types. You need these functions if you want to:

- Look at and analyze the definition and implementation of a particular object type
- Create a new object type or change an existing object type
- Change the release status of an object type

Integration
If required, you can get an overview of the existing object types including their relationships using the Business Object Repository Browser [Seite 1169]. The Business Object Repository Browser is called from the Business Object Builder.

You can use the object types supplied by SAP and extend their definitions if required. For further information, refer to Extending and Adapting Object Types [Seite 1157].

Features
You create a new object type with initially just its basic data [Seite 1103]. This assigns the appropriate position in the Business Object Repository to the object type.

You can then edit the object type components [Extern] in detail.

Activities

- To switch directly to the maintenance of an object type, you enter the ID of the object type in question and select Display or Change.
- For further information on creating a new object type, refer to Creating Object Types [Seite 1102]
- You select Business Object Repository to call the Business Object Repository Browser.
Creating Object Types

Use
There are two ways of creating a new object type:
- You create a completely new object type
- You create an object type as a subtype [Extern] of an existing object type, which inherits the components of the existing object type automatically. For further information, refer to Creating Subtypes [Seite 1158].

Procedure
   This displays the screen Business Object Builder: Initial Screen.
2. Select Create.
   The Create Object Type dialog box appears.
3. Edit the fields Object type, Object name, Name and Description. When doing so, bear in mind the general naming conventions for object types [Extern]. Leave the field Supertype blank.
4. Enter the name of a new program for the implementation of the object type. Bear in mind the naming conventions in your application.
5. Specify the code letter of your application (usually Y or Z).

Result
You have created a new object type in the Business Object Repository [Extern]. The system has put all the entries (ID, object name, name, description) into the basic data of the object type automatically.

The new object type inherits only the attributes and methods that are in the standard interface [Seite 1119], which is supported by all object types.
Editing the Basic Data of an Object Type

Use
The basic data contains both changeable, general information about the object type and information generated automatically by the system.

Prerequisites
You have selected an object type for editing in the Business Object Builder. You are on the screen Change Object Type <object type>.

Procedure
1. Select the button. The takes you to the screen Object Type <object type>: Edit Basic Data. The object type information displayed above the tab pages cannot be changed.

2. On the tab page General, edit the name and the description of the object type. Assign the object type to the data model if required, in order to define it as a business object.

3. If the object type corresponds to an SAP organizational entity, select Organizational type.


Result
The transport data defined when the object type was created is displayed on the tab page Transport data. If the object type was created as a delegation type, the delegation data is displayed on the tab page Customizing. Information about entry, last change and generation is located on the tab page Change data.
Processing Object Types

Prerequisites
To work through the following procedures, you must switch to the Business Object Builder and enter the ID of the object type to be processed in the field Object/Interface type on the Business Object Builder: Initial Screen.

Procedure
Change supertype
To change the supertype of the object type, choose Object type → Inheritance → Change supertype.

When you change the supertype, all the methods, attributes and events inherited from the supertype are also changed.

Check
To check that the implementation program with the attribute and method accesses is syntactically correct and complete with regard to the object type definition, select .
The result of the check is displayed in the status bar.

Change release status
To change the release status of the object type, choose Object type → Change release status to .
To change the release status of an object type component, proceed as follows:
1. Open the object type in change mode.
2. Select the object type component.
3. Choose Edit → Change release status → Object type component .

Generate
To generate an object type and make it available for the runtime system, select .

The system can only generate an object type if none of the superordinate object types contain errors. If these object types have not yet been generated, they are generated as well. When an object type is generated, all subtypes are generated as well provided they do not contain any errors.

Test/Execute
1. To test whether the implementation of the methods and attributes of the object type was successful, select Test.
2. If objects already exist for the object type, select an object for testing the instance-dependent attributes and methods.
   This displays the screen for testing an object type.
   If no objects exist yet for the object type, only the attributes and methods that are defined as instance-independent are offered for testing. To test the attributes and methods that
operate on an object as well, select *Create instance*. You are then asked to enter the key fields of an object of the type to be executed in order to specify it.

3. A screen with the attributes of the object and their current values is displayed. You execute the methods on this object by double-clicking on the appropriate line.

**Print**

To print the definition of the object type, select 

**Copy**

1. To copy the object type, select 
   
   The *Copy object type* dialog box appears.

2. Enter the data required and decide whether the documentation is to be copied as well.

   The new object type must be assigned to a development class or saved as a local object. The new object type hence does not necessarily belong to the same application component as the original object type.

   If the original object type was a subtype, the copy is also a subtype of the same supertype.

**Rename**

To rename the object type, select 

**Reassign**

To assign the object type to a new development class, select and enter a new development class.

**Delete**

To delete the object type, select 

   Along with the object type definition, you can also delete the program containing the implementation.

**Documentation**

You can create documentation for the object type. Proceed as follows:

1. Open the object type in change mode.

2. Select the object type ID.

3. Select .
Definition of Object Type Components

Purpose
You need object type components to work with an object type. A newly created object type only has the object type components that it inherits from the standard interface, which is supported by every object type.
To set up the object type for your tasks, you define key fields, attributes, methods and events.
You can view existing object type components to check suitability. If you establish that only a few object type components are missing, you can extend the existing object type.

Prerequisites
To define or display object type components, the object type must be created.
You must open the object type in change or display mode in the Business Object Builder.
The object type is displayed with its ID, its release status symbol and its description. The object type components are displayed in a hierarchy.
Object type components inherited from the supertype are shown on a red background. You cannot edit these object type components. The object type components that were created for this object type and that you can edit are shown on a gray background.

Process Flow
To define an object type, you do not have to edit all object type components. You should follow the sequence below:
1. You check whether any object type components required are already contained in interfaces and implement them.
2. You define the key fields.
3. You create the necessary attributes.
4. You create the necessary methods, for which you can define method parameters and exceptions.
5. You create the necessary events, for which you can create additional event parameters.
Maintenance of an Object Type Component

Use
You use these functions to process individual object type components [Extern].

Prerequisites
You are on the screen Change Object Type <object type> in the Business Object Builder.

Features

Editing an object type component
Position the cursor on an object type component and select .

If you position the cursor on the object type component name with a blue background, a tabular overview of all superordinate, defined object type components is displayed.

Creating an object type component
Position the cursor on the object type component name with a blue background and select .
You can now process the newly-created object type component directly.

Redefining an object type component
Redefining an object type component inherited from the supertype [Extern] allows you to process it.
Position the cursor on the object type component and select . A redefined object type component loses its red background.

Key fields cannot be redefined.

Deleting and renaming individual object type components
Position the cursor on the object type component to be processed. To delete it, select . To rename it, select .

You can only delete or rename object type components in the same R/3 System release.

Documentation for an object type component
If you select the ID of an object type component and choose , you can create documentation. You cannot create your own documentation for object type components of the kinds below. Instead, the documentation stored in the system is displayed.

<table>
<thead>
<tr>
<th>Kind of object type component</th>
<th>Documentation displayed</th>
</tr>
</thead>
<tbody>
<tr>
<td>Method in which a BAPI is executed</td>
<td>BAPI documentation</td>
</tr>
<tr>
<td>Attribute that references a data element</td>
<td>Data element documentation</td>
</tr>
</tbody>
</table>
### Maintenance of an Object Type Component

<table>
<thead>
<tr>
<th>Key field that references a data element</th>
<th>Data element documentation</th>
</tr>
</thead>
</table>
Displaying Details of an Object Type Component

Prerequisites
You must open the object type whose components you want to maintain in change mode in the Business Object Builder.

Procedure

Details on key fields, attributes, methods and events
1. Position the cursor on the object type component.
2. Select.

Details on method parameters and exceptions
1. Position the cursor on a method.
2. Select Parameters or Exceptions.
   The parameters or exceptions are displayed in a list.
3. Position the cursor on a line in the table.
4. Choose Edit → Choose.

Details on event parameters
1. Position the cursor on an event.
2. Select Parameters.
   The parameters are displayed in a list.
3. Position the cursor on a line in the table.
4. Choose Edit → Choose.

Implementation program
1. Position the cursor on an entry for which you want to process the implementation programming.
2. Select Program.
You go to the ABAP Editor. There you can process the program containing the implementation. If there is corresponding programming for the chosen entry, this is displayed immediately.
Implementation of Interfaces

Use
You use this function to define the interfaces that your object type supports. SAP supplies interfaces that you can use when defining your own object types. Using interfaces standardizes the definition of the object type components.

Integration
Using an interface guarantees that the object type fully supports the methods (with all parameters and all exceptions), attributes and events in the interface.

If your object type is to support an abstract interface, you must redefine and implement the inherited attributes and methods.

When redefining methods you can only change the implementation of the method in the program. The interface (parameters and exceptions) can only be extended.

Features
All objects types support the standard interface. This contains the interfaces Check existence and Display. The functionally related attributes, methods and events already defined in the interfaces are hence available to the object type.

Activities
You can add other interfaces to your object type. Position the cursor on the entry on a blue background with the component name Interfaces and select . This displays a selection list in which you can choose the interfaces your object type is to support.
Interface Aggregate

Use
You can specify the aggregate type of an object type in its basic data if the interface Aggregate is supported. The aggregate type is used to specify functional relationships between object types. There is no inheritance. The attribute Aggregate contains the reference to this aggregate type. You do not have to implement anything in the program of the object type.

⚠️
The input field in the basic data is only available if the interface Aggregate is included.

The object type OrderItem contains the interface Aggregate and the object type Order is entered as aggregate type in its basic data. This means that the attribute Aggregate is available in the object type OrderItem and contains a reference to the object type Order at runtime.

Activities
Position the cursor on the entry on a blue background with the component name Interfaces and select 📄. Enter IFAGGREGAT in the input field.
Interface Approval

Use
You use the interface Approval when you want to offer a method for your object type, which can be used in an approval task [Extern]. This interface makes wizard-driven definition of approval workflows easier.

The implementation of this method is always incomplete and not adapted to the current conditions of the object type. You must check and supplement the implementation program.

Result of this method
The method Approval has one result [Extern] with two possible values. These are specified using the fixed values for the domain SWF_APPRES:

- 0 (object approved)
- 4 (object rejected)

In the implementation of this method, the container element RESULT of the method container must be assigned one of the two values at the end.

Activities
Position the cursor on the entry on a blue background with the component name Interfaces and select 

Enter IFAPPROVE in the input field.
Interface Extended Approval

Use
You include the interface Extended approval when you want to offer a method for an object type, which can be used in an extended approval task [Extern].

The implementation of this method is always incomplete and not adapted to the current conditions of the object type. You must check and supplement the implementation program.

Result of this method
The method XApproval has one result [Extern] with four possible values. The values are specified using the fixed values for the domain SWF_XAPPRE:

- 0 (object approved)
- 1 (object proposed for approval)
- 3 (object proposed for rejection)
- 4 (object rejected)

In the implementation of this method, the container element RESULT of the method container must be assigned one of the four values at the end.

Activities
Position the cursor on the entry on a blue background with the component name Interfaces and select 

Enter IFAPPROVEX in the input field.
**Interface Create**

**Use**
When you want to offer a method for creating an object for your object type, you use the interface Create with its method Create.

The implementation of this method is always incomplete and not adapted to the current conditions of the object type. You must check and supplement the implementation program.

**Exceptions for this method**
Three exceptions are defined in the interface Create for the method Create. You must implement these exceptions at the appropriate points in the program. Use the macro following macro instruction:

```
EXIT_RETURN <Code> <Variable1> <Variable2> <Variable3> <Variable4>
```

As the <Code>, enter the exception’s 4-digit number specified when the exception was defined. The variables (maximum of four) are derived from the definition of the message that is linked to the exception.

A function is offered in the Business Workplace for creating (new) objects for a work item. The F4 input help for this function only offers the object types that support the interface Create.

**Activities**
Position the cursor on the entry on a blue background with the component name Interfaces and select . Enter IFCREATE in the input field.
Interface Display

Use
The interface Display contains the method Display for displaying an object.

![Interface Display Diagram]

The interface Display is always supported if the standard interface [Seite 1119] is supported.

The method Display is implemented for the interface in such a manner that the key field values of the object are displayed as standard. It is also entered as default method in the basic data.

![Implementation Diagram]

You should redefine the method in the program of the object type with the specific display functionality for the object type concerned so that the object is displayed with the information required.

Activities
Position the cursor on the entry on a blue background with the component name Interfaces and select . Enter IFDISPL in the input field.
Interface *Edit*

**Use**

When you want to offer a method for editing an object for your object type, the object type should support the interface *Edit* with its method *Edit*.

The implementation of this method is always incomplete and not adapted to the current conditions of the object type. You must check and supplement the implementation program.

**Exceptions for the method**

The exception *Object does not exist* is defined for the method *Edit* under the number 0001 in the interface *Edit*. You must implement this exception at the appropriate point in the program. Use the macro following macro instruction:

`EXIT_RETURN 0001 SPACE SPACE SPACE SPACE`.

**Activities**

Position the cursor on the entry on a blue background with the component name *Interfaces* and select ![ ]( ). Enter **IFEDIT** in the input field.
Interface *Check Existence*

**Use**

The method `ExistenceCheck` cannot be implemented generally for the interface *Check existence*. You must therefore implement this method for each of your object types. In the implementation, you scan the database fields concerned and check whether the object identified by the key fields actually exists.

The interface *Check existence* is always supported if the [standard interface][Seite 1119] is supported. You must redefine the method `ExistenceCheck` so that it can be used effectively.

**Exceptions for the method**

The exception *Object does not exist* is defined for the method `ExistenceCheck` under the number 0001 in the interface *Check existence*. You must implement this exception at the appropriate point in the program. Use the macro following macro instruction:

```
EXIT_RETURN 0001 SPACE SPACE SPACE SPACE.
```

Other methods (for example, the method `Find`) call the method `ExistenceCheck` in their implementations. Every object type should support this interface (and therefore this method).

**Activities**

Position the cursor on the entry on a blue background with the component name *Interfaces* and select `IFEXIST`. Enter `IFEXIST` in the input field.

For further information, refer to [Implementation of Method ExistenceCheck][Extern].
Interface Find

Use
With this method you can use the inherited source code unchanged. The source code is not entered in the
program of your object type, but only in the program of the interface Find, from which it is inherited.
The system sets up an object reference in the inherited implementation of the method Find. To do this, the
system generates a dialog box with input fields for the key fields of the object. The system also checks
within the implementation whether the key fields entered actually identify an existing object. To do this,
the method ExistenceCheck (if available) is called within the method Find.

So that the method Find really does only return objects that exist, you must also
define and implement the method ExistenceCheck (interface Check existence) for
the object type.

Exceptions for the method
Two exceptions are defined for the method Find:

- Exception 0001: Object does not exist
- Exception 1017: Cancellation by user

The exceptions are implemented in the program of the interface Find.

Activities
Position the cursor on the entry on a blue background with the component name Interfaces and select.
Enter IFFIND in the input field.
Standard Interface

Use
All object types support the standard interface and hence inherit the attribute ObjectType and the interfaces Display and Check existence. These contain the methods Display and ExistenceCheck.

Attribute ObjectType
The attribute ObjectType returns the type of an object. You do not have to implement the attribute.

Methods ExistenceCheck and Display
For further information, refer to Interface Check Existence [Seite 1117] and Interface Display [Seite 1115].

Activities
Position the cursor on the entry on a blue background with the component name Interfaces and select.
Enter IFSAP in the input field.
**Interface Status Management**

**Use**

An object type supporting this interface has methods with which a user can set the user status, and attributes with which a user can query a status.

The interface Status management must be used if you want to define object status attributes [Extern].

This interface is not available in a Basis-only system.

**Attributes StatusObjNumber, StatusObjTyp and StatusProfile**

You must implement the attributes StatusObjNumber and StatusObjTyp in order to define status attributes.

Adopt the inherited implementation of the attribute StatusProfile without changing it.

**Method UserstatusSet (set user status)**

The method UserstatusSet sets or deletes a user status without dialog. The mandatory parameter is a status in external format, meaning in the format in which the status management displays the user statuses in dialog. (The internal IDs of the user statuses cannot be seen on the user interface).

Since the external ID of the status is language-dependent, the method has an optional parameter (otherwise logon language).

**Method UserstatusSetPopup (set user status with dialog box)**

The method UserstatusSetPopup offers a dialog box with those user statuses of the status object, which have been set and which can be set. You can set or delete a status here.

Adopt the inherited implementation of the methods UserstatusSet and UserstatusSetPopup without making any changes.

**Activities**

Position the cursor on the entry on a blue background with the component name Interfaces and select ☐.

Enter IFSTATUS in the input field.

Make the following changes to the proposed source text for the two attributes:

- **Attribute** StatusObjNumber:
  
  Assign the status object number, usually derived from the status object type and the object key, to the variable OBJECT-STATUSOBJNUMBER.

- **Attribute** StatusObjTyp:
  
  Assign the status object type (3-character abbreviation) to the variable OBJECT-STATUSOBJTYPE.
Interface **Structure Object**

**Use**
The interface *Structure object* provides the attribute *Agents*. This attribute returns the positions or organizational units that are linked to an SAP organizational object.

This interface should only be used in the definition of [SAP organizational objects](Seite 1291).

You do not have to change the source text.
For more information, see [Agent as Attribute of SAP Organizational Object](Seite 1302).

**Activities**
Position the cursor on the entry on a blue background with the component name *Interfaces* and select 🔄. Enter **IFSTROBJCT** in the input field.
Defining Key Fields

Use
Objects are always identified using one or more key fields [Extern]. You specify what key information is required to uniquely identify an object of this type.

Prerequisites
You are on the screen Change Object Type <object type ID> in the Business Object Builder.

Procedure
1. Position the cursor on the component name Key fields (blue background).

   You cannot change the key fields in a subtype [Extern].

2. Select [ ]

3. Answer YES to the question Create with ABAP Dictionary field proposals?

   The Create with Data Dictionary Field Proposals dialog box appears.

4. Enter the table from which you want to reference the fields, and select the fields that are the key fields of your object type.

   Only alphanumeric data types are allowed as key fields. The total length allowed for all key fields of an object type is 70 characters.

5. Edit the ID and texts of the key field as required. Bear in mind the rules for naming object type components [Extern].

6. Select [ ]

For further information, refer to Data Declaration for Variables Used [Seite 1149].
Definition of Attributes for an Object Type

Use
You must define attributes [Extern] for an object type when you want to query particular properties, statuses or values at runtime.

Features
Attributes have various sources of information. You can split attributes according to the following sources:

- **Virtual [Extern]**
  The information is not determined until runtime, for example by calculation.

- **Database field [Extern]**
  The information is the content of a database field.

- **Object status [Extern]**
  The information is an object status.

The attribute information is returned in two different data type references [Extern]:

- As field or structure reference to a value in the ABAP Dictionary

- As **object reference [Extern]** to an object of an object type

You can declare one object type attribute as **default attribute [Extern]**. For further information, refer to Editing the Basic Data of an Object Type [Seite 1103].

After definition of an attribute, you can add source text to the implementation program [Extern] automatically. To do this, select the attribute in the list and choose **Program**.

Activities
To create an attribute, position the cursor on the component name **Attributes** (blue background), and select . Bear in mind the rules for naming object type components [Extern].

Database field attribute
Always create an attribute of this kind with a reference to a field of the ABAP Dictionary. Answer YES to the question Create with ABAP Dictionary field proposals?, and have the system make proposals for the fields.

With a database field attribute that provides the information as a field or structure reference, choose the table and field. Define whether it as a mandatory attribute.

With a database field attribute that provides the information as an object reference, choose the table, the field as source and the object type for the data type reference. Define whether it as a mandatory attribute.

The object type for the object reference can only have one key field and must be identifiable using the field. If these conditions are not met, you must define a virtual attribute.
If the attribute is derived from a field in a table whose key fields do not correspond to or only correspond partly to the key fields of the object type, the extensions in the implementation program are not sufficient. The source text contains question marks (?????????????) at the relevant points. Revise the source text at these points. Otherwise, the source text created is complete. For further information, refer to Programming Database Field Attributes [Extern].

**Virtual attribute**

With virtual attributes, it is not worth having the system make ABAP Dictionary field proposals. Answer the question with NO. A dialog box with the attribute data is displayed. Select the indicator Virtual and give the attribute an ID and texts. Define the other attribute properties and enter a data type reference. If you use the ABAP Dictionary as data type reference, only the data type of the field specified is evaluated.

Multiline attributes and attributes with a data type reference to an object type with several key fields must be implemented as virtual attributes.

The extensions in the implementation program are not sufficient. For further information, refer to Implementation Program for a Virtual Attribute [Extern].

**Object status attribute**

You can only define an object status attribute if the object type supports the interface Status management [Seite 1120]. It is not worth having the system make ABAP Dictionary field proposals. Answer the question with NO. A dialog box with the attribute data is displayed. Select the indicator Object status and give the attribute an ID and texts. Define the other attribute properties and the object status. The system enters the field OBJSTATUS from the table SWCEDITOR as data type reference. The source text created in the implementation program is complete.

**Further procedure**

Relationships between two object types are described using attributes, which return their information as object references to the other object type. In the field Inverse attribute, enter the attribute of the other object type, which contains the object reference to this object type.

Before the attribute is used, you must change its release status [Extern]. To check the attribute you must generate the object type, after which you can test it. For further information, refer to Processing Object Types [Seite 1104].

You can use macros to access the attribute. For further information, refer to Macro Instructions for Accessing Objects, Attributes and Methods [Seite 1146]
Definition of Methods for an Object Type

Use
With a method you define a function with which you can edit, create, search for or delete an object of this object type. The methods refer to ABAP functions (transactions, function modules, dialog modules, and so on) and make them available to the workflow system.

Prerequisites
Before you create a new method for an object type, check whether the desired function has already been defined as a method of an interface. If so, include this interface in the definition of your object type.

Features
You can also define the following data for each method:
- Import parameters (for synchronous and asynchronous methods)
- A result (only for synchronous methods)
- Export parameters (only for synchronous methods)
- Exceptions (only for synchronous methods)

You can declare one object type method as default method. For further information, refer to Editing the Basic Data of an Object Type.

To edit an existing method, you open it by double-clicking on the method ID.

You cannot edit a method in the list of methods displayed, which was inherited from the supertype or an interface. But you can redefine an inherited method by selecting it and choosing . You can then edit the method.

You normally change the implementation of inherited methods in the implementation program of the object type in order to adapt it to the object type. The ID of the method is not changed.

You cannot change or delete the inherited parameters and exceptions that define the interface of the method. You can, however, add further parameters and exceptions.

Activities
To create methods for an object type, position the cursor on the name Methods (blue background), and select . Bear in mind the rules for naming object type components.

If the new method is based on a function module, you should answer the question with YES. You are then supported by a wizard in the definition of the method.

If you do not set the indicator Dialog on the tab page General, the method is executed in the background at runtime. You cannot trigger any messages, error messages or warnings in the implementation of these methods, because they are not evaluated by the system and terminate the program.
If you implement a background method with a function module, you must return any exception and error situations via the interface of the function module and include them in the method exceptions.

If you set the indicator *Synchronous* on the tab page *General*, you define a synchronous method. If you do not set it, you define an asynchronous method. For further information on implementation, refer to *Programming Synchronous Methods* [Seite 1152] and *Programming Asynchronous Methods* [Seite 1150]. These two types of method differ in their feedback properties. For further information, refer to *Definition as Synchronous or Asynchronous Method* [Extern].

If you set the indicator *Result parameter* on the tab page *General*, you must specify the data type reference [Extern] of the result parameter on the tab page *Result type*.

If you set the indicator *Instance-independent* on the tab page *General*, no object reference is passed to the method when it is called. However, this method can create or establish an object (methods *Create* and *Find*). In order to inform the method caller of the reference to the object created, the key field structure or key field established in the method must be passed to the relevant variables OBJECT-KEY-... of the program. For further information on implementation, refer to *Programming Instance-Independent Methods* [Extern].

You enter the function to be executed when the method is called on the tab page *ABAP*. If a transaction is to be called, refer to the information in *Method Implementation with CALL TRANSACTION* [Extern].

If you want to define parameters or exceptions for the method, refer to the information in *Definition of Method Parameters* [Seite 1127] and *Definition of Exceptions* [Seite 1128].

The system uses all the specifications you make when defining an object method with its method parameters and exceptions in the automatic generation of the source text of the implementation program.

Define the method in full to simplify the implementation of the method in the implementation program of the object type.

For implementation examples, refer to *Implementation of Method ExistenceCheck* [Extern] and *Creating an "Execute Together" Method* [Extern].

### Further procedure

Before the method is used, you must change its release status [Extern]. To check the method you must generate the object type, after which you can test it. For further information, refer to *Processing Object Types* [Seite 1104].

You can use macros to access the method. For further information, refer to *Macro Instructions for Accessing Objects, Attributes and Methods* [Seite 1146]
Definition of Method Parameters

Use
You define method parameters for a method under the following circumstances:

- When the method requires other input values apart from the object reference, which are not requested in dialog.
- When the method returns parameters apart from the object reference.

A method often only has import parameters and no export parameters. Possible reasons for this are:

- A synchronous method returns its result via a result parameter.
- The method is asynchronous.

The reference to the object to be processed in the method is not a parameter (nor a result) of the method. The caller of the method (the workflow system in this case) actually passes the reference to the object to be processed to the method internally or retrieves it after execution of the method. (The latter is the case if the object was created by the method.)

It is quite possible for a method not to require any method parameters.

For further information, refer to Result, Export Parameters and Workflow Definition.

Integration
The following data type references exist for method parameters:

- ABAP Dictionary reference
- Object type reference

Activities
To define method parameters for a method, position the cursor on the method and select Parameters.

You must ensure that the export parameters of a method do not have the same names as the result of the method.

When you create a method parameter whose data type reference can be specified by an ABAP Dictionary table field, you should always have the system propose the relevant table fields. Answer YES to the question Create with ABAP Dictionary field proposals?

If you want to create a method parameter with an object type reference, you cannot use field proposals. Answer the question with NO.

You cannot change or delete parameters inherited with their method from a supertype or an interface.

If the method was inherited from an interface, you cannot flag import parameters created for this method as mandatory.

For further information, refer to Programming Methods with Parameters.
Definition of Exceptions

Use
You define exceptions for a method when you want to “publicize” error situations in a standardized manner in order to give the caller of the method the opportunity to react to the error situations.

Prerequisites
You can only define exceptions for synchronous methods [Extern].

Activities
To define exceptions for a method, position the cursor on the method and select Exceptions. The existing exceptions for the method are displayed and you select to create a new exception.
Give the exception an identifying number in the dialog box Add Exceptions, which is used to address it in the implementation program. The following number ranges apply:

- **0001-1000**: Exceptions defined for interfaces
- **1001-7999**: Application-specific exceptions, reserved for SAP development
- **8000-8999**: Exceptions triggered by the object manager
- **9000-9999**: Customer-defined exceptions, reserved for customers

Exceptions are classified according to error types.

- **Temporary error**: This error type indicates that certain system resources are not available at the moment, and that it is advisable to call the method again at a later point in time. Temporary errors can also occur if the method is temporarily locked by another user for processing.

- **Application error**: This error type is caused within the application called in the method. A table that cannot be maintained or a document that does not exist are typical application errors.

- **System error**: This error type shows incorrect configuration of the object manager or an inconsistency between object type definition and method call. Missing mandatory method parameters when calling a method are an example of a system error.

Specify the class and number of the message to be displayed if the exception occurs. You must specify when the exceptions are triggered in the implementation program.

You cannot change or delete exceptions inherited with their method from a supertype [Extern] or an interface [Extern].

For further information, refer to Programming Exceptions [Seite 1156].
Definition of Events for an Object Type

Use
You define events for an object type, if a status change in an object of this type is to be published across the system. Potential event receivers can then react to this event.
You use an event in a workflow under the following circumstances:

- In a workflow or task definition as a triggering event
  The workflow or task is started when the event is triggered.

- In a task definition as a terminating event
  The task is terminated when the event is triggered.

- In a workflow definition for steps of the types Event creator and Wait for event.
  The event is triggered (event creator) or the workflow is continued when the event is triggered (wait for event).

The event is not normally triggered in the implementation program, but instead by an event creation. You can use the function modules `SWE_EVENT_CREATE` or `SAP_WAPI_CREATE_EVENT` to trigger an event.

Integration
An event is published without the creating application being informed as to whether a receiver reacts to this event. The system enters potential receivers in a linkage table, which is evaluated by the event manager. Every event has an event container that contains data about the context of its creation. The data is passed to the task or workflow via a binding. For further information, refer to Binding Definitions from the Event Container.

Prerequisites
Before creating a new event, check whether the event is already defined in an interface. If so, include this interface in the definition of your object type.

Activities
To create an event, position the cursor on the name `Events` (blue background), and select.
Assign an event ID, a name and a description. Bear in mind the rules for naming object type components.

The system always enters the event ID into the event receiver table in upper case regardless of how you wrote it.
Set the indicator `Triggering object does not exist` if the object has been deleted.
The workflow system elements are in the event container as standard. If the event container is to contain other container elements, you must define parameters for this event. For further information, refer to Definition of Event Parameters.
Definition of Event Parameters

Use
You define event parameters [Extern] to supplement the system event parameters [Extern]. The event parameters are displayed as event container elements and you use them in binding definitions [Seite 1218] between the event container and the task or workflow container.

Integration
There are two different types of data type reference [Extern] for event parameters:
- ABAP Dictionary reference
- Object type reference

Activity
To create event parameters for an event, position the cursor on the event ID and select Parameters. When you create an event parameter whose data type reference can be specified by an ABAP Dictionary table field, you should always have the system propose the relevant table fields.
Answer YES to the question Create with ABAP Dictionary field proposals?
If you want to create an event parameter with an object type reference, you cannot use field proposals.
Answer the question with NO.
Programming in the Implementation Program

Purpose
The information specified about the object type components is only of a descriptive nature. Attribute accesses and method calls are programmed in the implementation program [Extern] of the object type. This program is called and executed by the system at runtime.

Process Flow

ABAP Editor
During object type definition you select Program to go to the ABAP Editor. You go directly to the lines in the implementation program that contain the relevant source text for the selected method or attribute.

Automatic generation of source text
Parts of the source text can be generated automatically to help implementation. The source text is generated based on the specifications you made when defining the object type and the object type components, and inserted into the implementation program.

For this automatic program generation to work properly, you must first define the new object type with all its object type components in full.

Manual postprocessing
When you have completed the automatic part of program generation, you need to check that the program is correct since automatic program generation can only provide templates.

Changes after program generation
Changes made to object type components do not automatically change the program. This means that you must incorporate all subsequent changes made to attributes and methods manually into the implementation program.
Only in the following operations is the source text changed automatically in the implementation program:

- Deletion of methods
  The associated source text is deleted.

- Deletion of virtual attributes
  The associated source text is deleted.

- Renaming of methods
  The method ID in the source text is changed.

- Renaming of virtual attributes
  The attribute ID in the source text is changed. However, the use of the OBJECT-<ATTRIBUTE> variables must be adapted manually.

Including macro instructions
Macro instructions for processing containers and accessing objects are used in the implementation program. The macro instructions are available if the macro <OBJECT> has been included into the implementation program with the INCLUDE command:

* Include macro instructions
********************************************************************************

INCLUDE <OBJECT>.

The source text above is incorporated already with automatic program generation.
You can use the macro instructions for processing the container [Seite 1133] outside the implementation program as well. You can also process the container of the implementation program. This is always called CONTAINER and has the following container elements:
The method parameters [Extern] and result [Extern] for all methods
All virtual attributes [Extern]
The IDs of the container elements are the same as those of the parameters and attributes. The result is stored in the container element RESULT. You must access the container with these macro instructions under the following circumstances:
- To read the import parameters of a method
- To write the export parameters and the result of a synchronous method
- To read and write the virtual attributes
The macro instructions for accessing objects, attributes and methods [Seite 1146] can only be used within the implementation program.
Macro Instructions for Processing a Container

Use

You can use these macro instructions for the following programs:
- Implementation programs of object types
- Function modules for event creation
- Function modules for role resolution

Processing a container instance includes:
- Accessing the value of a container element
- Entering a value for a container element

Only the columns Element ID and Value are relevant for this.

When values are assigned to the container no check is made as to whether the elements entered and the data type of their values are in accordance with the container definition. You must ensure that the element IDs and the data types that you use are in accordance with the definition.

A container element ID is always managed internally in upper case. It does not matter whether you use upper or lower case when calling the macro instructions, because the element ID is always converted to upper case before access.

Prerequisites

The include file <CNTN01> must have been included into the program in order to use these macros. Use the following instruction to do this:

INCLUDE <CNTN01>.

Since the include file <OBJECT> already contains the include file <CNTN01>, the include file <CNTN01> cannot be included in an implementation program again.

The container must be declared and initialized with the following macro instructions:

SWC_CONTAINER <Container>. "Declaration
SWC_CREATE_CONTAINER <Container>. "Initialization

For further information, refer to Declaration and Initialization of a Data Structure of a Container [1137].

Features

You can assign the content of a field to a container element or write the content of a container into a field. This content can be multiline, that is, in the form of an internal table [Extern]. The content of a field can be an elementary data type [Extern], a structure [Extern] or an object reference [Extern].

An object reference refers to an object of an object type defined in the Business Object Repository. The key required for unique identification of the object is described there. The key is comprised of one or more key fields. Key fields are, for example, an order number or company code, document number and posting year, which together form a key.

Within the transaction for creating an order, an event is triggered when this order has been successfully created.

The object created is to be made known to a potential receiver of this event. You therefore write a reference to this object described by the key fields of the object type Order into the event container.

Entries for a particular material in the material master data are to be maintained in a workflow step. The role "material administrator" is to have responsibility for this activity.
Macro Instructions for Processing a Container

The role resolution is carried out by a function module provided by the application on the basis of the information about the material. The workflow system passes a container to this function module containing the object reference to the object of the type Material. Within the function module, the object reference is read from the container and the evaluation for determining the selected agent(s) carried out.

Activities

The macros required for reading and writing and their call procedures differ with the content of the container element. Deleting and copying container elements does not vary.

Deleting a container element

SWC_DELETE_ELEMENT <Container> <ContainerElement>.

If the container element does not exist in the container, the error code 1 is returned with SY-SUBRC, otherwise SY-SUBRC has the value 0.

Copying a container element

SWC_COPY_ELEMENT <SourceContainer> <SourceElement> <TargetContainer> <TargetElement>.

The container element is copied into the target container. If a container element with the same name exists there it is overwritten, otherwise a new container element is created.

Elementary type or structure

Use the macros below to read structures or single variables from a container or to write them into a container.

Writing a field value

SWC_SET_ELEMENT <Container> <ContainerElement> <Value>.

For further information, refer to Writing a Field Value into a Container [Seite 1138].

Reading a field value

SWC_GET_ELEMENT <Container> <ContainerElement> <FieldVariable>.

If the container element does not exist in the container, the error code 1 is returned with SY-SUBRC, otherwise SY-SUBRC has the value 0.

Multiline elementary type or multiline structure

Use the macros below to read multiline structures or multiline variables from a container or to write them into a container.

Writing a multiline field value

SWC_SET_TABLE <Container> <ContainerElement> <Value>.

For further information, refer to Writing a Table into a Container [Seite Error! Bookmark not defined.].

Reading a multiline field value

SWC_GET_TABLE <Container> <ContainerElement> <TableVariable>.

If the container element does not exist in the container, the error code 1 is returned with SY-SUBRC, otherwise SY-SUBRC has the value 0.

Object reference as container element

Use the macros below to read object references from a container or to write them into a container. When an object reference is written into a container, the system:

- Checks whether the associated object type is defined and activated in the Business Object Repository.
- Sets an indicator in the container denoting that the entry is an object reference.

Writing an object reference

You declare and create an object reference and then write it into the container.
Macro Instructions for Processing a Container

DATA <Object> TYPE SWC_OBJECT.
SWC_CREATE_OBJECT <Object> <ObjectType> <ObjectKey>.
SWC_SET_ELEMENT <Container> <ContainerElement> <Object>.
For further information, refer to Writing an Object Reference into a Container [Seite 1140].

Reading an object reference
You declare an object reference, fill it from the container, and can then establish the associated object type and key from the object reference.
DATA <Object> TYPE SWC_OBJECT.
SWC_GET_ELEMENT <Container> <ContainerElement> <Object>.
SWC_GET_OBJECT_KEY <Object> <ObjectKey>.
SWC_GET_OBJECT_TYPE <Object> <ObjectType>.
For further information, refer to Reading an Object Reference from a Container [Seite 1142].

Multiline object reference as container element
Use the macros below to read multiline object references from a container or to write them into a container. When an object reference is written into a container, the system:
- Checks whether the associated object type is defined and activated in the Business Object Repository.
- Sets an indicator in the container denoting that the entry is an object reference.

Writing a multiline object reference
You declare a multiline object reference and a single object reference. You then create each object reference to be inserted into the table individually, append them to the table and finally write the table into the container.
DATA <ObjectList> TYPE SWC_OBJECT OCCURS 0.
DATA <Object> TYPE SWC_OBJECT.
SWC_CREATE_OBJECT <Object> <ObjectType> <ObjectKey>.
APPEND <Object> TO <ObjectList>.
...
SWC_SET_TABLE <Container> <ContainerElement> <ObjectList>.
For further information, refer to Writing a Multiline Object Reference into a Container [Seite 1143].

Reading a multiline object reference
You declare a multiline object reference and fill it from the container.
DATA <ObjectList> TYPE SWC_OBJECT OCCURS 0.
SWC_GET_TABLE <Container> <ContainerElement> <ObjectList>.
For further information, refer to Reading a Multiline Object Reference from a Container [Seite 1145].
Declaration and Initialization of a Container

Use
You can use macros to create or initialize a container at runtime.
A container has to be created if you program the event creation yourself and require an event container, for example.
You do not have to create a container in the following programs:

- Function module for role resolution
  The container is declared in the interface of the function module.
- Implementation program [Extern]
  The container is called CONTAINER and is already created.

Features

Declaring a container data structure at runtime
An empty internal table [Extern] is created with the structure SWCONT. Use the macro following macro instruction:
SWC_CONTAINER <Container>.

Initializing a container
You must initialize a container before you use it. This deletes all entries in the internal table. Use the macro following macro instruction:
SWC_CREATE_CONTAINER <Container>.

The macros SWC_RELEASE_CONTAINER and SWC_CLEAR_CONTAINER have the same functionality.
Writing a Field Value into a Container

Prerequisites
The following prerequisites must be fulfilled in order to write a field value into a container:

- The container must have been created and initialized [Seite 1137].
- The include file <CNTN01> must have been included into the program.

These prerequisites are always fulfilled if you are in the implementation program [Extern]. The container is then addressed with the name CONTAINER.

Procedure
1. Create a variable for the field value. Specify a database field as reference.
   
   If the container is used in a workflow binding, you must ensure that the field value has the same data type reference [Extern] as the container element defined in the container definition.

2. Assign a value to the variable.

3. Write the field value into the container using the following macro instruction:
   
   SWC_SET_ELEMENT <Container> <ContainerElement> <Value>.

   Upper and lower case are not distinguished for the container element ID. If the container is used in the binding of a workflow, you must ensure that a container element of the same name is defined in the container definition.

Result
If the container element with this ID is not yet in the container, it is inserted into the container with its value. If there is already a container element with this ID in the container, the old value is overwritten with the new value.

Example
This example is part of the implementation program for object type BUS1001.

* Type declarations
DATA:
  ENDLEADTME LIKE BAPICM61M-WZTER.
A value is assigned to the field ENDLEADTME when a function is called. The value in the field ENDLEADTME is written with the container element ID Endleadtme into the container CONTAINER.
SWC_SET_ELEMENT CONTAINER 'Endleadtme' ENDLEADTME.
Writing a Table into a Container

Prerequisites
The following prerequisites must be fulfilled in order to write a multiline field value into a container:

- The container must have been created and initialized [Seite 1137].
- The include file <CNTN01> must have been included into the program.

These prerequisites are always fulfilled if you are in the implementation program [Extern]. The container is then addressed with the name CONTAINER.

Procedure
1. Create a variable for the table.
   
   If the container is used in a workflow binding, you must ensure that the table has the same data type reference [Extern] as the container element defined in the container definition.

2. Assign values to the table.

3. Write the table into the container using the following macro instruction:
   
   SWC_SET_TABLE <Container> <ContainerElement> <Value>.

   Upper and lower case are not distinguished for the container element ID. If the container is used in the binding of a workflow, you must ensure that a container element of the same name is defined in the container definition.

   To append a value to a table, you must first read the table from the container, then append the value, and finally write the table back into the container.

Result
If the container element with this ID is not yet in the container, it is inserted into the container with its value. If there is already a container element with this ID in the container, the old value is overwritten with the new value.
Writing an Object Reference into a Container

Prerequisites
The following prerequisites must be fulfilled in order to write an object reference into a container:

- The container must have been created and initialized. (Seite 1137)
- The include file <CNTN01> must have been included into the program.

These prerequisites are always fulfilled if you are in the implementation program [Extern].

- The object key must be available in a variable (designated as Object key).

Procedure

1. Create a variable for the object reference. Use the following instruction to do this:
   
   `DATA <Object> TYPE SWC_OBJECT.`

2. Create the object reference. Use the following instruction to do this:
   
   `SWC_CREATE_OBJECT <Object> <ObjectType> <ObjectKey>.

3. Write the object reference into the container using the following macro instruction:
   
   `SWC_SET_ELEMENT <Container> <ContainerElement> <Object>.

   Upper and lower case are not distinguished for the container element ID. If the container
   is used in the binding of a workflow, you must ensure that a container element of the
   same name is defined in the container definition.

4. Repeat steps 2 and 3 until the container is filled.

5. Make the object references of the container persistent using the following macro instruction:
   
   `SWC_CONTAINER_TO_PERSISTENT <Container>.

   You need persistent object references if the container is also to be used in another
   program context. This applies if you pass the container as a parameter in a function call,
   for example if:
   - You want to evaluate or change the container as an event container in a check
     function module or receiver type function module
   - You prepare the container as an event container and use it as a parameter of the
     function modules SWE_EVENT_CREATE or SAP_WAPI_CREATE_EVENT.

Result

The object reference created is a runtime reference that only exists in the environment of the creating
program. This object reference is written into the container in a container element. If the container
element with this ID is not yet in the container, it is inserted into the container with its value. If there is
already a container element with this ID in the container, the old value is overwritten with the new
value. If the container is needed in another program context, you must make the object references
contained persistent.

Example

An object of the type KNA1 (customer) is described as a key field via the customer number. The
customer number is under KUNNR in the table KNA1 (customer master: general section).
* Type declaration for key field
  * Object: Customer
    DATA KUNNR LIKE KNA1-KUNNR.
    "Customer number"
A value is assigned to the field KUNNR:
  * Customer number
    KUNNR = '000004711'.
The variable CUSTOMER is declared as an object reference.
  * Type declarations for object
    DATA CUSTOMER TYPE SWC_OBJECT.
An object reference is created for the object type KNA1 (customer) mentioned above.
  * Create object reference for object type KNA1
    SWC_CREATE_OBJECT CUSTOMER 'KNA1' KUNNR.
The customer is entered into the container MY_CONTAINER in the container element Customer.
  * Write object reference into container
    SWC_SET_ELEMENT MY_CONTAINER 'Customer' CUSTOMER.
The object references of the container are made persistent.
    SWC_CONTAINER_TO_PERSISTENT MY_CONTAINER.
Reading an Object Reference from a Container

Prerequisites
The following prerequisites must be fulfilled in order to read an object reference from a container:

- The include file `<CNTN01>` must have been included into the program.
- These prerequisites are always fulfilled if you are in the implementation program [Extern]. The container is then addressed with the name CONTAINER.
- The object references of the container must not be persistent.
- Containers passed via a function module interface always contain persistent object references. Before you read object references from a container and store them in a variable, you must make all object references in the container into runtime references. To do this, execute the following macro instruction:
  
  ```
  SWC_CONTAINER_TO_RUNTIME <Container>.
  ```

You must process a container with this macro if you are:
- Reading a role container for role resolution
- Reading an event container within a check function module [Extern] or a receiver type function module [Extern]

Procedure
12. Create a variable for the object reference. Use the following instruction to do this:
  
  ```
  DATA <Object> TYPE SWC_OBJECT.
  ```

13. Read the object reference from the container. Use the following instruction to do this:
  
  ```
  SWC_GET_ELEMENT <Container> <ContainerElement> <Object>.
  ```

You can read the object type and the key from the object reference. To do this, use the macros `SWC_GET_OBJECT_TYPE` and `SWC_GET_OBJECT_KEY`. For further information, refer to Macro Instructions for Accessing Objects, Attributes and Methods [Seite 1146]

If you only want to read one object reference, you should also only convert this object reference into a runtime reference for performance reasons. To do this, proceed as follows:

```
DATA <Object> TYPE SWC_OBJECT
  <Variable> LIKE SWOTOBJID.
SWC_GET_ELEMENT <Container> <ContainerElement> <Variable>.
SWC_OBJECT_FROM_PERSISTENT <Variable> <Object>.
```
Writing an Multiline Object Reference into a Container

Prerequisites
The following prerequisites must be fulfilled in order to write a multiline object reference into a container:
The container must have been created and initialized. The include file CNTN01 must have been included into the program.

These prerequisites are always fulfilled if you are in the implementation program [Extern]. The container is then addressed with the name CONTAINER.
The key of the objects must be available in a variable (designated as Object key).

Procedure
Create a variable for the multiline object reference. Use the following instruction to do this:
DATA <ObjectList> TYPE SWC_OBJECT OCCURS 0.
The internal table created does not have a table header. You fill it by creating each object reference individually and appending them to the table.
Create a variable for an object reference. Use the following instruction to do this:
DATA <Object> TYPE SWC_OBJECT.
Create the first object reference. Use the following instruction to do this:
SWC_CREATE_OBJECT <Object> <ObjectType> <ObjectKey>.
Append the object reference to the table. Use the following instruction to do this:
APPEND <Object> TO <ObjectList>.
Repeat the previous two steps until the table is full.
Write the multiline object reference into the container. Use the following instruction to do this:
SWC_SET_TABLE <Container> <ContainerElement> <ObjectList>.

Upper and lower case are not distinguished for the container element ID. If the container is used in the binding of a workflow, you must ensure that a container element of the same name is defined in the container definition.

Make the object reference persistent. Use the following instruction to do this:
SWC_CONTAINER_TO_PERSISTENT <Container>.
You need persistent object references if the container is also to be used in another environment. This applies if you pass the container as a parameter in a function call, for example if:
You want to evaluate or change the container as an event container in a check function module [Extern] or receiver type function module [Extern]
You prepare the container as an event container and use it as a parameter of the function modules SWE_EVENT_CREATE or SAP_WAPI_CREATE_EVENT

Result
The object references created are runtime references that only exist in the environment of the creating program. These object references are written into the container in a container element. If the container element with this ID is not yet in the container, it is inserted into the container with its value. If there is already a container element with this ID in the container, the old value is overwritten with the new
value. If the container is needed in another program context, you must make the object references contained persistent.

**Example**

The variable POSITIONLIST (item list) is declared as a multiline object reference.

* Type declarations for multiline object reference

```plaintext
DATA POSITIONLIST TYPE SWC_OBJECT OCCURS 0.
```

The variable POSITION (item) is declared as an object reference. This variable serves as an auxiliary variable in the processing of the multiline object list.

* Type declarations for object reference

```plaintext
DATA POSITION TYPE SWC_OBJECT.
```

Values are assigned to the internal table POSITIONKEY (item key). An object reference is created in the variable POSITION for each entry and written into the multiline object reference POSITIONLIST.

* Type declarations for key fields of
  * object "order item"

```plaintext
DATA: BEGIN OF POSITIONKEY,
  DOCUMENT LIKE VBAP-VBELN, "Sales document"
  ITEM     LIKE VBAP-POSNR, "Sales document item"
END OF POSITIONKEY.
```

* Process multiline object reference

```plaintext
SELECT * FROM VBAP
  WHERE VBELN BETWEEN '0000002500' AND '0000002600'.
```

* Fill key fields

```plaintext
POSITIONKEY-DOCUMENT = VBAP-VBELN.
POSITIONKEY-ITEM     = VBAP-POSNR.
```

* Create object reference

```plaintext
SWC_CREATE_OBJECT POSITION 'VBAP' POSITIONKEY.
```

* Insert object reference into multiline object reference

```plaintext
APPEND POSITION TO POSITIONS.
```

The multiline object reference POSITIONLIST is written into the container MY_CONTAINER in the container element Items.

* Write multiline object reference into container

```plaintext
SWC_SET_TABLE MY_CONTAINER 'Items' POSITIONS.
```

For another example, refer to [Implementation Program for a Virtual Attribute [Extern]].
Reading a Multiline Object Reference from a Container

Prerequisites

The following prerequisites must be fulfilled in order to read an object reference from a container:
The include file <CNTN01> must have been included into the program.

This prerequisite is always fulfilled if you are in the implementation program [Extern]. The container is then addressed with the name CONTAINER.

The object references of the container must not be persistent.

Containers passed via a function module interface always contain persistent object references. Before you read an object reference from a container and store it in a variable, you must make all object references in the container into runtime references. To do this, execute the following macro instruction:

SWC_CONTAINER_TO_RUNTIME <Container>

You must process a container with this macro if you are:

- Reading a role container for role resolution.
- Reading an event container within a check function module [Extern] or a receiver type function module [Extern].

Procedure

Create a variable for the multiline object reference. Use the following instruction to do this:

DATA <ObjectList> TYPE SWC_OBJECT OCCURS 0.

Create a variable for an object reference. Use the following instruction to do this:

DATA <Object> TYPE SWC_OBJECT.

Read the multiline object reference from the container. Use the following instruction to do this:

SWC_GET_TABLE <Container> <ContainerElement> <ObjectList>.

To access the individual object references, read the individual object references from the multiline object reference in a loop. Use the following instruction to do this:

READ TABLE <ObjectList> <Index> <Object>.

You can read the object type and the key from the object reference. To do this, use the macros SWC_GET_OBJECT_TYPE and SWC_GET_OBJECT_KEY. For further information, refer to Macro Instructions for Accessing Objects, Attributes and Methods [Seite 1146]
Macro Instructions for Accessing Objects, Attributes and Methods

Use
You can use these macros for simplifying access to Business Object Repository objects.

Prerequisites
The include file <OBJECT> must be incorporated into the program in order to use these macros.

Features
You must declare variables in which an object reference is to be stored with the following instruction:
DATA: <Object> TYPE SWC_OBJECT.

General macros
SWC_CREATE_OBJECT <Object> <ObjectType> <ObjectKey>.
Creates an object reference for an object key of an object type.
You must pass the object-specific key in <ObjectKey> and the ID of the object type in <ObjectType>. You must pass <ObjectType> either as a character string or as a variable declared with LIKE SWOTOBJID-OBJTYPE.
The object reference created is returned in the variable <Object>. You must declare <Object> with type SWC_OBJECT.

If the variable <ObjectKey> has the value SPACE, a reference to the object type is returned in <Object>. You can use this to query properties of the object type (attributes, methods, events).

SWC_REFRESH_OBJECT <Object>.
Object attributes that reference a database field are read from the database again when next accessed.
You must pass the object reference in <Object>. You must declare <Object> with type SWC_OBJECT.

You should use this macro instruction after execution of methods that can change attributes (for example Edit, Update, Change).

SWC_GET_OBJECT_TYPE <Object> <ObjectType>.
The object type for the object reference is established.
You must pass the object reference in <Object>. You must declare <Object> with type SWC_OBJECT.
The object type is returned in the variable <ObjectType>. You can declare <ObjectType> with LIKE SWOTOBJID-OBJTYPE.

SWC_GET_OBJECT_KEY <Object> <ObjectKey>.
The object key for the object reference is established.
You must pass the object reference in <Object>. You must declare <Object> with type SWC_OBJECT.
The object key is returned in the variable <ObjectKey>. You can declare <ObjectKey> with LIKE SWOTOBJID-OBJKEY.

SWC_CALL_METHOD <Object> <Method> <Container>.
The method defined for an object type is executed on an object of this object type.
You must pass the object reference in <Object> and the method to be executed in <Method>. You must declare <Object> with type SWC_OBJECT.
Macro Instructions for Accessing Objects, Attributes and Methods

The import parameters of the method are passed in the variable <Container>. The result and the export parameters of the method executed are also stored in <Container>. If the method called raises an exception, the number of the exception is stored in the system variable SY-SUBRC and any values in the variables SY-MSGV1, ..., SY-MSGV4.

An example of a call of this kind can be found in Programmed Call of Method [Extern].

If you pass SPACE for <Method>, the default method [Extern] is executed.

You must pass a container even if no parameters are defined for the method called. Therefore you create and initialize an empty container with the following macro instructions:

```
SWC_CONTAINER <Container>.
SWC_CREATE_CONTAINER <Container>.
```

```
SWC_GET_PROPERTY <Object> <Attribute> <AttributeValue>.
SWC_GET_TABLE_PROPERTY <Object> <Attribute> <AttributeValue>.
```

The value of the specified attribute [Extern] is established. If it is a virtual attribute [Extern], this is done dynamically.

You must pass the object reference in <Object> and the attribute or key field to be read in <Attribute>. You must declare <Object> with type SWC_OBJECT.

The value of the attribute or key field is returned in the variable <AttributeValue>.

```
SWC_GET_PROPERTY returns a single value, SWC_GET_TABLE_PROPERTY returns a multiline value in an internal table.
```

An example of a call of this kind can be found in Programmed Access to an Attribute [Extern].

Executing macros on own object

All the macros in the section General macros can also be executed directly on the object itself in the implementation program. To do this, you replace <Object> with SELF in the macro calls.

```
Attribute values of the current object can be read by calling SWC_GET_PROPERTY
SELF <Attribute> <AttributeValue>.
```

Setting an object key

When implementing methods with which an object is created, the object key must be set. The typically applies to the methods Create and Find. Use the macro following macro instruction:

```
SWC_SET_OBJECTKEY <ObjectKey>.
```

The object key to be set is passed in the variable <ObjectKey>. An example of a call of this kind can be found in Programming Instance-Independent Methods [Extern].

Raising exceptions

Within the implementation program, you can raise the exceptions defined for a method. Use the macro following macro instruction:

```
EXIT_RETURN <Exception> <Var1> <Var2> <Var3> <Var4>.
```

Pass the number of the exception in <Exception> and the four task parameters displayed in <VarN>. The parameters can be SPACE. <Exception> must be declared with LIKE SWOTINVOKE-CODE.

For an example, refer to Implementation of Method ExistenceCheck [Extern].
Data Declaration for Variables Used

Use
The system automatically creates an area for the data declaration for key fields, attributes and method parameters in the implementation program. This area is between the two macro instructions BEGIN_DATA OBJECT and END_DATA OBJECT at the beginning of the source text. The data structure of the object key is between the macro instructions BEGIN OF KEY and END OF KEY and is created automatically by the system. The method parameters and attributes are then declared.

Activities
The data declaration is supplemented when the implementation program is first called after you have created the object type component.

⚠️ Since the system obtains the information about the names of the variables and the field references directly from the definition of the individual object type components, you cannot make any changes here.

Example
The data declaration for the object type BUS1001 (Material) is used here as an example for data declaration:
BEGIN_DATA OBJECT. " Do not change.. DATA is generated
* only private members may be inserted into structure private
DATA:
" begin of private,
" to declare private attributes remove comments and
" insert private attributes here ...
" end of private,
BEGIN OF KEY,
MATERIAL LIKE MARA-MATNR,
END OF KEY,
PURCHASEREQUISITION TYPE SWC_OBJECT OCCURS 0,
PURCHASEORDER TYPE SWC_OBJECT OCCURS 0,
PURCHASINGINFO TYPE SWC_OBJECT OCCURS 0,
PURCHOUTLINEAGREEMT TYPE SWC_OBJECT OCCURS 0,
DOCUMENT TYPE SWC_OBJECT OCCURS 0,
COSTESTIMATE TYPE SWC_OBJECT OCCURS 0,
PIECEOFEQUIPMENT TYPE SWC_OBJECT OCCURS 0,
_MAKT LIKE MAKT,
_MARA LIKE MARA.
END_DATA OBJECT. " Do not change.. DATA is generated
Programming Asynchronous Methods

You program an asynchronous method in the implementation program between the macro instructions BEGIN_METHOD <Method> and END_METHOD. The system generates an implementation from the specifications you made when defining the method. The unique ID of the object is available in the structure of the key fields under the variable OBJECT-KEY.

An asynchronous method can be implemented with a transaction. The transaction is called in the program with the ABAP command CALL TRANSACTION. The input fields in the first screen of the transaction are assigned values from the key fields of the object and possibly from the import parameters of the method via processing parameters (“SET/GET parameters”). This first screen is then skipped when the transaction is called(... AND SKIP FIRST SCREEN).

With asynchronous methods you cater for feedback to the method caller via terminating events [Extern].

Asynchronous Method Call

Workflow execution

Work item manager

• Executes a workflow step
• Calls the asynchronous method stored

E.g. QM task: Process notification (TS24500066):
Object type: BUS2078
Method: EditAsynchron
BEGIN_METHOD editasynchron changing
SWC_GET_ELEMENT...
SWC_GET_ELEMENT...
CALL TRANSACTION ... AND SKIP FIRST SCREEN.
END_METHOD.

Application

Object manager

Object type: BUS2078
Method: EditAsynchron
BEGIN_METHOD editasynchron changing container.
  
  SWC_SET_ELEMENT...
  SWC_SET_ELEMENT...
  CALL TRANSACTION ... AND SKIP FIRST SCREEN.
END_METHOD.

Example

Implementing the method EditAsynchronous (Change quality notification asynchronously).
The method operates on an object of the type BUS2078 (quality notification). This object type has the key field Number (notification number). This method has no import parameters.

You implement this method by calling transaction QM02.

When called, the transaction requires the sales document that is available as key of the object from the variable OBJECT-KEY-NUMBER.

* Method Edit (simplified)

BEGIN_METHOD EditAsynchronous CHANGING CONTAINER.
  SET PARAMETER ID 'IQM' FIELD OBJECT-KEY-NUMBER.
  CALL TRANSACTION 'QM02' AND SKIP FIRST SCREEN.
END_METHOD.

The system automatically generates the method implementation described above from the specifications you made when defining the method.
Programming Synchronous Methods

You program a synchronous method in the implementation program between the macro instructions BEGIN_METHOD <Method> and END_METHOD. The unique ID of the object is available in the structure of the key fields under the variable OBJECT-KEY.

A synchronous method can be implemented with a function module. The function module is called with the ABAP command CALL FUNCTION in the program. The application called from the method should contain no COMMIT WORK and therefore no update task call for synchronous methods.

In the case of synchronous methods, a result [Extern] and/or the export parameters [Extern] can be returned to the caller if you pass the result to the container CONTAINER within the method implementation. To do this, use the macro instruction SWC_SET_ELEMENT for a single-line result or export parameter, or the macro instruction SWC_SET_TABLE for a multiline result or export parameter.

The result is always put into the container element RESULT. The export parameters are put into the container with their IDs. The result is available in the task container in the container element _WI_RESULT.

Synchronous Method Call

Example

Implementing the synchronous method Approve, which returns a result:
The method operates on an object of the object type FORMABSEN (form: notification of absence). This object type has the key field Number. This method has one exception an no import parameters.
You implement this method by calling the function module SWX_FORMABS_APPROVE. The result of the method, which can be derived from the export parameter of the function module, has the reference field PROCSTATE from the reference table SWXFORMABS as data type reference.

* Method Approve

BEGIN_METHOD APPROVE CHANGING CONTAINER.
DATA:
  APPROVE LIKE SWXFORMABS-PROCSTATE.
CALL FUNCTION 'SWX_FORMABS_APPROVE' 
  EXPORTING
    FORMNUMBER = OBJECT-KEY-NUMBER
  IMPORTING
    PROC_STATE = PROC_STATE
  EXCEPTIONS
    DOCUMENT_NOT_FOUND = 01
END_METHOD.
ABORTED = 02
OTHERS = 03.
CASE SY-SUBRC.
WHEN 00.
  * If successful: Assignment of result
  * to element RESULT in container CONTAINER
  *******************************************************
    SWC_SET_ELEMENT CONTAINER RESULT APPROVE.
WHEN 01.
    EXIT_RETURN 1301 SPACE SPACE SPACE SPACE.
WHEN 02.
    EXIT_CANCELED.
WRITE OTHERS. "to be implemented".
ENDCASE.
END_METHOD.

An exception is defined for this object method with the number 1301 (document not found), which you raise with the macro instruction EXIT_RETURN.

With the macro instruction EXIT_CANCELED, you can leave the method implementation if a user cancels. A single-step task in which this method is referenced is not yet completed in this case. The relevant work item retains the status In process.

If you have fully maintained the ABAP function type of the method and the method parameters, the system implements the call to the relevant function and the data declaration of the method parameters used.
Programming Methods with Parameters

Use

Import parameters of the method are read in the program of the object type from the container CONTAINER using the macros SWC_GET_ELEMENT (for single-line elements) and SWC_GET_TABLE (for multiline elements), and passed on to the function module or the processing parameters ("SET/GET parameters") of a transaction.

Within method implementation in the program of the object type, export parameters are put into the container CONTAINER. The macro instructions SWC_SET_ELEMENT or SWC_SET_TABLE are used to do this for single-line export parameters or a multiline export parameters respectively.

Example

Implementing the method GetInternalNumber (assign new internal numbers).

The method operates on an object of the object type BUS1001 (material). This object type has the key field Material (material number). The method has the following parameters:

<table>
<thead>
<tr>
<th>Parameters</th>
<th>Description</th>
<th>Type</th>
</tr>
</thead>
<tbody>
<tr>
<td>MaterialType</td>
<td>Material type</td>
<td>Import</td>
</tr>
<tr>
<td>IndustrySector</td>
<td>Sector</td>
<td>Import</td>
</tr>
<tr>
<td>RequiredNumbers</td>
<td>Amount of material numbers required</td>
<td>Import</td>
</tr>
<tr>
<td>Return</td>
<td>Return parameter</td>
<td>Export</td>
</tr>
<tr>
<td>MaterialNumber</td>
<td>Assigned material numbers</td>
<td>Import/export</td>
</tr>
</tbody>
</table>

You implement the method by calling the API function BAPI_MATERIAL_GETINTNUMBER.

BEGIN_METHOD GETINTERNALNUMBER CHANGING CONTAINER.
DATA:
  MATERIALTYPE LIKE BAPIMATDOA-MATL_TYPE,
  INDUSTRYSECTOR LIKE BAPIMATDOA-IND_SECTOR,
  REQUIREDNUMBERS LIKE BAPIMATALL-REQ_NUMBERS,
  RETURN LIKE BAPIRETURN1,
  MATERIALNUMBER LIKE BAPIMATINR OCCURS 0.
SWC_GET_ELEMENT CONTAINER 'MaterialType' MATERIALTYPE.
SWC_GET_ELEMENT CONTAINER 'IndustrySector' INDUSTRYSECTOR.
IF SY-SUBRC <> 0.
  MOVE SPACE TO INDUSTRYSECTOR.
ENDIF.
SWC_GET_ELEMENT CONTAINER 'RequiredNumbers' REQUIREDNUMBERS.
IF SY-SUBRC <> 0.
  MOVE 1 TO REQUIREDNUMBERS.
ENDIF.
CALL FUNCTION 'BAPI_MATERIAL_GETINTNUMBER'
  EXPORTING
    REQUIRED_NUMBERS = REQUIREDNUMBERS
    INDUSTRY_SECTOR = INDUSTRYSECTOR
    MATERIAL_TYPE = MATERIALTYPE
  IMPORTING
    RETURN = RETURN
  TABLES
    MATERIAL_NUMBER = MATERIALNUMBER
EXCEPTIONS
  OTHERS = 01.
CASE SY-SUBRC.
  WHEN 0.            " OK
    WHEN OTHERS.     " to be implemented
ENDCASE.
SWC_SET_ELEMENT CONTAINER 'Return' RETURN.
SWC_SET_TABLE CONTAINER 'MaterialNumber' MATERIALNUMBER.
END_METHOD.
Programming Exceptions

Use
You implement exceptions within a method, that is within the macro instructions BEGIN_METHOD <Method> and END_METHOD, using the macro instruction EXIT_RETURN.

EXIT_RETURN <Code> <Variable1> <Variable2> <Variable3> <Variable4>.

You must declare <Code> with the type SWOTINVOKE-CODE. For Code, enter the exception’s four-digit number specified when the exception was defined. The variables (maximum of four) are derived from the definition of the message that is linked to the exception.

For examples of implementing exceptions, refer to Implementation of Method ExistenceCheck [Extern].

Standard exceptions
As an alternative to the exceptions defined for the method and raised with the EXIT_RETURN macro described above, you can also use one of the macro instructions below to leave the program in the event of exception situations.

⚠️ You cannot model in a workflow definition for the exceptions you implement with the macro instructions below because they do not appear there for selection.

<table>
<thead>
<tr>
<th>EXIT_OBJECT_NOT_FOUND</th>
<th>Object not found</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Do not use this macro instruction in implementation of the method ExistenceCheck.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>EXIT_CANCELLED</th>
<th>Canceled by user</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>If the method executed in a work item raises this exception, the work item is not yet completed. Instead it retains the status In process and can be executed again.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>EXIT_NOT_IMPLEMENTED</th>
<th>Method not implemented</th>
</tr>
</thead>
</table>

| EXIT_PARAMETER_NOT_FOUND | Mandatory parameter not found |

You can only use the macro instructions above within BEGIN_METHOD - END_METHOD.
Extending and Adapting Object Types

Purpose
You can extend an object type to add more object type components not in the standard version. This ensures that productive workflows and tasks remain executable in the same manner with the original object type.

You have to extend an object type if you want to add more events to it, for example.

Process Flow
You cannot change the object types supplied by SAP. If you want to use one of these object types, but would like to make specific extensions or adaptations in certain areas, you proceed as follows:

You create a new object type as a subtype [Extern] of the object type that you want to extend. For further information, refer to Creating Subtypes [Seite 1158].

The new object type inherits the components and their implementation from the supertype [Extern].

You modify and add to this object type. You can redefine existing attributes and methods and create new object type components. For further information, refer to Maintenance of an Object Type Component [Seite 1107].

You make this object type into a delegation type [Extern] of the supertype. For further information, refer to Defining Delegation Types [Seite 1159].
Creating a Subtype

Prerequisites
There is an object type in the system, which you want to extend without modifying it directly.

Creating a subtype and then editing it is the only way to extend an object type supplied by SAP.

Procedure

Select Create.

The Create Object Type dialog box appears.

Maintain the fields Object type, Object name, Name and Description. When doing so, bear in mind the general naming conventions for object types [Extern]. In the field Supertype, enter the ID of the object type for which you want to create a subtype.

Specify a new program, which will include the implementation of the object type and assign the code letter of your application (usually Y or Z).

Select.

The takes you to the screen Change Object Type <object type>: The new object type automatically inherits all the key fields, methods, attributes and events defined for the supertype.

Add your own events, methods or attributes. These extensions must always begin with Y or Z. Edit the existing object type components [Extern] if applicable.

The object type has to be generated after editing.
Defining Delegation Types

Prerequisites
You are on the initial screen of the Business Object Builder. You have created a subtype [Extern] for the object type for which you want to carry out delegation.

Procedure
You can define the delegation on a frontend-specific basis.

The settings you make for the person responsible and delegation are displayed in the basic data of the object type.

Defining a customer-specific delegation type across the system
Choose Settings  Delegation → System-wide.
You are on the screen Display View "Customizing Object Types": Details.
Select 🎨.
Select New entries.
The screen New Entries: Details of Added Entries is displayed.
Enter the ID of the object type for which you want to define a delegation type [Extern]. in the field Object type.
Enter the name of a person responsible.
Enter the ID of the subtype for the object type in the field Delegation type.

This entry now applies system-wide and is cross-client.
If the delegation is only to apply to a particular frontend, or a system-wide delegation is to be overridden with another delegation for a particular frontend, set the indicator GUI-specific.

Defining a customer-specific delegation type for a specific frontend
On the screen Business Object Builder: Initial Screen, enter the ID of an object type as a frontend-specific delegation type in the field Object/interface type.

You can only specify a frontend-specific delegation for an object type if you checked "GUI-specific" beforehand for this object type when maintaining system-wide delegation.

Choose Settings  Delegation → Frontend.
Object Type SELFITEM

Use
The object type SELFITEM provides methods that operate on a work item that represents its own task. When you execute the work item, the object on which the method operates is this work item itself.

Definition

<table>
<thead>
<tr>
<th>Task</th>
<th>Work item 123456</th>
</tr>
</thead>
<tbody>
<tr>
<td>refers to SELFITEM.Note_Create</td>
<td></td>
</tr>
</tbody>
</table>

Runtime

- Execute work item
- Create attachment
- and add to work item 123456

Features

Methods of the object type SELFITEM

<table>
<thead>
<tr>
<th>Method</th>
<th>Explanation</th>
</tr>
</thead>
<tbody>
<tr>
<td>SendTaskDescription</td>
<td>Sending a description of the task represented by the work item as a mail. For further information about the parameters, refer to Method SendTaskDescription [Seite 1162].</td>
</tr>
<tr>
<td>Note_Create</td>
<td>A document is added to the work item as an attachment. This document is either created for the attachment or is imported from the PC or an R/3 folder. The attachment is available in the workflow.</td>
</tr>
<tr>
<td>Note_Display</td>
<td>Displaying a workflow attachment.</td>
</tr>
<tr>
<td>Note_Change</td>
<td>Changing a workflow attachment.</td>
</tr>
<tr>
<td>PrintTaskDescrBatch</td>
<td>Printing a task description in the background. The printer and the print attributes are passed to this method as parameters.</td>
</tr>
<tr>
<td>PrintTaskDescrDialog</td>
<td>Printing a task description in dialog. The information required for printing is requested in dialog.</td>
</tr>
<tr>
<td>DisplayTaskDescrtn</td>
<td>The task description is displayed.</td>
</tr>
<tr>
<td>Note_Append</td>
<td>Attaching an attachment to a work item. The attachments must be objects of the type SOFM.</td>
</tr>
<tr>
<td>Notes_Append</td>
<td>Attaching several attachments.</td>
</tr>
</tbody>
</table>
Method SendTaskDescription

Use
The method SendTaskDescription sends the task description created in the task definition as a mail.

Features

Method parameters of method SendTaskDescription

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Reference</th>
<th>Explanation</th>
</tr>
</thead>
<tbody>
<tr>
<td>AddressStrings</td>
<td>Field reference SOXNA-FULLNAME</td>
<td>Multiline element for containing the recipient names. The recipient names are stored as a character string. The structure of a recipient name depends on the recipient type. If this structure is not in accordance with the recipient type specified in the parameter TypeID, the text is not sent.</td>
</tr>
<tr>
<td>TypeID</td>
<td>Field reference SOS04-L_ART</td>
<td>Indicator for specifying a recipient type. For further information, refer to Recipient Type [Extern].</td>
</tr>
<tr>
<td>Receivers</td>
<td>Object reference ADDRESS</td>
<td>Multiline element for containing the recipient addresses. The recipient addresses are object references to objects of the type ADDRESS. These object references are established in separate steps (Create address). (Generally other objects can also serve as recipients of texts. This is always possible if the object type in question provides a Receive method.)</td>
</tr>
<tr>
<td>Express</td>
<td>Field reference SOS04-L_SEX</td>
<td>Indicator for sending as express mail.</td>
</tr>
<tr>
<td>Language</td>
<td>Field reference SYST-LANGU</td>
<td>Indicator for the language in which the text is to be sent.</td>
</tr>
<tr>
<td>Attachments</td>
<td>Object reference SOFM</td>
<td></td>
</tr>
</tbody>
</table>

It is possible for some of the recipients to be specified by their names (method parameter AddressStrings) and some by their addresses as address objects (method parameter Receivers).
Object Type WF_TASK

Use

The object type WF_TASK provides methods for finding and assigning agents for steps in the workflow. An object of the type WF_TASK is a task specified with type (two characters) and number (eight digits).

Features

Method AllAgentsOfTaskGetAndDispatch

The method AllAgentsOfTaskGetAndDispatch (Determine all agents and select) enables you to select recipients for the work item of a step. The task referenced in the step is an import parameter of the method.

The method determines a list of all possible agents [Extern] for the task, if it is not a general task [Extern]. The agent of the method can make their choice from this list. This selection list is an export parameter of the method.

The method requests a recipient list from the agent of the method for a general task. This list can be made up of user names or Office names. Distribution lists can also be entered. Since the system only searches for distribution lists if it was not able to find a name, you must ensure that you only use distribution lists whose names are not also used as Office names.

Parameters of method AllAgentsOfTaskGetAndDispatch

The parameters of the method AllAgentsOfTaskGetAndDispatch are listed and explained in the table below.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Type</th>
<th>Explanation</th>
</tr>
</thead>
<tbody>
<tr>
<td>Weg_ID</td>
<td>Import</td>
<td>Evaluation path</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Positions and users should be the result of the evaluation.</td>
</tr>
<tr>
<td>Act_Org_Obj</td>
<td>Import</td>
<td>Starting point of the evaluation</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Any organizational object can be specified here.</td>
</tr>
<tr>
<td>Task_ID</td>
<td>Import</td>
<td>Task ID</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Two-character task type and eight-digit number. The possible agents of this task are used in a comparison.</td>
</tr>
<tr>
<td>SelectMultiple</td>
<td>Import</td>
<td>Indicator: Multiple selection allowed</td>
</tr>
<tr>
<td></td>
<td></td>
<td>If the indicator is set, the result of the selection is returned in the multiline element SelectedObjectList. Otherwise the result is returned in the single-line element SelectedObject.</td>
</tr>
<tr>
<td>SelectedObject</td>
<td>Export</td>
<td>Selected agent</td>
</tr>
<tr>
<td></td>
<td></td>
<td>The agent is identified by the two-character abbreviation and eight-digit number or twelve-character user name.</td>
</tr>
<tr>
<td>SelectedObjectList</td>
<td>Export</td>
<td>List of selected agents</td>
</tr>
<tr>
<td></td>
<td></td>
<td>The agents in the list are identified by the two-character abbreviation and eight-digit number or twelve-character user name.</td>
</tr>
</tbody>
</table>

When the method is used in a task, container elements with the same names as the method parameters are created in the task container and the relevant binding is defined.
from/to the task container as standard.

If it is not necessary to compare the agents determined using the evaluation path with the possible agents of the single-step task, the evaluation path can be set so that only the possible agents are determined.

Use the same task as in Task_ID as Act_Org_Obj.

Use the evaluation path SAP_TAGT as Weg_ID.

**Method Dispatch**
The method Dispatch (Display and select agents) enables agents to be selected from a list. The list must be passed to the method as an import parameter.

**Process flow when executing the method Dispatch**

The method receives a list of agents as import parameter. These agents can be organizational units, jobs, positions and users.

This list is compared with the possible agents [Extern] of the task for which the agents are to be established.

The list then contains only the agents who are both possible agents and included in the result of the evaluation path.

The comparison only works if the evaluation path only returns positions and users.

This (reduced) list is offered to the agent of the method so that they can make a choice of agents.

The result of the evaluation is made available as export parameter of the method.

**Parameters of method Dispatch**
The parameters of the method Dispatch (Display and select agents) are listed and explained in the table below.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ObjectList</td>
<td>Import</td>
<td>List of agents as input for reconciliation</td>
</tr>
<tr>
<td>Task_ID</td>
<td>Import</td>
<td>Task ID Two-character task type and eight-digit number.</td>
</tr>
<tr>
<td>SelectMultiple</td>
<td>Import</td>
<td>Indicator: Multiple selection allowed If the indicator is set, the result</td>
</tr>
<tr>
<td>SelectedObject</td>
<td>Export</td>
<td>Recipient The agent is identified by the two-character abbreviation and</td>
</tr>
<tr>
<td>SelectedObjectList</td>
<td>Export</td>
<td>List of recipients The agents in the list are identified by the two-character</td>
</tr>
</tbody>
</table>

April 2001
The method Dispatch is called internally by the method GetListAndDispatch.

**Method Resolve**
The method Resolve resolves a list of organizational objects consisting of two-character type and eight-digit number by evaluating the organizational plan up to user level. The resolved list is returned sorted alphabetically, with any duplicates being omitted.

**Parameters of method Resolve**

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ObjectList</td>
<td>Import</td>
<td>List of organizational objects</td>
</tr>
<tr>
<td></td>
<td></td>
<td>The objects in the list are identified by the two-character abbreviations and eight-digit number or twelve-character user name.</td>
</tr>
<tr>
<td>ResolvedObjectList</td>
<td>Export</td>
<td>List of user names</td>
</tr>
<tr>
<td></td>
<td></td>
<td>The objects in the list are identified by the two-character abbreviation &quot;US&quot; and the twelve-character user name.</td>
</tr>
</tbody>
</table>

**Method ResolveAndDispatch**
The method ResolveAndDispatch connects the methods Resolve and Dispatch and processes the following steps:
A list of organizational objects consisting of two-character type and eight-digit number is resolved up to user level by evaluating an organizational plan.
This list is passed to the selection dialog.
The selection in dialog constitutes the return value of the method.
It is not the organizational objects stored in the task definition that are offered for selection in the selection dialog, but the users connected with these organizational objects.
You should use the virtual attribute AllAgentsOfTask to read the task definition and use it as an import parameter for the method.

**Parameters of method ResolveAndDispatch**

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ObjectList</td>
<td>Import</td>
<td>List of organizational objects that are offered for selection, resolved to user level.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>The objects in the list are identified by the two-character abbreviations and eight-digit number or twelve-character user name.</td>
</tr>
<tr>
<td>Title</td>
<td>Import</td>
<td>Title of the dialog box for agent selection.</td>
</tr>
<tr>
<td>SelectMultiple</td>
<td>Import</td>
<td>Indicator: Multiple selection allowed</td>
</tr>
<tr>
<td></td>
<td></td>
<td>If the indicator is set, the result of the selection is returned in the multiline container element ResolvedSelectedObjectList. Otherwise the result is returned in the single-line element ResolvedSelectedObject.</td>
</tr>
<tr>
<td>ResolvedSelectedObject</td>
<td>Export</td>
<td>Selected agent</td>
</tr>
<tr>
<td></td>
<td></td>
<td>The agent is identified by the two-character abbreviation &quot;US&quot; and twelve-character user name (cf. import parameter &quot;SelectMultiple&quot;).</td>
</tr>
<tr>
<td>ResolvedSelectedObjectList</td>
<td>Export</td>
<td>Selected agents</td>
</tr>
<tr>
<td></td>
<td></td>
<td>The agents in the list are identified by the two-character abbreviation &quot;US&quot; and twelve-character user name (cf. import parameter &quot;SelectMultiple&quot;).</td>
</tr>
</tbody>
</table>
Method AllAgtsofTskGetDispatchResolve

The method AllAgtsofTskGetDispatchResolve executes the methods AllAgentsOfTaskGetAndDispatch and Resolve in succession, meaning that the recipients returned are all user names. It is possible to restrict the organizational objects offered for selection and deactivate multiple selection within this list. However, a multiline container element is always returned irrespective of the status of multiple selection.

Parameters of method AllAgtsofTskGetDispatchResolve

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>SelectMultiple</td>
<td>Import</td>
<td>Indicator: Multiple selection allowed</td>
</tr>
<tr>
<td>Title</td>
<td>Import</td>
<td>Title of the dialog box for agent selection.</td>
</tr>
<tr>
<td>AllowedAgentTypes</td>
<td>Import</td>
<td>List of the Organizational Management types that can be selected at all, for example &quot;O&quot; for &quot;organizational unit&quot;.</td>
</tr>
<tr>
<td>SelectedObjectList</td>
<td>Export</td>
<td>List of selected recipients</td>
</tr>
</tbody>
</table>

Attribute AgentsOfTask

The (multiline) virtual attribute provides all possible agents of the task by reading the task definition.
Business Object Repository

Use
The Business Object Repository [Extern] contains the definition of the object type components and their implementation (hence all information about the object types). Only the object types defined in the Business Object Repository can be addressed in the definition and runtime components of SAP Business Workflow.

Integration
You access the Business Object Repository and the object types therein via the Business Object Repository Browser.

Features
The Business Object Repository contains the following classes of object type:
- Business object types reflected in the data model
- Organizational types such as company code, plant or sales organization
- Technical object types such as text, note, work item or archived document
You can define your own object types taking into account the naming conventions [Extern].

Assignment between object type, development class and component
Each object type is assigned to an application component indirectly via its development class. Object types stored as local objects are not assigned to any application component.

Relationships between object types
Regardless of its assignment to development class (and the resulting assignment to an application component), an object type can have various relationships to other object types:

| Inheritance | Inherits from | Relationship between object types allowing common attributes and methods to be passed automatically from supertypes to subtypes. The object type from which attributes and methods are inherited is the supertype [Extern]. The subtype [Extern] inherits the attributes [Extern] and methods [Extern] from the supertype. The supertype's implementation program [Extern] associated with the attributes and methods is referenced. Inherited attributes and methods can be redefined for the subtype. The interface, however, may only be extended. If the attributes and methods of the supertype are known, it is only necessary to describe the differences (changes and extensions) in the subtype in question. All other attributes and methods defined once are used again which reduces the implementation work involved. The error probability when changes are made later also decreases. In this inheritance the subtype has the same key fields as its supertype, but extended functionality. |
|Composition | Is contained in | "Is part of" relationship between object types. The object type OrderItem "is part of" the object type Order. The object type Order is then the aggregate type of the object type OrderItem. With composition the "is part" object type normally has a key that is an extended version of that of the aggregate type [Extern] (for example the order key plus the item) and completely different functionality. For further information about maintaining this relationship, refer to Interface Aggregate [Seite Error! Bookmark not defined]. |

April 2001
### Business Object Repository

<table>
<thead>
<tr>
<th>Relationship</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Contains</td>
<td>Relationship expressing which object types support an interface. To maintain this relationship, the relevant interfaces must be entered as components of the object type.</td>
</tr>
<tr>
<td>Implements</td>
<td></td>
</tr>
<tr>
<td>Is implemented by</td>
<td></td>
</tr>
<tr>
<td>Association</td>
<td>Relationship between object types resulting from an attribute referencing another object type as an object reference.</td>
</tr>
<tr>
<td>Is referenced</td>
<td></td>
</tr>
<tr>
<td>References</td>
<td></td>
</tr>
</tbody>
</table>

The object type Customer is referenced in the attribute OrderingParty of the object type SalesOrder.
Business Object Repository Browser

Use
Using this function, you can access the Business Object Repository and navigate within it.
You call this function:
If you want an overview of the existing object types.
If you want to maintain individual object types from the overview.

You cannot create new object types in the Business Object Repository Browser.
You can maintain object types you know directly from the Business Object Builder.

Features
When you call the Business Object Repository Browser, the system takes account of the filter criteria set.

Filter criteria when calling
You define the filter criteria by selecting 🗠. The filter criteria are on the following tab pages:
Filter object types
You use these filters to select the object types displayed according to class, transport property and release status.
Filter relationships
You use these filters to select the relationships displayed between the object types.
Other filters
You use these filters to influence whether and how the names of the object types and application components are displayed.
You can save the filter criteria set for the next time the Business Object Repository Browser is called.
To do this, choose Settings → Save filter.

Displaying the object types
The display of the object types in the Business Object Repository Browser depends on the filter criteria with which you called the Business Object Repository Browser.

If you choose to have the assignment to development class (application component) as well as the relationships to other object types displayed for the object types, only the object types assigned directly to the component are displayed in the component assignment.

Searching in the Business Object Repository Browser
You can search for object types in the hierarchy of the Business Object Repository.
If an “appropriate” object type is found and you have selected in the whole hierarchy as a search criterion, the hierarchy is expanded at the relevant point.

Activities
You call the Business Object Repository Browser from the Business Object Builder. Choose Business Object Repository and select the number of object types to be displayed in the dialog box Business Object Repository Browser: Entry Screen. Select ☑ Other settings to specify individual filter criteria.
Workflow Configuration

Use
To adapt an SAP Workflow, you can create a configuration. In this configuration you can redefine values for every step of the workflow definition. These values are evaluated at runtime instead of the values originally defined.

Features
You can set the following data individually in each step definition:
- Responsible agents [Extern]
- Excluded agents [Extern]
- Message recipient for completion [Extern]
- Priority [Extern]
- Requested start [Extern]
- Indicator denoting whether the step is included into the workflow log
- Activation of a latest end, a latest start or a requested end with the reaction Send mail
- Reference date/time for latest end, latest start and requested end
- Message recipient for missed deadline [Extern]

Information about the work item display

Activities
You start workflow configuration by choosing Tools → Business Workflow → Development → Definition Tools → Workflow Configuration. This displays the screen Process Workflow Configuration for 'workflow name'. Using a workflow configuration is like using the Workflow Builder [Seite Error! Bookmark not defined.]. Double-clicking on a step in the workflow definition displays the step definition, where you can enter values. These values are then used at runtime instead of the values defined in the workflow definition.

Once you have saved a workflow configuration for a workflow, it is active. To delete the configuration, choose Configuration → Delete.

If a workflow configuration exists for a workflow definition, this is apparent in the Workflow Builder when you check the workflow. The workflow configuration belongs to the workflow version for which it was created. If a new version is created, the new version also has this configuration.

A workflow configuration is client-specific. For further information on workflow configuration transport, refer to Transport and Client Copy of Tasks [Seite 1582].
Tasks and Task Groups

Use
Within SAP Business Workflow you use tasks for the following purposes:
To use methods of an object type in a workflow (single-step task [Extern])
As a framework for defining a workflow (multistep task [Extern])
You use a task group [Extern] to collect tasks used in a common context.

Prerequisites
To create a single-step task, you must know the object type [Extern] and its method [Extern] to be executed with the task.
To create a multistep task, you must be familiar with the business process you wish to map as a workflow definition, its flow and its individual steps.

Features
You can process tasks of the following types.
Standard task (single-step task, task type TS)
Customer task (single-step task, task type T)
Workflow template (multistep task, task type WS)
Workflow task (multistep task, task type WS)
Task group (task type TG)

Only cross-client single-step tasks (= standard tasks) and cross-client multistep tasks (= workflow templates) can be created now.
When a workflow task/customer task is copied, the task type is changed automatically to workflow template/standard task.
For further information, refer to Creating and Maintaining Tasks [Seite 1173]. Also refer to the relevant task type:
Definition of a Single-Step Task [Seite 1175]
Definition of a Multistep Task [Seite 1194]
Definition of Task Groups [Seite 1208]
Creating and Maintaining Tasks

Creating tasks

Choose Tools → Business Workflow → Development → Definition Tools → Tasks/Task Groups → Create.

Select the relevant task type.

Select .

Changing tasks


Identify the task that you want to process by:

Selecting the type of task and entering the 8-digit task number

Entering the task abbreviation in the field Task (using the F4 input help if applicable)

Select .

In a customer system you should not change workflows and tasks supplied by SAP directly. If the only changes you want to make to the task are to triggering events, terminating events or texts, you can extend the definition. To make this extension, open the definition in display mode and go to the relevant tab pages. If you want to make more extensive changes, copy the task or workflow.

Displaying tasks


Identify the task that you want to process by:

Selecting the type of task and entering the 8-digit task number

Entering the task abbreviation in the field Task (using the F4 input help if applicable)

Select .

If you are not in the original system of the task, you can create additional triggering events and terminating events, as well as change the work item text and the texts on the tab page Description.

Copying tasks


Identify the task that you want to process by:

Selecting the type of task and entering the 8-digit task number

Entering the task abbreviation in the field Task (using the F4 input help if applicable)

Select .

Enter a new abbreviation and a new name for the copy.
The abbreviation and name of the copied task only exist in the logon language of the user who copied the task. The abbreviation and name of the original task, which may already exist in other languages, are not copied. However, translations of the work item text and other texts are retained in the copying process.

Deleting tasks


Identify the task that you want to process by:

- Selecting the type of task and entering the 8-digit task number
- Entering the task abbreviation in the field Task (using the F4 input help if applicable)

Select [ ].

The system checks whether there are still work items [Extern] for the task to be deleted. If this is the case, you cannot delete the task. To delete the task anyway, you must remove the work items from the system. For further information, refer to Archiving Work Items [Extern].

Result

If you have deleted or copied a task, you remain on the same screen. Otherwise you go to the screen for displaying or processing the selected task. For further information, refer to the relevant task type:

- Definition of a Single-Step Task
- Definition of a Multistep Task
- Definition of Task Groups
Definition of a Single-Step Task

Use
You use a single-step task to define an activity that can be executed within a workflow definition or independently. This activity can be a transaction in the R/3 System.

⚠️ You have to create new single-step tasks as standard tasks. Customer tasks can only be edited now. There will be no functional enhancements for customer tasks.

Single-step task as a single step
It is not always necessary for a complex workflow definition involving several steps to be executed to process a business task. Often a single step in the form of a standard task can perform the function required.

Single-step task as part of a workflow definition
A single-step task can be part of a workflow definition as a step of the type activity. In the workflow definition, additional information can also be specified regarding responsibilities and deadlines. If dialog with a user is required to execute the single-step task, all possible agents receive a dialog work item in their Business Workplaces. If one of the recipients executes the work item, the relevant activity is executed.

Integration

Single-step task activity
You define the activity of the single-step task using an object type method of one of the object types defined in the Business Object Builder.

Possible agents of single-step tasks
The possible agents of a single-step task are specified in the relevant definition. These possible agents can be defined with reference to the organizational plan of the enterprise, with the consequence that the same task is offered for execution to several users, who have the same authorizations from an organizational perspective, in their Business Workplaces. If the task is reserved by a recipient, it is removed from the Business Workplaces of the other recipients. If the task is used within a workflow, you can define responsible agents in the workflow definition. The recipients of the work item are all users who are both responsible agents and possible agents.

Prerequisites
The object type method and the object type are defined, tested and generated. You do not have to have defined the object type and its methods yourself. You can also use methods supplied as standard.

Features
The definition of single-step tasks is spread across several tab pages. You can make all the mandatory specifications on the tab page Basic data. The specifications on the other tab pages are optional.

Customer tasks are processed on a screen without tab pages. But the activities to be carried out are the same.

Activities
You should adhere to the following sequence when defining single-step tasks (and analyzing existing single-step tasks):
## Definition of a Single-Step Task

<table>
<thead>
<tr>
<th>What?</th>
<th>Where?</th>
</tr>
</thead>
<tbody>
<tr>
<td>Maintain names, work item text, object type and method</td>
<td>Tab page Basic data</td>
</tr>
<tr>
<td>Maintain agent assignment</td>
<td>Additional data Agent assignment Maintain</td>
</tr>
<tr>
<td>Maintain elements in task container</td>
<td>Container</td>
</tr>
<tr>
<td>Maintain triggering events</td>
<td>Tab page Triggering events</td>
</tr>
<tr>
<td>Maintain terminating events</td>
<td>Tab page Terminating events</td>
</tr>
<tr>
<td>Maintain default roles</td>
<td>Tab page Default roles</td>
</tr>
<tr>
<td>Maintain description and notification texts</td>
<td>Tab page Description</td>
</tr>
<tr>
<td>Maintain additional data</td>
<td>Additional data Selection criteria...</td>
</tr>
<tr>
<td>Maintain SAPphone properties</td>
<td>Tab page SAPphone</td>
</tr>
</tbody>
</table>
Maintenance of Basic Data of Single-Step Task

Use
The basic data is used to identify a single-step task. You also specify the activity that is to be executed with the task in the basic data.

Prerequisites
You must know the object type and its method.

Features
Basic data maintenance is carried out on the tab page Basic data and includes the following functions:
- Processing names and work item text
- Maintaining the reference to an object method
- Maintaining the execution properties

Identification of a single-step task
Each single-step task is identified by the object identifier T (for customer task) or TS (for standard task) and an 8-digit task number allocated automatically by the system during creation. (The settings required for the number assignment are made in SAP Business Workflow Customizing [Extern]).

Assignment to application component
If you do not save the single-step task as a local object, you must specify a development class when saving. The system completes the fields Development class and Application component from this.

Transport
As cross-client transport objects (transport object PDTS), standard tasks are always connected to the transport system. Customer tasks are only connected to the transport system (transport object PDT) in a particular client if Automatic recording of changes is set for this client in table T000. In other clients, you can include customer tasks in a transport request manually. To do this, choose Task Transport.
Maintenance of Reference to Object Method

Use
You define the activity to be executed with a single-step task by specifying an object type method.

Activities
You determine the reference to the object method on the tab page Basic data by specifying object type [Extern] and method [Extern]. The object type must be activated and defined with the method in the Business Object Builder. The system takes the following from the definition of the object method, as applicable:

- Synchronous object method
- Object method with dialog

You cannot change these. If method parameters are defined for the object method, the system gives you the option of creating relevant container elements automatically in the task container. The names of these container elements are then identical in the task container and the method container. You should make use of this option. For further information, refer to Binding Definition from Method Container and Binding Definitions from Task Container.

If the task is executed as a form task [Extern] for SAP forms, you can specify a method as an alternative. You enter this method on the tab page Alternative methods in the line Form task. The alternative method must have the same interface as the method entered on the tab page Basic data.

The alternative method ProcessExternally should be entered for form tasks that use the method Process. The alternative method is used if the execution is not started from the Business Workplace in SAP GUI for Windows. The alternative method entry is deleted automatically if you enter another method on the tab page Basic data.

The specifications in the lines WEB METHOD and MASS PROCESSING (IN R/3) are not currently evaluated.

Navigation
By double-clicking on the object type ID or method ID entered, you can go to the relevant object type in the Business Object Builder and process it.

Constraints
For technical reasons, the workflow system can only process method parameters created with a data type reference [Extern] to a structure or a table under the following circumstances:

- The total length of all fields of the table or the structure does not exceed 255 characters.
- Only alphanumeric data types are used in the fields of the table or structure.

If a method is used in a single-step task with parameters, the system checks whether the method parameters meet the requirements formulated above in the single-step task definition. If this is not the case, you cannot define the single-step task.
Maintenance of Execution Properties

Use
The execution properties determine additional processing characteristics of the single-step task or the work item that represents the single-step task at runtime.

Features
The following execution properties are recognized by the system:

Background processing
Set this indicator if the object method does not necessitate dialog with a user, and the processing is to be started automatically by the system.

If this indicator is set, the single-step task is executed at runtime as a background work item as soon as it is its turn in the sequence. No work item appears in the Business Workplace. You can, however, find a background work item via the work item selection.

If this indicator is not set, a dialog work item appears in the Business Workplaces of the recipients. If this work item is executed by one of its recipients, the object method is executed in the background without any further dialog.

You cannot set this indicator if the indicator Object method with dialog is set for this object method.

Confirm end of processing
If this indicator is set, the end of processing of the single-step task must be confirmed explicitly at runtime. After execution of a single-step task, the system asks via a dialog box whether the processing of the task has actually been completed. It is also possible, for example, to enter an attachment at this stage.

Until this confirmation is made, the work item in question can be processed and forwarded. This may be useful if a single-step task is to be processed several times and by different people.

The property Confirm end of processing cannot be assigned to single-step tasks that are to be executed in the background (see above).

The property Confirm end of processing can be assigned both to single-steps task that do have terminating events and to those that do not. If there are terminating events, the relevant work item must be completed using the function Set to 'Done' in the Business Workplace.

Executable as form
If this indicator is set, the task meets the technical requirements for execution as an electronic form.

The system cannot check whether this property has been assigned correctly.

Form tasks always have this property. If you define a form task, you must set this indicator.
Editing Names

Use
A task is identified by its abbreviation. It can also be used in input fields as a search term. The name is output in logs to describe the task.
The work item that represents the task is displayed with the work item text in the Business Workplace, in the work item selection hit list and in the workflow log. If you do not specify a work item text, the name of the task is used instead.
The release status is of an informal nature and indicates the development status of the task.

Procedure
Edit the abbreviation.
Edit the name.
Enter the work item text.

If you want to use expressions [Extern] in the work item text, position the cursor at the relevant point in the text and select the button.

The expression is replaced at execution time with the current values from the relevant container [Extern]. Note that container elements used in the work item text must be filled via a binding.

If you are looking at a task in display mode, whose original system is not the current system, you can redefine the work item text. Select the button and enter a text that is to replace the SAP standard text. This creates an extension of the task definition.

Select an appropriate release status for the task.
Maintenance of Agent Assignment

Use
This function assigns possible agents to a task.

No possible agents are assigned initially to an SAP task or an SAP workflow. You can perform this assignment in Customizing. You can change and extend these assignments at a later date as well.

Integration

Establishing recipients at runtime
The system establishes the recipients of a work item at runtime. The recipients are derived from the intersection of possible agents and responsible agents barring excluded agents.

You specify the responsible agents for a step in the step definition. You can specify not only a system user, but also an object of Organizational Management such as a position, a job, an organizational unit or a work center. Alternatively, you can define a container element that contains the object of Organizational Management at runtime or a role for dynamic agent determination.

You can define agents excluded from processing within the step definition.

The user who actually executes the work item at runtime is the actual agent.

The possible agents for a task must be maintained so that the recipients can be established at runtime.

Using a default role
You can specify a default role when defining a single-step task so that the agents are established dynamically when this single-step task is executed.

The default role is evaluated if the single-step task is executed alone. The default role takes on the function of the responsible agents in this case.

The default role is also evaluated if the single-step task is used in a workflow definition and no responsible agents are specified in the step definition in question.

Features

General

Task
The possible agents of a task are all of the users who are organizationally responsible and authorized to execute the task at runtime. Only they can receive a work item for this task in their Business Workplaces. Starting the task in dialog outside a workflow is included in execution.

You must assign possible agents to a task for the task to be started and executed at runtime.

You can create several assignments for a task and also use various types of agent doing so.

The task “Check invoice” is linked with the following possible agents.

Organizational unit Purchasing
Organizational unit Internal auditing
Job Accountant
You do not have to assign possible agents to tasks designated as background tasks. You must assign possible agents to those background tasks that are also to be started as single steps in dialog.

**Multistep task**
The possible agents of a multistep task are those users who are authorized to start the workflow connected with the task manually.
You only have to assign possible agents to a multistep task if the workflow is to be started directly by users. If the workflow is only to be started by events, do not assign any possible agents to the multistep task.

**Task groups**
If you assign possible agents to a task group, this assignment is passed on to all tasks contained in the task group.
Check whether the assignment defined here really is to be apply to all task groups, single-step tasks and multistep tasks contained.

**Basic properties of tasks**

**Task**
Choose the basic property of the task from the following alternatives:

**General task**
If you define a task as a general task [Extern], all users can execute the task. This is useful if the task is used in a workflow and you only want to define the recipients in the step definition. A recipient can forward associated work items to all users.

Work items whose tasks are defined as general tasks and for which no responsible agents or default roles are defined are offered to all users of the R/3 System for execution in their Business Workplaces.

**General forwarding allowed**
A work item that represents a task with this property can be forwarded by one of its recipients to all users, even if they are not possible agents of the task.

**General forwarding not allowed**
A work item that represents a task with this property can be forwarded by one of its recipients only to the possible agents of the task.

**Multistep task**
With a multistep task you can only decide whether it is a general task or not.

**Activities**
To process the basic properties of a task, choose Additional data → Classification → Change in task maintenance.
To assign a task you have defined to its possible agents or process its basic properties, choose Additional data - Agent assignment - Maintain and select the task.
Select Attributes... to define the basic properties of the task.
Choose Agent assignment → Create... to assign the possible agents.

You can process the possible agents of the task and workflow (multistep task) from the Workflow Builder as well.
Maintaining Task Texts

Prerequisites
You enter the following information on the tab page Description:

Task description
You can describe the task or make notes and recommendations for processing. The task description is displayed in the work item display.

Notification texts
These texts are sent as notification to the message recipients for completion [Extern] or missed deadlines [Extern].

Procedure
Switch to the tab page Description.
Select the text type that you want to create or change.
Select 📝.

If you are looking at a task in display mode, whose original system is not the current system, you can change the existing texts. This creates an extension of the task definition.

Compose your text.
To add text variables to your text, choose Include Expression. You can add as many text variables to the text as you want. For further information, refer to Text Variables and Replacement [Extern] and Text Processing with the SAPscript Editor [Extern].

You want to display the number of the accounting document currently being processed in the task description for a standard task. The document number is defined under the name DocumentNo as a key field for the object type BUS2068 (accounting document).

The reference to the processed object (of type BUS2068 in this example) is located in the task container under the element _WI_Object_ID. To reference the document number in the text, write &BUS2068.DocumentNo& at the appropriate point in the text.
Definition of the Task Container

Use
This function is used to process the container elements of the task container [Extern]. Each task container already contains the workflow system elements [Extern] and any additional elements for the method parameters of the referenced object method.
You have to add additional elements to the task container if:
Additional values are required for variable replacement in the work item text or in the long texts
Values are assigned to the task container at runtime in the following ways:
By initial value assignment .
Via binding from the workflow, event or method container
Initial values are overwritten by bindings. Initial value assignment and processing of the task container are only possible during definition of the single-step task.
Creating Container Elements

Prerequisites
You are in one of the definition tools and want to add a new element to the container.

Procedure
Select Container.

Select .

Enter the technical name of the container element into the field Element.

You give each container element a technical name that can be used to identify it uniquely. The technical name must begin with a letter. This can be followed by letters, underscores or digits. Not case-sensitive. The technical name you choose should be in English.

Under Texts maintain the name and the description.

This text can include upper and lower cases.

Under Properties, select whether the new element is to be an import and/or an export element. Mark the element as mandatory if applicable.

The values for each of the container elements may be in the form of a multiline list of field values or object references. This list is then saved under an element name. You can select multiline in the container definition so that this container element can hold a multiline list of values.

According to the data type reference [Exterm] of the container element, make the following entries. If the new container element:

Is to reference a structure, enter the table or structure in the field Reference table
Is to reference an ABAP Dictionary field, specify a reference field as well
Is to reference an object type, enter the ID of the object type in the field Object type

You can use the F4 input help for all of these.

Specification of a ABAP Dictionary field reference is not mandatory. If no field reference is specified, the data type is C(255) (upper and lower case). Specification of an object type is not mandatory either. If no object type is specified, the container element can be assigned a reference to an object of any type.

Select .
Assigning Initial Values

Use
The initial value assignment of constants to container elements should be regarded as a preassignment of values to container elements.

Procedure
Select Initial values.

All container elements designated as import elements are displayed. The import workflow system elements are not displayed until you select _WI_Object_Id.

You can only assign values to those container elements that are designated as import elements.

Enter the initial value in the field Expression.

To assign several values to a target element defined as multiline, create the additional input lines using 

Result
At runtime the system executes the initial value assignment in accordance with your definition.

If there is a binding definition from another container to the container with the initial values, the system executes this binding at runtime after the initial value assignment. Any initial values are overwritten.
Creating Triggering Events

Procedure
You maintain the triggering events [Extern] on the tab page Triggering events.
Identify the event by specifying its instance ID and object type. The event must be defined for this object type.

If you are looking at a task in display mode, whose original system is not the current system, you can create additional triggering events. This creates an extension of the task definition.

To activate the event receiver linkage, select in the column . Active event linkages are denoted by .

To define the binding from the event container, select the relevant event line and choose . For further information, refer to Binding Definitions from the Event Container .

Select the triggering event and choose . The properties of the event linkage are displayed and can be changed.

Result
The workflow system enters the linkage between the event and the event-receiving task in the relevant linkage table [Extern] automatically. For further information on linkage transport, refer to Transport and Client Copy .

If an error occurs when a task or workflow is being started with an event, the standard procedure is for the system to deactivate the event linkage to avoid any follow-on errors. At the same time, the system sends an e-mail to the workflow system administrator. But you can change this in event queue administration.
Definition of Terminating Events of Task

Use

This function is used to define the terminating events of the task. Tasks that refer to an asynchronous object method must always be defined with at least one terminating event. At runtime, the relevant work item is only terminated if one of the defined terminating events occurs.

Tasks that refer to a synchronous object method can also be defined with terminating events. At runtime, the relevant work item is generally terminated when the synchronous method has been successfully executed. But the work item is also terminated if one of the defined terminating events occurs.

Whether a method is defined as synchronous or asynchronous depends on its implementation in the Business Object Builder.

The terminating events of a task express the possible end statuses of the task. A terminating event always refers to one object for which the event must be triggered. You define the object for which the event must be triggered in a container element in the task container.

The event DELETED of object type BUS2032 is defined as a terminating event for a task. At runtime the sales order 123456 is passed via the task container. The task is now only terminated if the sales order 123456 is deleted and the event BUS2032.DELETED triggered. If another sales order is deleted, the task is not terminated.

Integration

If the task is used as a step in a workflow definition, you can go to a separate modeling branch when the event occurs. For further information, refer to Maintenance of Tab Page Outcomes.

The task Release budget has the terminating event Budget cancelled.

If a task is terminated by an event, the execution of the workflow is continued. The method does not necessarily have to have been executed. If a task is aborted or non-defined processing statuses arise, the work item is not terminated since no event is created. The workflow is not continued.

Activities

You maintain the terminating events on the tab page Terminating events.

You identify the event by specifying its instance ID and object type. The event must be defined for this object type.

If you are looking at a task in display mode, whose original system is not the current system, you can create additional terminating events. This creates an extension of the task definition.

You specify a container element of the task container, which at runtime contains a reference to the object for which the event has occurred. This is generally the task container element _WI_Object_ID.

The workflow system enters the instance linkage required into the linkage table automatically after the task is started.

If you want to check the properties of the terminating event, select . The properties of the instance linkage are displayed and can be changed.
If you change the properties of terminating events, all terminating events that exist for this task in the workflow system are affected by the change. Therefore do not change the indicator for activation of the event.
Roles in Single-Step Tasks

Use
You use roles [Extern] to define the responsible agents [Extern] of the single-step task and the recipients of notifications.

Features
To determine the responsible agents and recipients of notifications, you can specify roles. Depending on the use of the task, these roles are evaluated as follows:

Task is started in dialog or via an event
The system resolves the role Agents at runtime and determines the responsible agents of the task. All responsible agents who are also possible agents [Extern] become recipients [Extern] of the relevant work item.

Task is started from a workflow
The roles entered are only resolved if nothing is specified in the step definition.

Activities
You make the presettings on the tab page Default roles.

Binding
You must define the binding from the task container to the role container for each role. For further information, refer to Role Resolution.
Maintenance of Classification and Lock Indicators

Use

Classification
You can arrange single-step tasks into various task classes within the SAP R/3 System. The task classes professional, personal or disciplinary are standard. The system checks the task class under the field name TASK_CLASS in the authorization object S_WF_WI. This system uses this authorization object to check the authorization for carrying out specific actions on particular work items. The assignment of single-step tasks to classes is also important when defining substitutes. You can specify that only the single-step tasks of a particular class are displayed to your substitute. Take this into account when you create new classes and assign single-step tasks to the classes.

Lock indicators
You can set the following lock indicators for single-step tasks:
Locked against instantiation
   The system does not create a work item for this single-step task.
Locked against execution
   The system does create a work item for this single-step task, but it cannot be executed.

Integration

Adding new task classes in the system
In Customizing for Organizational Management, you can add more task classes to the predefined classes.

Activities
The maintenance of task classes and lock indicators is carried out under Additional data Classification.
Maintenance of SAPphone Properties

Use
The SAPphone properties determine the possible uses of a task in the SAPphone environment.

Integration
SAPphone integrates telephony functions into the R/3 System. R/3 applications use this to integrate the telephone into the work process. SAPphone provides the Basis technology required. The processing of outgoing and incoming telephone calls is supported with data from the R/3 System environment (for example, telephone number or name). An incoming call starts a search for caller data. The result of the search is displayed automatically in an information window. For further information, refer to SAPphone [Extern].

Features
If you label the task accordingly, it is displayed and can be executed in the event of incoming or outgoing calls. The task container must contain appropriate container elements to enable the information derived from the telephone data to be used in the task.
Definition of a Multistep Task

Use
A multistep task [Extern] is a formal framework for a workflow definition. When you create a workflow directly in the Workflow Builder, the multistep task is created by the system. Multistep tasks can be divided up into workflow templates [Extern] and workflow tasks [Extern].

Attention
New multistep tasks must be created as workflow templates. Workflow tasks can no longer be created, but can be edited. There will be no functional enhancements for workflow tasks.

Integration
Reference to workflow definition
A multistep task contains a reference to a workflow definition [Extern]. The business activity described with the multistep task therefore usually covers several steps in what may be a complex arrangement. This is a major difference when compared with tasks.

Multistep task as interface description
A multistep task defines the formal framework for a workflow definition with the description of the interface [Extern]. In addition to the reference to the workflow definition, the multistep task contains other information:
Definition of the workflow container (interface description)
Triggering events

Features
The definition of workflow templates is spread across several tab pages.
Basic data
Description
Triggering events
SAPphone
Workflow tasks are defined on a screen without tab pages. But the activities to be carried out are the same.

**Activities**

You should adhere to the following sequence when defining multistep tasks (and analyzing existing multistep tasks):

<table>
<thead>
<tr>
<th>What?</th>
<th>Where?</th>
</tr>
</thead>
<tbody>
<tr>
<td>Maintain abbreviation, name and work item text</td>
<td>Tab page <strong>Basic data</strong></td>
</tr>
<tr>
<td>Define elements in workflow container</td>
<td>![Container] <strong>Container</strong></td>
</tr>
<tr>
<td>Maintain triggering events if required</td>
<td>Tab page <strong>Triggering events</strong></td>
</tr>
<tr>
<td>Maintain description and notification texts</td>
<td>Tab page <strong>Description</strong></td>
</tr>
<tr>
<td>Maintain agent assignment</td>
<td>Additional data <strong>Agent assignment</strong> <strong>Maintain</strong></td>
</tr>
<tr>
<td>Maintain SAPphone properties</td>
<td>Tab page <strong>SAPphone</strong></td>
</tr>
</tbody>
</table>
Maintenance of Basic Data of Multistep Task

Use
The basic data is used to identify a multistep task.

Features
The tab page Basic data is used for processing the names of the multistep task. You can also call the Workflow Builder from there.

Identification of a multistep task
Each multistep task is identified with an 8-digit task number and the organizational object type WS (for workflow template) or WF (for workflow task). The number is assigned automatically by the system when a newly created task is saved.
You define the first three digits of the number in Customizing [Extern].

Transport
As cross-client transport objects (transport object PDWS), workflow templates are always connected to the transport system.
Workflow tasks are only connected to the transport system in a particular client if Automatic recording of changes is set for this client in table T000. In other clients, you can include workflow tasks in a transport request manually. To do this, choose Workflow task Transport.

Activities
If you do not save the multistep task as a local object, you must specify a development class when saving. The system completes the fields Development class and Application component from this.
Editing Names

Use
A task is identified by its abbreviation. It can also be used in input fields as a search term. The name is output in logs to describe the task. The work item that represents the task is displayed with the work item text in the Business Workplace, in the work item selection hit list and in the workflow log. If you do not specify a work item text, the name of the task is used instead. The release status is of an informal nature and indicates the development status of the task.

Procedure
Edit the abbreviation.
Edit the name.
Enter the work item text.
If you want to use expressions [Extern] in the work item text, position the cursor at the relevant point in the text and select ![Expression].

The expression is replaced at execution time with the current values from the relevant container [Extern]. Note that container elements used in the work item text must be filled via a binding.

If you are looking at a task in display mode, whose original system is not the current system, you can redefine the work item text. Select ![Edit] and enter a text that is to replace the SAP standard text. This creates an extension of the task definition.

Select an appropriate release status for the task.
Maintenance of Agent Assignment

Use
This function assigns possible agents to a task.

No possible agents are assigned initially to an SAP task or an SAP workflow. You can perform this assignment in Customizing. You can change and extend these assignments at a later date as well.

Integration

Establishing recipients at runtime
The system establishes the recipients of a work item at runtime. The recipients are derived from the intersection of possible agents and responsible agents barring excluded agents.

You specify the responsible agents for a step in the step definition. You can specify not only a system user, but also an object of Organizational Management such as a position, a job, an organizational unit or a work center. Alternatively, you can define a container element that contains the object of Organizational Management at runtime or a role for dynamic agent determination.

You can define agents excluded from processing within the step definition. The user who actually executes the work item at runtime is the actual agent.

The possible agents for a task must be maintained so that the recipients can be established at runtime.

Using a default role
You can specify a default role when defining a single-step task so that the agents are established dynamically when this single-step task is executed.

The default role is evaluated if the single-step task is executed alone. The default role takes on the function of the responsible agents in this case.

The default role is also evaluated if the single-step task is used in a workflow definition and no responsible agents are specified in the step definition in question.

Features

General

Task
The possible agents of a task are all of the users who are organizationally responsible and authorized to execute the task at runtime. Only they can receive a work item for this task in their Business Workplaces. Starting the task in dialog outside a workflow is included in execution.

You must assign possible agents to a task for the task to be started and executed at runtime.

You can create several assignments for a task and also use various types of agent doing so.

The task “Check invoice” is linked with the following possible agents.

- Organizational unit Purchasing
- Organizational unit Internal auditing
- Job Accountant

You do not have to assign possible agents to tasks designated as background tasks. You must assign possible agents to those background tasks that are also to be started as single steps in dialog.
**Multistep task**

The possible agents of a multistep task are those users who are authorized to start the workflow connected with the task manually.

You only have to assign possible agents to a multistep task if the workflow is to be started directly by users. If the workflow is only to be started by events, do not assign any possible agents to the multistep task.

**Task groups**

If you assign possible agents to a task group, this assignment in passed on to all tasks contained in the task group.

Check whether the assignment defined here really is to be apply to all task groups, single-step tasks and multistep tasks contained.

**Basic properties of tasks**

**Task**

Choose the basic property of the task from the following alternatives:

*General task*

If you define a task as a general task [Extern], all users can execute the task. This is useful if the task is used in a workflow and you only want to define the recipients in the step definition. A recipient can forward associated work items to all users.

Work items whose tasks are defined as general tasks and for which no responsible agents or default roles are defined are offered to all users of the R/3 System for execution in their Business Workplaces.

*General forwarding allowed*

A work item that represents a task with this property can be forwarded by one of its recipients to all users, even if they are not possible agents of the task.

*General forwarding not allowed*

A work item that represents a task with this property can be forwarded by one of its recipients only to the possible agents of the task.

**Multistep task**

With a multistep task you can only decide whether it is a general task or not.

**Activities**

To process the basic properties of a task, choose Additional data → Classification → Change in task maintenance.

To assign a task you have defined to its possible agents or process its basic properties, choose Additional data Agent assignment Maintain and select the task.

Select Attributes... to define the basic properties of the task.

Choose Agent assignment → Create... to assign the possible agents.

You can process the possible agents of the task and workflow (multistep task) from the Workflow Builder as well.
Maintaining Task Texts

Prerequisites
You enter the following information on the tab page Description:

- **Task description**
  You can describe the task or make notes and recommendations for processing. The task description is displayed in the work item display.

- **Notification texts**
  These texts are sent as notification to the message recipients for completion [Extern] or missed deadlines [Extern].

Procedure
10. Switch to the tab page Description.
11. Select the text type that you want to create or change.
12. Select 📝.

If you are looking at a task in display mode, whose original system is not the current system, you can change the existing texts. This creates an extension of the task definition.

7. Compose your text.

To add text variables to your text, choose Include → Expression. You can add as many text variables to the text as you want. For further information, refer to Text Variables and Replacement [Extern] and Text Processing with the SAPscript Editor [Extern].

You want to display the number of the accounting document currently being processed in the task description for a standard task. The document number is defined under the name DocumentNo as a key field for the object type BUS2068 (accounting document).

The reference to the processed object (of type BUS2068 in this example) is located in the task container under the element _WI_Object_ID. To reference the document number in the text, write &BUS2068.DocumentNo& at the appropriate point in the text.
Definition of the Workflow Container

Use
In the definition of the workflow container [Extern], you process the container elements required for the workflow execution. This may be data that you require for controlling the workflow execution directly, or data that you want to forward from one step execution to another. Every workflow container already contains workflow system elements [Extern].

⚠️ The system no longer evaluates activity-related container elements as of Release 4.6A. The IDs of the activity-related container elements begin with ACTIVITY_<StepNumber>_.

You can assign initial values to container elements at the start of the workflow. You must carry out this initial value assignment in the definition of the multistep task. For further information, refer to Assigning Initial Values [Seite 1205].

Integration
You can process the workflow container both during definition of the multistep task and from the Workflow Builder.

Features
Container elements for which the import or export indicator is set form the interface of the workflow.

Definition of the data interface
Both the import and the export indicators can be set for container elements.

Container elements of the interface are not subject to the versioning [Seite 1019] of the workflow definition.

Container elements with import indicator set
These container elements hold the values that can be passed when execution of the workflow is started. If the values have to be passed, set the indicator mandatory as well.

These container elements can be supplied with values in the following ways at the start of execution:

- Via a binding from the event container [Seite 1227] if the workflow has one or more triggering events
- Via initial value assignment [Seite 1205]
- Via an electronic form [Seite 1722] if the workflow is started with it
- Via direct entry if the workflow is started with the function Start workflow (test environment) [Seite 1452]

If the object to be processed is not created in the workflow, a container element with import indicator set must be defined in the workflow container, which holds the object reference. You create this container element with a data type reference to the relevant object type.
Definition of the Workflow Container

**Container elements with export indicator set**
These container elements hold the values which the workflow is to return to a calling workflow. You need them if the workflow is a subworkflow of a superordinate workflow and is to return data to its superordinate workflow.

**Definition of local container elements**
All container elements for which neither the *import* nor the *export* indicator is set are local container elements of the workflow definition. They are only valid in this workflow definition. Typical applications are:

- Passing data from one task container to another
- Determining the responsible agents for an activity dynamically
- Setting deadlines dynamically
- Implementing counting loops

All local container elements are subject to the *versioning* ([Seite 1019](#)) of the workflow definition.

If you create a local container element in the Workflow Builder, it is only included into the workflow definition currently being processed. If you create it in the definition of the multistep task, it is included into all versions existing at the time.

For further information, refer to [Creating Container Elements](#).
Creating Container Elements

Prerequisites
You are in one of the definition tools and want to add a new element to the container.

Procedure
25. Select 
26. Select .
27. Enter the technical name of the container element into the field Element.
   You give each container element a technical name that can be used to identify it uniquely. The technical name must begin with a letter. This can be followed by letters, underscores or digits. Not case-sensitive. The technical name you choose should be in English.
28. Under Texts maintain the name and the description.
   This text can include upper and lower cases.
29. Under Properties, select whether the new element is to be an import and/or an export element. Mark the element as mandatory if applicable.

The values for each of the container elements may be in the form of a multiline list of field values or object references. This list is then saved under an element name. You can select multiline in the container definition so that this container element can hold a multiline list of values.
30. According to the data type reference [Extern] of the container element, make the following entries. If the new container element:
   – Is to reference a structure, enter the table or structure in the field Reference table
   – Is to reference an ABAP Dictionary field, specify a reference field as well
   – Is to reference an object type, enter the ID of the object type in the field Object type
   You can use the F4 input help for all of these.

   Specification of a ABAP Dictionary field reference is not mandatory. If no field reference is specified, the data type is C(255) (upper and lower case). Specification of an object type is not mandatory either. If no object type is specified, the container element can be assigned a reference to an object of any type.
11. Select .
Assigning Initial Values

Use
The initial value assignment of constants to container elements should be regarded as a preassignment of values to container elements.

Procedure
1. Select **Initial values**.
   
   All container elements designated as import elements are displayed. The import workflow system elements are not displayed until you select **Initial values**, except for _WI_Object_Id.
   
   ![Image](image)
   
   You can only assign values to those container elements that are designated as import elements.

2. Enter the initial value in the field **Expression**.
   
   To assign several values to a target element defined as multiline, create the additional input lines using **Expression**.

Result
At runtime the system executes the initial value assignment in accordance with your definition.

![Image](image)

If there is a binding definition from another container to the container with the initial values, the system executes this binding at runtime after the initial value assignment. Any initial values are overwritten.
Creating Triggering Events

Procedure

You maintain the triggering events [Extern] on the tab page Triggering events.

13. Identify the event by specifying its instance ID and object type. The event must be defined for this object type.

   If you are looking at a task in display mode, whose original system is not the current system, you can create additional triggering events. This creates an extension of the task definition.

14. To activate the event receiver linkage, select in the column . Active event linkages are denoted by .

15. To define the binding from the event container, select the relevant event line and choose . For further information, refer to Binding Definitions from the Event Container [Seite 1227].

16. Select the triggering event and choose . The properties of the event linkage are displayed and can be changed.

Result

The workflow system enters the linkage between the event and the event-receiving task in the relevant linkage table [Extern] automatically. For further information on linkage transport, refer to Transport and Client Copy [Seite 1528].

   If an error occurs when a task or workflow is being started with an event, the standard procedure is for the system to deactivate the event linkage to avoid any follow-on errors. At the same time, the system sends an e-mail to the workflow system administrator. But you can change this in event queue administration [Seite 1515].
Maintenance of SAPphone Properties

Use
The SAPphone properties determine the possible uses of a task in the SAPphone environment.

Integration
SAPphone integrates telephony functions into the R/3 System. R/3 applications use this to integrate the telephone into the work process. SAPphone provides the Basis technology required.

The processing of outgoing and incoming telephone calls is supported with data from the R/3 System environment (for example, telephone number or name). An incoming call starts a search for caller data. The result of the search is displayed automatically in an information window. For further information, refer to SAPphone [Extern].

Features
If you label the task accordingly, it is displayed and can be executed in the event of incoming or outgoing calls. The task container must contain appropriate container elements to enable the information derived from the telephone data to be used in the task.
Definition of a Task Group

Use
You define task groups if you want to collect and group standard tasks and workflow templates that are logically related. You can also collect other task groups within a task group, hence making hierarchies of task groups.

Task groups are not restricted within applications. You can collect tasks from within one application component as well as tasks from different application components.

A standard task or workflow template can be used in several task groups.

You can use task groups to delimit the search range when calling the Business Workflow Explorer [Seite 975].

Integration
The application component is also available as a collective evaluation criterion for standard tasks and workflow templates. Each task is assigned to an application component automatically via its development class.

Features
You define the task group with its basic data and enter the tasks and task groups that it is to contain. Enter a description for the task group.

Agent assignment
It is not necessary to assign possible agents to a task group. If you do, the possible agents are passed on to the tasks and task groups contained.

Activities
You should adhere to the following sequence when defining single-step tasks (and analyzing existing single-step tasks):

<table>
<thead>
<tr>
<th>What?</th>
<th>Where?</th>
</tr>
</thead>
<tbody>
<tr>
<td>Maintain abbreviation and name. The system assigns the identifying number of the task group when saving.</td>
<td>Tab page Basic data</td>
</tr>
<tr>
<td>Maintain task description [Seite 1201].</td>
<td>Tab page Description</td>
</tr>
<tr>
<td>Enter tasks and task groups collected in this task group.</td>
<td>Tab pages Standard tasks, Workflow templates and Task groups</td>
</tr>
<tr>
<td>Maintain agent assignment [Seite 1198]</td>
<td>Extras → Agent assignment → Maintain</td>
</tr>
<tr>
<td>Graphical representation of the hierarchy of the task group, including the subordinate task groups.</td>
<td>Hierarchy</td>
</tr>
</tbody>
</table>
Definition of a Container

Use
The elements of a container are described with ID, data type reference and other properties. Container definition is carried out with the container editor.

Container

Table-like data structure for holding
- Field values (including multiline)
- Object references (including multiline)

<table>
<thead>
<tr>
<th>Element ID</th>
<th>Data type reference</th>
<th>Description</th>
<th>Multiline</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>CostCenter</td>
<td>VBAK-KOSTL</td>
<td>Cost center</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Customer</td>
<td>KNA1</td>
<td>Customer</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Items</td>
<td>VBAP</td>
<td>Document items</td>
<td>X</td>
<td></td>
</tr>
</tbody>
</table>

Integration
A container definition is not an independent action. Container definitions are always carried out within a definition tool using the editor provided.

Features

Container definition
To describe which data a container can hold at runtime, of which type and under which name, the container must first be defined. This container definition defines which container elements a container holds. You can define the following containers as required:

- Method container
- Event container
- Role container
- Task container
- Workflow container
Definition of a Container

Usually containers are already defined. These containers hold standard container elements defined by the workflow system. These container elements will often be adequate for your requirements meaning that a more extensive container definition will not be necessary.

Container instance
A container instance is an instance of a container at runtime. It contains a value of the defined data type for each container element.

Binding
Values can be assigned to container elements in different ways.

- Value assignment via a binding [Seite 1218] from another container according to a binding definition carried out previously.
- Value assignment directly to a container instance in the program of an application.
- Value assignment with an initial value determined at definition time when the container instance is created.

When processing container instances within a program for object type implementation (write and read accesses), use the macro instructions [Seite 1133] provided.

Activities
You can perform these activities from the container editor. The editor is called at the following places:

- For the method container or the event container in the Business Object Builder

  When you create a method parameter or an event parameter, it is automatically stored in the relevant container.

- For the task container or the workflow container in the task definition or the workflow definition

- For the role container in the definition of the standard roles

Creating container elements
To create a new container element [Seite 1212], select .

Editing container elements
Double-click on the relevant table entry to edit a container element.

Copying, cutting or pasting container elements
You select an element in a container by positioning the cursor on the relevant line and choosing . You can copy or cut selected container elements. If they are pasted in again, you must rename them.

Display conventions
The following conventions apply in the container editor:

- Workflow system elements have a green background and are locked for processing
- Container elements with import or export indicator set have a gray background
- Local container elements have a white background
Container elements have the following symbols in the Workflow Builder:
Definition of a Container

- Local container element
- Container element with import indicator set
- Container element with import and export indicators set
- Container element with export indicator set
Creating Container Elements

Prerequisites
You are in one of the definition tools and want to add a new element to the container.

Procedure
31. Select \( \text{Container} \).
32. Select \( \text{.} \).
33. Enter the technical name of the container element into the field \( \text{Element} \).
   
   You give each container element a technical name that can be used to identify it uniquely. The technical name must begin with a letter. This can be followed by letters, underscores or digits. Not case-sensitive. The technical name you choose should be in English.

34. Under \( \text{Texts} \) maintain the \( \text{name} \) and the \( \text{description} \).
   
   This text can include upper and lower cases.

35. Under \( \text{Properties} \), select whether the new element is to be an \( \text{import} \) and/or an \( \text{export} \) element. Mark the element as \( \text{mandatory} \) if applicable.
   
   The values for each of the container elements may be in the form of a multiline list of field values or object references. This list is then saved under an element name. You can select \( \text{multiline} \) in the container definition so that this container element can hold a multiline list of values.

36. According to the data type reference \( \text{[Extern]} \) of the container element, make the following entries. If the new container element:
   
   - Is to reference a \( \text{structure} \), enter the table or structure in the field \( \text{Reference table} \)
   - Is to reference an \( \text{ABAP Dictionary field} \), specify a reference field as well
   - Is to reference an \( \text{object type} \), enter the ID of the object type in the field \( \text{Object type} \)
   
   You can use the F4 input help for all of these.

   Specification of a ABAP Dictionary field reference is not mandatory. If no field reference is specified, the data type is \( \text{C(255)} \) (upper and lower case). Specification of an object type is not mandatory either. If no object type is specified, the container element can be assigned a reference to an object of any type.

12. Select \( \checkmark \).
Definition of the Workflow Container

Use
In the definition of the workflow container, you process the container elements required for the workflow execution. This may be data that you require for controlling the workflow execution directly, or data that you want to forward from one step execution to another. Every workflow container already contains workflow system elements.

⚠️ The system no longer evaluates activity-related container elements as of Release 4.6A. The IDs of the activity-related container elements begin with ACTIVITY_<StepNumber>.

You can assign initial values to container elements at the start of the workflow. You must carry out this initial value assignment in the definition of the multistep task. For further information, refer to Assigning Initial Values.

Integration
You can process the workflow container both during definition of the multistep task and from the Workflow Builder.

Features
Container elements for which the import or export indicator is set form the interface of the workflow.

Definition of the data interface
Both the import and the export indicators can be set for container elements.

Container elements of the interface are not subject to the versioning of the workflow definition.

Container elements with import indicator set
These container elements hold the values that can be passed when execution of the workflow is started. If the values have to be passed, set the indicator mandatory as well.

These container elements can be supplied with values in the following ways at the start of execution:

- Via a binding from the event container if the workflow has one or more triggering events
- Via initial value assignment
- Via an electronic form if the workflow is started with it
- Via direct entry if the workflow is started with the function Start workflow (test environment)

If the object to be processed is not created in the workflow, a container element with import indicator set must be defined in the workflow container, which holds the object reference. You create this container element with a data type reference to the relevant object type.
Container elements with export indicator set
These container elements hold the values which the workflow is to return to a calling workflow. You need them if the workflow is a subworkflow of a superordinate workflow and is to return data to its superordinate workflow.

Definition of local container elements
All container elements for which neither the import nor the export indicator is set are local container elements of the workflow definition. They are only valid in this workflow definition. Typical applications are:

- Passing data from one task container to another
- Determining the responsible agents for an activity dynamically
- Setting deadlines dynamically
- Implementing counting loops

All local container elements are subject to the versioning [Seite 1019] of the workflow definition.

If you create a local container element in the Workflow Builder, it is only included into the workflow definition currently being processed. If you create it in the definition of the multistep task, it is included into all versions existing at the time.

For further information, refer to Creating Container Elements [Seite 1212].
Definition of the Task Container

Use
This function is used to process the container elements of the task container [Extern]. Each task container already contains the workflow system elements [Extern] and any additional elements for the method parameters of the referenced object method.
You have to add additional elements [Seite 1212] to the task container if:

- Additional values are required for variable replacement in the work item text or in the long texts
- Additional information is required to resolve a default role

Values are assigned to the task container at runtime in the following ways:
- By initial value assignment [Seite 1205]
- Via binding from the workflow, event or method container

Initial values are overwritten by bindings. Initial value assignment and processing of the task container are only possible during definition of the single-step task.
Definition of the Method Container

Use
The method container holds the method parameters [Extern], which are used for information exchange between the caller of a method and the method. At runtime, method parameters are:
- Passed to the method (import parameters)
- Returned from the method (export parameters)
The definition of the parameters determines the interface of the method call.
The method container is called when the binding from the method to the single-step task is called.

Features
All method parameters are container elements of the method container, which is created automatically by the system when the method is defined.

The object reference to the object on which the method operates is not a container element of the method container.

You can display the content of the method container from the binding definition in the container editor.
Definition of Event Container

Use
The event container contains the event parameters [Extern] as container elements. If you define triggering [Extern] events or terminating [Extern] events for a task, you can specify the binding from the event container to the task container. If you define triggering events for a workflow, you can specify the binding from the event container to the workflow container.

Features
The event container contains workflow system elements [Extern] as standard. You can add more container elements to the event container within object type definition in the Business Object Builder. This is not usually necessary, however.
Definition of Binding

Use
At definition time you create binding definitions for the workflow. This involves specifying the assignment rules for how data is exchanged between two containers. At runtime these binding definitions are executed making workflow execution possible. You can also allocate values to container elements using initial value assignment at definition time. These initial values are always overwritten by a binding.

Application
Within the transaction for creating an accounting document (object type AccountingDocument), an event is generated (event created), which indicates that a document has been successfully created. This event is to be the triggering event of a workflow.

Event
The event container is passed with this event. This container always contains an element _Evt_Object with a reference to the object created.

Workflow container
In the container of the workflow that is to be started by the event created of the object type AccountingDocument, you have created a mandatory import element (under the ID AccountingDocument for example) with a data type reference to the object type AccountingDocument. The element can therefore hold the reference to the object created (type AccountingDocument) at runtime.

Binding to the workflow
The event parameter (= element in event container) _Evt_Object can be passed in a binding to the workflow container of the workflow that is to be started in response to the event. You define the following binding for this:

_evt_Object (event container) ➔ AccountingDocument (workflow container)

Features
The graphic below illustrates which bindings can be defined.
Possible Binding Definitions

Binding definition from task container
You define the bindings from the task container in the following two places.

In the definition of a task
There are binding definitions to the:
- Method container (for import parameters of method).
- Role container (for definition of default roles).

In the step definition in the Workflow Builder
The binding definition to the workflow container.

Binding definition from workflow container
You define the bindings from the workflow container within the Workflow Builder. There are binding definitions to the:
- Task container (activity, user decision, document from template, send mail).
- Event container (event creator).
- Role container (for definition of responsible agents and recipients).

Binding definition from method container
You define the binding from the method container to the task container (for export parameters of the method with synchronous methods) during definition of a task.

Binding definition from event container
You define the bindings from the event container in the following three places.
Activities

- In the definition of a task for the binding definition to the task container (for triggering or terminating events of the task).
- In the Workflow Builder in the basic data of a workflow for the binding definition to the workflow container (for triggering events of the workflow).
- In the Workflow Builder in the definition of a wait step for the binding definition to the workflow container.

The system makes proposals automatically for the binding definition between the workflow container and the task container. You can change these proposals. For further information, refer to Rule-Based Proposal for Binding Definitions [Seite 1221].

Activities

You define the bindings in the binding editor [Seite 1232], which is always called via a definition tool. This cannot be done separately.
Rule-Based Proposal for Binding Definitions

Use
The workflow system automatically makes proposals for the following binding definitions:
- Binding definition from workflow container to task container
- Binding definition from task container to workflow container
- Binding definition from task container to method container
- Binding definition from method container to task container

If additional elements are required in a container for this proposal, the system offers to create these elements.

Features
The system evaluates the data type references of the relevant container elements and follows specific rules to generate a proposal for the binding definition.
If the system cannot establish a valid binding definition with the elements available, it proposes an appropriate new element for the workflow container and a binding definition based on this new element.
If the system can establish more than one valid binding definition with equal probability using the elements available, it does not propose any binding definition.
The system displays the proposals for new elements in the container in the upper part of the screen and the proposals for the binding definitions in the lower part of the screen.

Rules for binding generation
For each binding definition proposal, you can display the rules used by the system to establish the proposal. To do this, double-click or of the binding definition.

Further Information
If the system does not make any proposals for the binding definition to the target container for a particular element, you can display the reasons behind this.
To do this, choose Further information. A dialog box with a list of the relevant elements is displayed.
Select an element and choose for further information.

Activities
You can adopt the proposed container elements and binding definitions without changing them.
If you do not want to adopt a proposal for a new element or a binding definition, you proceed as follows:
4. You deselect the proposal that you do not want to adopt.
5. You select . If applicable, the binding definition that uses a deselected container element is then also deselected.
6. To adopt a proposal after changes, you select .
If you want the system to create a new element in the container for this binding, even though a proposal has been made for the binding using an existing element, you proceed as follows:
4. You position the cursor on the line with a proposed binding.
5. You select . The system then proposes a new element for the container, which has the same name as the element in the source container.
Rule-Based Proposal for Binding Definitions

The system enters this proposal in the upper part of the screen. The proposal for the binding refers to the new element.

6. To adopt the proposal after changes, you choose 📐. You can also make more changes as described above.
Binding Definitions from Workflow Container

Use
You can define bindings from the workflow container to the following containers:

- Task container
- Role container
- Event container

You can allocate an expression to the elements of the target container, which refers to an element of the workflow container.

Features

Binding definition to task container
This binding can be defined for step types that refer to a task.

The workflow system proposes this binding definition. If new elements are required in the workflow container, proposals are also made for these. You can edit and confirm these proposals or reject them. For further information, refer to Rule-Based Proposal for Binding Definitions.

Special information on object reference
A binding must generally be defined from the element in the workflow container with the reference to the object to be processed, to the element _WI_Object_ID in the task container. Only in this way can the right object be processed with the single-step task.

- If the workflow container contains an element which is defined with a data type reference to the same object type that is created in the referenced single-step task, this binding definition is proposed automatically when an activity is defined.

- If no element with an appropriate data type reference is available in the workflow container, a suitable element is added to the workflow container and the binding entered.

Binding definition to event container
This binding definition is only possible for the step type event creator.

- The binding definition to the event container is not necessary if the latter only contains the standard event parameters. These standard event parameters are assigned values automatically from the information you specify when defining the event creator.

- The binding definition to the event container is necessary if the latter contains elements other than the standard event parameters.

Binding definition to role container
This binding definition is only necessary if you use a role for establishing the agents or recipients.
Activities
For further information, refer to Operation of the Binding Editor [Seite 1232].
Binding Definitions from Task Container

Use
You can define bindings from the task container to the following containers:

- Workflow container
- Role container
- Method container

You can allocate an expression to the elements of the target container, which refers to an element of the task container.

Features

Binding definition to workflow container
The binding is applied after execution of the work item. The binding can be defined in step type definitions that refer to a task. It is necessary if you require information for the workflow, which is available after execution of the underlying object method.

Special information on object reference
If an object is created (for example, with the method Create) or established (for example, with the method Find) in task processing, you must define the binding from the element _WI_Object_ID in the task container to the element in the workflow container, which can contain the object reference. Only then is this information available in the workflow and therefore for the subsequent steps.

- If the workflow container contains an element which is defined with a data type reference to the same object type that is created in the referenced single-step task, this binding definition is proposed automatically when an activity is defined.
- If no element with an appropriate data type reference is available in the workflow container, a suitable element is added to the workflow container and the binding entered.

Current agent: Element _WI_Actual_Agent in task container
If you want to transfer the current agent, available under the element ID _WI_Actual_Agent in the task container, to the workflow container for further use, you define an appropriate element in the workflow container to contain this information.

You create this element with a reference to the ABAP Dictionary field WFSYST-AGENT.

Result of the object method: Element _WI_Result in the task container
The result of the object method is available under the element ID result (_WI_Result) in the task container. You must define an appropriate element in the workflow container, if you want to transfer the result into the workflow container. When defining the container element, use the same data type reference that the element _WI_Result has in the task container.

You require the result of this activity in a subsequent step of the type loop (UNTIL) in order to formulate the loop condition.
Binding definition to role container
You define this binding in the definition of a single-step task if you enter roles on the tab page Default roles.
You must add the elements required to resolve the role to the task container in order to define a binding between these new elements and the appropriate elements of the role container.

> Note that you must create a binding from the workflow container to the task container in the step definition, via which these "new" container elements are given their values.

Binding definition to method container
You define this binding in the definition of a single-step task. It is only necessary if import parameters are defined for the object type method.
The workflow system proposes this binding definition if you have specified an object method with parameters during definition of the single-step task. If new elements are required in the task container, proposals are also made for these. You can edit and confirm these proposals or reject them. For further information, refer to Rule-Based Proposal for Binding Definitions [Seite 1221].
You can modify this binding definition. But this will usually only be necessary in exceptional situations.

Catering for the container elements _WI_Object_ID
The container element _WI_Object_ID in the task container is not incorporated into the binding definition to the method container. At runtime, the workflow system ensures that the object reference from the element _WI_Object_ID is available in the key fields in the program of the object type.

Activities
For further information, refer to Operation of the Binding Editor [Seite 1232].
Binding Definitions from Event Container

Use
You can define bindings from the event container to the following containers:

- Task container
- Workflow container

You can allocate an expression [Seite 1235] to the elements of the target container, which refers to an element of the event container.

The event may contain information that you want to use in the workflow. If so, you must read this information from the event container for the event. In order for the information that is transported as event parameters in the container for this event to reach the workflow, you can define the binding from the event container to the workflow container in the wait step.

Features

Binding definition to workflow container
This binding can be defined at the following places:

- In the definition of the multistep task on the tab page Triggering events
- In the Workflow Builder in the basic data of the workflow on the tab page Start
- In the Workflow Builder in the step definition of a wait step on the tab page Control

This binding enables you to transfer event parameters of a triggering event [Extern] or an event for which the workflow waits during execution into the workflow.

You can only use import elements of the workflow container in a binding definition from event parameters of a triggering event to elements of the workflow container.

Binding definition to task container
This binding can be defined in the definition of the task on the tab page Triggering events.

Activities

Assigning the object reference for triggering events
If you want to use the reference to the object whose status change the event reports in the task or workflow, you must assign this reference to an element of the task container or the workflow container.

The object reference is located in the event container under the element ID _Evt_Object.

- In the task container, the reference to the object to be processed must always be located in the element _WI_Object_ID.

- No element is provided for the object reference in the workflow container as standard. You define this element in the workflow container with a data type reference [Extern] to the relevant object type and assign it the value of the element _Evt_Object from the event container in the binding definition.
Assigning the event creator for triggering events

If you want to use the information about the user under whose responsibility the event was created in the task or workflow, you must assign this information to an element in the task container or workflow container.

The user name of the employee under whose responsibility the event was created is located in the event container under the element ID _Evt_Creator.

- No element is provided for this information in the task container as standard. If applicable, you define this element in the task container with a data type reference to the ABAP Dictionary field WFSYST-AGENT and assign it the value of the element _Evt_Creator from the event container in the binding definition.

- The element _WF_Initiator is provided for this information in the workflow container. In the binding definition, you assign this element the value of the element _Evt_Creator from the event container.

For further information, refer to Operation of the Binding Editor [Seite 1232].
Binding Definition from Method Container

Use
You can define a binding to the task container.
You can allocate an expression [Seite 1235] to the elements of the task container, which refers to an element of the method container.

Features
You define this binding in the definition of a single-step task. It is only necessary if export parameters are defined for the object type method (only for synchronous methods [Extern]). The binding from the method container to the task container is executed when a synchronous object method has been processed successfully.
The workflow system proposes this binding definition if you have specified an object method with parameters during definition of the single-step task. If new elements are required in the task container, proposals are also made for these. You can edit and confirm these proposals or reject them. For further information, refer to Rule-Based Proposal for Binding Definitions [Seite 1221].
You can modify this binding definition. But this will usually only be necessary in exceptional situations.

Catering for the container elements _WI_Object_ID and _WI_Result
The container elements _WI_Object_ID and _WI_Result of the task container are not incorporated into the binding definition between the method container and the task container.
At runtime, the system ensures the following:
- If an object was created for a method, the reference to the object created is entered in the container element _WI_Object_ID.
- The result (if applicable) of the method is entered in the container element _WI_Result.

Activities
For further information, refer to Operation of the Binding Editor [Seite 1232].
Binding Definition for Document from Template

Use
In the binding definition you enter elements of the workflow container, which are to either pass values to the task container or receive values from the task container after processing. Binding definitions are only necessary for the step type Document from template under the following circumstances:

- If you use your own single-step task
- If you have used expressions from the task container in the title or in the task texts of the single-step task
- If an attachment is to be created based on a document template
- If the creator of the document is still required in the workflow

The binding for the document templates required, which you specified on the tab page Document templates, is defined automatically. The binding from the workflow container to the task container for the element Flowitem and the binding from the task container to the workflow container for the element _WI_Object_Id are also defined automatically.

Features
The following container elements in the task container are particularly significant for this binding:

_WI_Actual_Agent (actual agent)
If you want to transfer the agent for the document generation to the workflow container for further use, you define an element with the ABAP Dictionary reference WFSYST-AGENT there to hold this information.

Note_Reference (template for notes)
If you assign a workflow container element containing a template (business object reference SOFM) to this container element, the agent can enter a document as an attachment at runtime, which is based on this template.

Activities
For further information, refer to Operation of the Binding Editor [Seite 1232].
Binding Definition for Decision Tasks

Use
In the binding definition you enter elements of the workflow container, which are to either pass values to the task container or receive values from the task container after processing.

For a user decision, a binding definition is only necessary from the workflow container to the task container of the user decision under the following circumstances:

- If you use your own decision task for the user decision
- If you have used expressions from the task container in the title or in the task texts of the decision task
- If an attachment is to be created based on a document template

A binding definition from the task container to the workflow container is only required for a user decision if you want to store the result or the current agent of the user decision in an element in the workflow container in order to access it again in a subsequent step.

Features
The following container elements in the task container are particularly significant for this binding:

complexContent

_result (result)
The result of the user decision is available in this element as a text field after processing. Depending on the number and sequence of possible decision options, these are the values 0001, 0002 up to nnnn. A binding definition is only required for this element if you need this value in the remainder of the workflow definition. This may be the case, for example, if you need the value again as a indicator in a decision. To implement this, you define an appropriate element (ABAP Dictionary reference SWD_LINES-RETURNCODE) in the workflow container to hold this user decision result.

_actual_agent (actual agent)
If you want to transfer the agent for the user decision to the workflow container for further use, you define an element with the ABAP Dictionary reference WFSYST-AGENT there to hold this information.

Note_Reference (template for notes)
If you assign a workflow container element containing a template (business object reference SOFM) to this container element, the agent can enter a document as an attachment at runtime, which is based on this template.

Activities
For further information, refer to Operation of the Binding Editor [Seite 1232].
Operation of the Binding Editor

Use
The binding editor is used to define bindings between two containers. In the case of binding definition between task container and workflow container as well as between task container and method container, you can process both directions of binding at the same time. You can only process one direction of binding for the remaining binding definitions.

Integration
Each binding definition is linked to the execution of an action which processes the data provided and produces results that can be returned. The following actions can be carried out:

- Execution of a step
- Execution of a single-step task or multistep task
- Execution of an object type method
- Resolution of a role

Features
The container elements of the target container of the binding definition are listed on the left-hand side. You can use to switch between displaying the technical name and displaying the element name. Only the following container elements in the target container are displayed as standard:

- Container elements with import or export indicator set, which are not workflow system elements
- _WI_Object_Id

In order to display the remaining workflow system elements in the target container as well, you select . The binding definition from the source container to the target container (binding direction ) is processed in the upper part of the screen. The associated binding is executed at runtime before the action. This binding definition is always available.

The binding definition from the target container to the source container (binding direction ) is processed in the lower part. The associated binding is executed at runtime after the action. This binding definition is only available if the action can produce results that can be made available to the source container.

Binding Definition from Source Container to Target Container
Here you process the allocation of values to elements in the target container. Only the elements in the target container for which the indicator Import is set are displayed. You can allocate an expression to each element of the target container.

For further information, refer to Binding Definitions from the Source to the Target Container.

Binding Definition from Target Container to Source Container
Here you process the allocation from elements in the target container. Only the elements in the target container for which the indicator Export is set are displayed. You can allocate a container element in the source container to each element of the target container. You can allocate the container element in the source container to a compatible container element in the target container
To a field of a structure, if the container element of the target container is defined with structure reference.

You can allocate one element in the source container to several elements in the target container. To make the source element available several times as a starting point for the binding definition, you select the source element and choose $\leftarrow$. For further information, refer to Binding Definitions from the Target Container to the Source Container [Extern].

Definitions of containers involved
You can call up the full definitions of the two containers involved by using the pushbuttons with the same names. Important information is displayed here about the data type references of the container elements.

Permitted entries for expressions
Permitted expression types for binding directions

<table>
<thead>
<tr>
<th>Expression type</th>
<th>Permitted binding direction</th>
<th>Explanation</th>
</tr>
</thead>
<tbody>
<tr>
<td>&amp;&lt;Element&gt;&amp;</td>
<td>$\leftarrow$ and $\rightarrow$</td>
<td>Source and target element with compatible field references</td>
</tr>
<tr>
<td>&amp;&lt;Object.Attribute&gt;&amp;</td>
<td>$\leftarrow$</td>
<td>Source and target element with compatible field references</td>
</tr>
<tr>
<td>&amp;&lt;Object.Object.Attribute&gt;&amp;</td>
<td>$\leftarrow$</td>
<td>Source and target element with compatible field references</td>
</tr>
<tr>
<td>&amp;&lt;Object&gt;&amp;</td>
<td>$\leftarrow$ and $\rightarrow$</td>
<td>Source and target element with compatible object references</td>
</tr>
<tr>
<td>&amp;&lt;Object.Object&gt;&amp;</td>
<td>$\leftarrow$</td>
<td>Source and target element with compatible object references</td>
</tr>
<tr>
<td>&amp;&lt;Structure&gt;&amp;</td>
<td>$\leftarrow$ and $\rightarrow$</td>
<td>Source and target element with compatible structure references</td>
</tr>
<tr>
<td>&amp;&lt;Structure&gt;-&lt;Table-Field&gt;&amp;</td>
<td>$\leftarrow$</td>
<td>Source and target element with compatible field references</td>
</tr>
<tr>
<td>Const</td>
<td>$\leftarrow$</td>
<td>Assignment of a constant</td>
</tr>
</tbody>
</table>

It is not possible to assign the value of an element in the target container to an object attribute in the source container.
Alternative Binding Definition

Use
As an alternative to describing a binding in the binding editor, you can define a binding by specifying a function module. This type of binding definition is particularly useful as an alternative to the line-oriented binding definition described up to now, if:

- You work on both sides of the binding definition with structure references and the binding editor does not enable the desired assignments
- You want to perform calculations or checks on the container elements before assignment
- You want to combine two or more container elements in some way before assignment

Features
To execute the binding, the system calls a function module, to which the two containers are passed as table parameters. This function module is programmed especially for this binding and is entered when the alternative binding is defined.
The binding definition based on container elements is deleted.

Activities
You define an appropriate function module for each binding direction. The interface for these function modules is defined. Use the function module SWA_TEMPLATE_EXECUTE_DATAFLOW as a template. The function module documentation contains all the required information on the interface and programming.
To enter the function modules, you choose Alternative binding in the binding editor context menu and make the entries in the relevant input field.
Using Expressions in Binding Definitions

Use
You use an expression to assign a value to a container element in the binding definition.

Features
Expressions must be assigned using the F4 input help. You have the following options for assigning values:

Constant
Fixed value or fixed object reference, which is assigned to a container element at runtime.

Container
Direct assignment of a container element or assignment using a container element from the source container.

- The value of the variable is derived directly from the container element
  The variable is created in the form &<ElementName>&

- The value of the variable is derived indirectly as an attribute or a key field of the object referenced in the container element.
  The variable is specified in the form &<ElementName>.<AttributeName>&.
  This indirect establishment of values can also have two or more levels. The variable then appears in the form
  &<ObjectReference>.<ObjectReference>.<...>.<AttributeName>&.

- The value of the variable is derived indirectly as a field value of the structure that is referenced in the container element.
  The variable is specified in the form &<ElementName>.<Table>-<Field>&.

System field
Field in table SYST.
The variable is specified in the form %<SystemFieldName>%. Depending on the environment, not all the expression components named are always offered.

Activities
Use the F4 input help when making assignments. Bear in mind the color legend. The colors indicate whether an assignment is permitted as far as the data types of the two partners are concerned, and whether the assignment may be defined. As an alternative to the “technical” element ID, you can display the name of the container element by choosing 

Activity
Choose between constant, container and system field to specify the type of expression. If you specify the expression as the target of a binding definition, you can only choose container.
Entries can occur twice when container elements are displayed, the second entry having an addition.

Addition "with index"
The addition "with index" only appears for a multiline container element if:
- The container element is a source element of a binding definition.
• The container element is entered as a multiline container element on the tab page Other
If this entry is chosen, only one line of the multiline container element is assigned to the single-line target element in the binding. For further information, refer to Binding Definitions from Source Container to Target Container [Extern].

This application should only be used within dynamic parallel processing.

Addition "add instead of change"
The addition “add instead of change” only appears for a multiline container element, if the container element is a target element of a binding definition. This addition means that the value of the source element is appended to the content (if any) of the target element. For further information, refer to Binding Definitions from Target Container to Source Container [Extern].
Document Processing

Use
You can use a document in the following ways in SAP Business Workflow:

- Send it as a "mail" from a workflow
- Create, display or edit it in a workflow step
- Add it to a workflow as an attachment

Features

Sending or displaying a text created at definition time
A text entered at definition time of a workflow and supplemented at runtime with information from the context of the workflow can be sent as a mail, and is displayed in the work item preview of the Business Workplace.

Adding a document to a workflow as an attachment
A document can be assigned to a workflow as an attachment. The recipients of the work items in this workflow can display this attachment in the work item preview.

Processing standard texts in a workflow
A SAPscript document can be created, edited or displayed in a workflow step. It can be supplemented with information from the task container.
Processing SAPscript Texts as a Step in the Workflow

Use

*SAPscript standard texts* can be entered and displayed at runtime as part of a workflow.

Integration

The object type `STD_TEXT (SAPscript standard text)` is defined in the *Business Object Repository* with methods for creating, editing and printing.

Activities

If you want to enter a text as a step in a workflow *at runtime* and this text is to be read by another employee in one of the subsequent steps, the following activities are required:

4. Define a single-step task for entering a text. The user who executes the work item in question at runtime has to enter a text. For further information, refer to *Defining the Standard Task "Create SAPscript Text"* [Seite 1239].

5. Define a single-step task for displaying the text. For further information, refer to *Defining the Standard Task "Display SAPscript Text"* [Seite 1240].

6. Include both steps into a workflow definition. For further information, refer to *Defining the Workflow (Section) "Process SAPscript Text"* [Seite 1241].

Procedure at runtime

At runtime the recipient of the work item enters a *SAPscript* text in the first step. Text variables can be used to reference values from the task container of this single-step task. Knowledge of the task container is required to use this variable replacement option.

This text is displayed to the user who executes the work item in the second step.

Printing standard texts

The following methods of the object type `STD_TEXT` enable users to print the standard texts, and can be used in single-step tasks:

- **ReplaceAndPrintBatch** (*Print text with symbol replacement in background*)
  
  The printer and print attributes must be passed as parameters.

- **ReplaceAndPrint** (*Replace text symbols and print text*)
Defining the Standard Task "Create SAPscript Text"


14. Select Standard task in the list field Task and then \(\text{Create}\).

15. Enter the following data:
   
   - **Abbreviation**: Any abbreviation from your namespace [Extern]
   - **Description**: Any task description
   - **Work item text**: Any text to identify the work item in the Business Workplace
   - **Object type**: STD_TEXT
   - **Method**: CREATE

16. Select \(\text{Create}\).

**Agent assignment**

17. Choose Additional data → Agent assignment → Maintain to specify the possible agents [Extern] for this single-step task.

18. Select \(\text{Maintain}\).

**Result**

You have defined a single-step task that enables a SAPscript text to be created when it is executed. This creates an object of the type STD_TEXT (SAPscript standard text). The reference to this object is stored in the container element _WI_Object_ID of the task container after execution of the method. If you use this task as an activity in a workflow definition, you must pass this reference in the binding from the task container to the workflow container. To do this you require a container element in the workflow container with a data type reference to the object type STD_TEXT.

This text is displayed in another single-step task Display standard text. Therefore this single-step task should really only be used as a step in a workflow definition.
Defining the Standard Task "Display SAPscript Text"

Prerequisites
A SAPscript text has been entered and the object reference is known in the workflow.

Procedure
20. Select Standard task in the list field Task and then .
21. Enter the following data:
   
   Abbreviation:  Any abbreviation from your namespace [Extern]
   Description:    Any task description
   Work item text: Any text to identify the work item in the Business Workplace
   Object type:    STD_TEXT
   Method:        REPLACEANDDISPLAY
22. Select .

Agent assignment
23. Choose Additional data → Agent assignment → Maintain to specify the possible agents [Extern] for this single-step task.
24. Select .

Result
You have defined a single-step task that enables a SAPscript text to be displayed when it is executed. The text to be displayed must be an object reference to an object of the type STD_TEXT in the container element _WI_Object_ID of the task container.
Defining the Workflow (Section) "Process Standard Text"

Prerequisites
The single-step task is defined, with which the method Create (object type STD_TEXT) is created. For a description of the procedure, refer to Defining the Standard Task "Create SAPscript Text" [Seite 1239].
The single-step task is defined, with which the method ReplaceAndDisplay (object type STD_TEXT) is executed. For a description of the procedure, refer to Defining the Standard Task "Display SAPscript Text" [Seite 1240].

Procedure
Add the following activities to the workflow definition. The two activities do not need to be consecutive.

Activity for creating a text
3. Define an activity in which you enter the single-step task for the method Create (object type STD_TEXT).
   This displays the dialog box Define container elements and binding, on which proposals for a binding and for new container elements of the workflow container are displayed.
4. Select .
   The system creates the container element STANDARDTEXT in the workflow container, in which the object reference of the SAPscript text created is stored. A binding is defined from the container element _WI_Object_ID in the task container to the container element STANDARDTEXT in the workflow container for this.

Activity for displaying a text
3. Define an activity in which you enter the single-step task for the method ReplaceAndDisplay (object type STD_TEXT).
   This displays the dialog box Define container elements and binding, on which proposals for a binding and for new container elements of the workflow container are displayed.
4. Select .
   The single-step task requires the object reference of the SAPscript text to be displayed.
   The system defines a binding from the container element STANDARDTEXT in the workflow container to the element _WI_Object_Id in the task container.
Specifying Agents of Subsequent Steps Dynamically

Use

The actual agent of the step Determine agents in a workflow can specify the recipients who are to receive the work item of the subsequent step Any activity in the same workflow. The step Any activity does not have to follow the step Determine agents directly.

You can only specify users as recipients if they are possible agents of the relevant task.

Use ad hoc agent assignment only if the user who starts the workflow is to specify recipients at runtime.

Features

You can use the wizard Include "Choose agent" to extend the workflow definition. In this wizard you specify the task for which you want to select recipients. The wizard extends the workflow definition by adding two steps:

- Activity Selection of agents for subsequent task
- Activity that references the subsequent task you specify

Activity Selection of agents for subsequent task

The system enters the workflow initiator as responsible agent for this task. The actual agent of this work item selects the recipients from the possible agents of the subsequent task. A list of the possible agents of the subsequent task is displayed if it is not a general task. If the subsequent task is a general task, the actual agent can select the recipients from all users of the system. The recipients are transported with a binding into a multiline workflow container element created by the system.

Activity that references the subsequent task you specify

The responsible agents of the subsequent task are read by the system from container element created. The graphic below illustrates the extension of the workflow definition and the bindings created.
### Specifying Agents of Subsequent Tasks

<table>
<thead>
<tr>
<th><strong>Task container</strong></th>
<th><strong>Workflow container</strong></th>
</tr>
</thead>
<tbody>
<tr>
<td>Task in workflow</td>
<td>Agent selection _nnnn</td>
</tr>
<tr>
<td>(_WL_Object_Id)</td>
<td>(Agent selection _nnnn)</td>
</tr>
<tr>
<td>Multiple selection</td>
<td>(SelectMultiple)</td>
</tr>
<tr>
<td>Number of task</td>
<td>(Act_Org_Object)</td>
</tr>
<tr>
<td>Object list sort</td>
<td>(SortObjectList)</td>
</tr>
<tr>
<td>Title of dialog box</td>
<td>(Title)</td>
</tr>
<tr>
<td>Selected objects</td>
<td>(SelectedObjectList)</td>
</tr>
<tr>
<td>Selected object</td>
<td>(SelectedObject)</td>
</tr>
</tbody>
</table>

- ID of the subsequent task for which the recipients are being determined, as an object reference
- Select several recipients
- Title of the dialog box with the list of possible agents
- Recipients selected
- Recipient selected

**Activities**

Execute the wizard *Include "Choose agent"*. You have to edit the steps created if:

- The recipients are not to be selected by the workflow initiator
- Deadlines are to be defined for the activities
- Other outcomes are to trigger a response in the workflow
- The work item display is to be formatted individually

You need your own task for the activity *Selection of agents for subsequent task* if, for example, you want to enter your own texts for the task.

For further information, refer to [Copying a Selection Task Seite 1244].
Copying a Selection Task

Use
The wizard uses the standard task TS30200146 as a selection task. If you want to use another task, create it as a copy of the standard task.

The standard task references the object type WF_TASK [Seite 1163].

Procedure
8. Copy the standard task TS30200146.
   For further information, refer to Creating and Maintaining Tasks [Seite 1173].
9. Open the task in change mode and make your changes.
   Select No if the system asks whether missing elements should be taken from the object method.
10. Go to the Workflow Builder and use the wizard to insert dynamic agent assignment into your workflow definition.
11. Open the step definition of the activity Selection of agents for subsequent task created by the wizard.
12. Go to the binding editor and note the binding created by the wizard.
13. Replace the standard task TS30200146 used by the wizard with your copy.
14. Define the same binding for the new task as the one the wizard created for the standard task TS30200146.
Copying Tasks into Another Plan Version

Use
You can copy customer tasks [Extern] and workflow tasks [Extern] to other plan versions.

Procedure

The Copy plan version dialog box appears.
6. Enter the source and target plan version.
7. Enter the task type to be copied.
8. Select 🔄.
Organizational Plan

Definition
Representation of the task-related, functional structure of your enterprise, created using tools from the Organizational Management component. This functional organizational plan differs from the administrative enterprise structure and the personnel structure whose elements are relevant to Payroll Accounting (company code, personnel subarea or employee group, for example). These structures are found in their corresponding components.

Use
You can create several organizational plans in different plan versions, this provides you with the following options in Organizational Management:

- in one plan version, you depict your current valid organizational plan which you use for your current business processes (evaluations, Workflow, personnel planning, for example).
- In additional plan versions, you can depict organizational plans as planning scenarios (for Business Process Re-Engineering, for example).

You can compare the current organizational plan with the planning scenarios and transfer data from the simulated structures into the current organizational plan.

Structure
Organizational plans are normally created by assigning objects of the following types to each other:

- Organizational Unit
- Position
- Job
- Task

If you are using your organizational plan for Workflow, the following object types are also available:

- Standard task
- Workflow template
- Task group
The main elements of an organizational plan are

- an organizational structure [Extern], with which the reporting structure and task distribution are created using organizational units (departments, for example)
- Staff assignments [Extern] for each organizational unit, in which the current persons (employees), users and vacancies are listed

**Integration**

Organizational plans are generally related to objects from other components.

- If Integration with Personnel Administration [Extern] is active, the personal data for the staff assignments comes from the Personnel Administration component.
  
  These functional (organizational plan) and administrative (personnel and enterprise structures) structures come into contact if a person is assigned to an organizational plan (as the holder of a position) as well as an enterprise or personnel structure (that is, to a personnel subarea etc.).

- If the enterprise structure [Extern] is active, account assignment data can be obtained from Controlling.
Working with Screen Areas

Purpose
You use the user interface in the Organization and Staffing or Organization and Staffing (Workflow) view to create, display and edit organizational plans. The user interface is divided into various areas, each of it which fulfills specific functions.

Together, the search area and the selection area make up the Objektmanager [Seite 1257]. In the search area you can choose between different object types. Depending on whether you are in the Organization and Staffing or Organization and Staffing (Workflow) view, you are offered different object types to choose from.

Each time you access the Organization and Staffing or Organization and Staffing (Workflow) views, the objects that you edited last are automatically called up, though (if appropriate) with a new date. You can continue editing the objects directly. In addition, the last user-specific settings relating to the size of the screen, and in some cases the preview period, are available again.

You can reset user-specific settings in the object manager using the report RH_DELETE_OM_USER_SETTINGS. For more information, see the report documentation.

You can reset the last object selection and the preview period using the report RH_DELETE_NF_USER SETTINGS. For more information, see the report documentation.

Note that the reset takes place across all applications.
Prerequisites
Before you work with screen areas in the Organization and Staffing or Organization and Staffing (Workflow) views, you must be familiar with Validity [Seite 1251]. You have decided to use either the Organization and Staffing or the Organization and Staffing (Workflow) view. If necessary, choose the required view over Goto → Change view.

Process Flow
A typical work process could look like the following:

3. In the Search area, find [Seite 1259] one or more objects that you want to display or edit, for example:
   - a complete organizational structure
   - all objects of a certain object type, positions, for example
   - one or more objects of a certain object type, for example, a particular organizational unit

4. The objects found are listed in the Selection area. Select one of these objects
   - by double clicking on it, if you want to display [Seite 1263] the object itself including its environment in the overview area and its characteristics in the detail area
   - by clicking on it, if you want to assign [Seite 1269] it to another object using drag & drop; you can assign a position to an organizational unit, for example

5. In the Overview area
   - the selected object including its environment is displayed.
     According to the object type the environment can be displayed as
     - an organizational structure
     - a report hierarchy
     - staff assignments in the form of a list or a tree structure
     - task assignments
     - organizational assignments
     - job assignments
     - a task hierarchy
     - an agent
     - account assignments
   - you can, if required
     - switch between these representations of the object’s environment using .
     - create [Seite 1266], copy [Seite 1268], assign/move/reposition [Seite 1269], or delimit/delete [Seite 1271] new objects.
   - select another object

6. In the Detail area, characteristics of the selected object are displayed [Seite 1263] on tab pages. You can edit [Seite 1263] the characteristics of this object or add new ones, as required.
Using the tool, you can hide the detail area, so that you can enlarge the overview area. You can display the detail area again using the tool.

In customizing you can define which tab pages should be displayed. For further information, see the Implementation Guide (IMG) under Personnel Management → Organizational Management → Hierarchy Framework.
Validity

Use
An enterprise’s organizational plan is constantly undergoing change. For this reason, *Organizational Management* allows you to edit the organizational structure, staff assignments as well as individual objects according to key dates.

You must be familiar with the concept of validity as it is relevant to every step in the *Organization and Staffing* and *Organization and Staffing (Workflow)* views.

Prerequisites
So that you can use validity completely, make sure

- that you know the period for which you want to find and select objects
- when you are creating objects and object characteristics, that you specify the period in which they are to be valid
- when you are assigning objects and object characteristics that you specify the period in which the assignments are to be valid

Features
You determine the validity of an object or assignment when you create it. You control the display of objects by entering key dates and preview periods. For more information, see *Validity of Objects and Assignments* [Seite 1252].

You determine the validity of object characteristics when you create them. Changes to data are presented in periods. For more information, see *Validity of Object Characteristics* [Seite 1256].
Validity of Objects and Assignments

Use
So that the presentation of your organizational plan can be exact, objects and assignments between objects exist for a specific period of time. You determine the validity period of objects and assignments when you create them, as required.

Features

Key Date and Preview Period
A key date and a preview period are always set in the Organization and Staffing and the Organization and Staffing (Workflow) views.

- Every time you log on, the current date is set as the key date. You can change the key date. Data valid on the date you have selected is displayed.

- When you logon initially, a preview period of 3 months is set, that is, all changes to data that happen in this period are displayed. You can change this preview period. Next time you log on, the preview period which you selected is set.

System Response
When you create an organizational unit or position in the overview area,

- the validity of the object begins with the key date set, this can be moved forward in the detail area

- the assignment begins with the key date set

If you want the validity of the organizational unit, position or assignment to be restricted, you can terminate them. For more information, see Terminating Organizational Units, or Terminating Positions. If you are creating a job or a task, specify the validity date in the dialog box. For more information, see Creating Jobs or Creating Tasks. Task and job assignments begin on the key date set.

If you are finding objects in the search area, the system selects all the objects that exist on the key date and preview period you set, and presents them for selection in the selection area. Objects whose validity begins only in the preview period set, are indicated by a symbol in the selection area. Objects, whose validity ends in the preview period, are indicated by a symbol.

If you have selected an object to be displayed, in the overview area, according to object type

- the relevant organizational structure and the organizational units assigned to it on the key date or in the preview period is displayed.

- the relevant staff assignments and the positions, persons (employees) or users assigned to it on the key date or in the preview period are displayed.

- the relevant staff assignments and the positions, persons (employees) or users from an organizational unit assigned to the selected job on the key date or in the preview period are displayed.

- the relevant task assignment and the objects assigned to it on the key date or in the preview period is displayed.

Objects whose assignment to another object begins only in the preview period set, are indicated by a symbol. Objects whose assignment to another object ends in the preview period, are indicated by a symbol.
Validity of Objects and Assignments

If you assign an object to another object in the **overview area** or **detail area**, the new assignment applies from the key date to the end of the validity of an object. If you have activated *Query time period for organizational changes*, assignments of organizational units and positions are valid in the period you entered in the *Query time period for organizational changes* dialog box.

Assignments can only be made in the detail area in key date mode. Change the mode by choosing.

**Activities**

Before you perform a step, check the key date and preview period that are set. Change the settings as required. For more information, see *Selecting a Date and Preview Period*. Activate/deactivate *Query Time Period for Organizational Changes*. 
Selecting a Date and Preview Period

Use
To display and edit data according to key dates and time periods, select a key date and preview period.

Procedure
9. Choose Settings → Date and Preview period (or )
   The Define Date and Preview Period dialog box appears.
10. Enter the key date in the Start Date field.
11. In the Preview period field, enter the preview period.
   When you confirm your entries with Return, the preview period you selected is displayed.
   Correct your entries as required.
12. Confirm your entries by choosing .

Result
You have set a date and a preview period. You can change these settings at any time.
Activating/Deactivating Query Time Period

Use
The validity of assignments normally begins on the key date you have specified. Alternatively, you can activate a dialog box, in which you can enter any validity date when assigning objects (this does not apply when you are assigning persons (employees) or users to positions).

Procedure
7. Choose Settings → Query time for organizational changes
   The Query time for organizational changes dialog box appears.
8. Select the Query time for organizational changes field, if you want to enter a validity date when you are assigning objects. Otherwise, you can deselect this field.
9. Confirm your entries by choosing ✅.

Result
If you have activated the time period query, a dialog box will appear when you move or assign objects, you can enter the validity of the assignment in this dialog box.
If you have deactivated the time period query, the validity of the assignment is determined according to the key date.
Validity of Object Characteristics

Use
You store the characteristics of an object on tab pages in the detail area. The characteristics of an object can change with time. For this reason, an exact representation of your organizational plan is available according to a specific time period. You determine the validity of object characteristics, according to your requirements, when you create them.

Features

Key Date and Periods
In the Organization and Staffing and the Organization and Staffing (Workflow) views
- period mode is always set, which displays all existing periods
- key date mode is set, this displays the key date and the current period

System Response
Each time data is changed using a tab page in the detail area, a new period is created for this tab page. It is valid from the date in the Valid from field and until the date in the Valid to field. The previous period is automatically terminated on the day before the new period.
A new period is also created, if you make a change outside the detail area, which affects the tab pages, for example, if you create a new position in the overview area.
In key date mode, if more than one period exists for a tab page, this is indicated by a 🚚. This indicated that assignments which you are making are valid alongside the current period. If there is only one period, this is indicated by a 🚙. In this case, the duration of the new assignment includes the duration of the current period.

Activities
Scroll through the existing periods for object characteristics using ⬇️ and ⬆️ or ← and →. Alternatively, you can use 📚 to select a period directly.
When you are changing data, enter the date from when the data is to be valid in the Valid to field, and the date until it is to be valid in the Valid to field.
By selecting 🚚 switch to key date mode, to assign objects in the detail area.
Object Manager

Use
With the object manager you can search for and select objects that you want to display or edit.

Prerequisites
You are familiar with the validity concept of the application. The validity concept determines which objects you can find during a search.

Features
The object manager consists of the search area and the selection area.

17. In the search area are one or more search functions for each object type, for example the Search Term and Structure Search functions. These search functions are marked with . In addition, the object type itself can contain a search function. The object types are marked with the respective object type-specific symbol.

If necessary, you can add more object types and search functions in customizing. You can also change the sequence of the search functions. For further information see the Implementation Guide (IMG) under Personnel Management → Global Settings in Personnel Management → Settings for Object Manager or the IMG for Organizational Management.

18. In the search area you can create search variants, so that you can reuse search criteria you have grouped together, or hits. These search variants are marked with .
19. In the **selection area** the system displays the objects that you searched for and actually found. According to the search function, this can be either a hit list or a structure.

20. You can scroll through search results in the selection area using 🔄 and 🔄.

21. You can completely hide or display the **object manager**, so that the other screen areas get correspondingly bigger or smaller. To do that, choose **Settings → Show Object Manager** or **Hide Object Manager**.

22. With 🔄 you can increase or reduce the size of the selection area, in order to show more hits. As you do that, the search area is hidden or displayed accordingly.

    🔄  

    The system saves the last settings relating to screen size and the last object selection user-specifically, and they are available next time you call up the transaction.

    You can reset these settings and the last object selection using the report **RH_DELETE_OM_USER_SETTINGS**. Note that the reset takes place across all applications. For more information, see the report documentation.

- You can call up **generic object services** for a selected object using the right mouse button menu. To do that, select an object in the selection area with the right mouse button and choose **Generic Object Services**. For further information about generic object services, see [List of Object Services](#).
Finding/Selecting Objects

Use
In the Search area you can search for one or more objects that you want to display or edit. For each object type there are various search functions, for example:
- Search Term
- Structural Search
- Free Search

Prerequisites
Objects you are searching for must already exist. Create new objects as required. The required object types and search functions are set up.

Procedure
3. In the search area select one of the available search functions and if necessary enter the required selection criteria.
   In the selection area the system displays all found objects with the corresponding object type, either as a list or as a structure.
4. Double-click on the required object.
Using Search Tools

Use
With the search tools for each object type you can search for objects in various object type-specific ways. These search tools are marked with \[\text{\textregistered}\]. In addition, the object type itself can contain a search tool. The object types are marked with the respective object type-specific symbol.

Prerequisites
You are familiar with how the search tools are assigned.

23. In the search area, the object type itself can contain a search tool. The object types are marked with \[\text{\textregistered}\].

24. Search tools can be positioned under the corresponding object type. In that case it is marked with \[\text{\textregistered}\].

Procedure

Searching for Objects Using a Search Term
3. Choose \textit{Search Term} (or the required object type, if it has this search function).

The \textit{Search for \textless object type\textgreater} dialog box appears.

8. Enter a name. This can be a name, abbreviation or numeric ID. You can also search using the entry *.

9. Restrict the number of hits, if required. Enter whether the object you are looking for is directly or indirectly assigned to another object.

10. Choose \textit{Search}.

The results of the search are displayed in the \textit{selection area}. The display in the overview and detail areas does not change.

9. If necessary, you can start another search for the same object type and then choose \textit{Insert} to add the new hits to the first results in the selection area.

10. Double-click on the required object in the selection area.

Searching for Objects Using Free Search

The \textit{Free Search} search tool uses the \textit{InfoSet Query}.

5. Choose \textit{Free Search} (or the required object type, if it has this search function).

The \textit{Find Objects of Type \textless Object type\textgreater} dialog box appears. For further information, see \textit{HR in the InfoSet Query [Extern]}. After the search, the system displays the search results in the \textit{selection area}.

6. Double-click on the required object in the selection area.

Searching for Objects Using Structure Search

3. Choose \textit{Structure Search} (or the required object type, if it has this search function).

In the \textit{selection area} the system displays all found objects of the relevant object type in a tree structure, ordered according to their assignment in the organizational plan.
If necessary, you can refresh the hits displayed in the selection area using . This is recommended, for example, if you have created new objects shortly before.

4. Expand the structure until the required object is revealed.

To display unrelated objects, choose .

5. Double-click on the required object in the selection area.
Using Search Variants

Use
In some search functions, for example Search Term, you can restrict the number of hits by using a combination of selection criteria. You can then save such a combination as a search variant so that you can use it again. You can also delete a search variant again, if necessary.

Procedure

Creating Search Variants
3. Use one of the search functions to search for objects.
   The system displays the hits in the selection area.
4. Choose and enter a name. Choose .
   The system saves the search criteria as a search variant and assigns them to the corresponding object type in the search area. The search variant is marked with .
5. Select the search variant you created and choose . You can check your search criteria.

Searching for Objects Using a Search Variant
5. Select a search variant.
   The hits are displayed in the selection area.
6. Double-click on the required object.

Deleting Search Variants
5. Select the search variant that you want to delete.
6. Choose .
   The search variant is deleted.
Displaying/Editing Objects

Use
You can allocate numerous characteristics to the organizational objects in an organizational plan. Select an object in the selection area or overview area so that you can display the characteristics of the object in the detail area and, if required, edit them.

Prerequisites
You are familiar with Validity [Seite 1251]. Select a key date and preview period. It can be that the detail area is hidden. In order to display and edit object characteristics, you may need to redisplay the detail area using .

Procedure

Selecting and Displaying Objects
You can select objects either in the selection area or in the overview area:

- In the overview area, select an object by double-clicking on it.

You can only select an object from the selection area if you have already searched for one or more objects in the search area [Seite 1259].

The object itself is displayed in the overview area. The following display types are possible:

- Organizational Unit
  - Organizational structure: structural display showing the assignment of an organizational unit to other organizational units.
  - Task assignment: structural display showing the assignment of an organizational unit to tasks and activity groups.
  - Report hierarchy: structural display showing the assignment of an organizational unit to a chief position and its holder (person or user).
  - Staff assignments (list): list display showing the assignment of an organizational unit to the subordinate positions and their holders.
  - Staff assignments (structure): structural display showing an organizational structure and the assignment of each organizational unit to the subordinate positions and their holders.

- Position
  - Task assignment: structural display showing the assignment of positions to tasks and activity groups.
  - Report hierarchy: structural display showing the higher- and lower-level assignments of positions.
  - Organizational assignment: structural display showing the assignment of a position to higher-level organizational units.
Displaying/Editing Objects

**Job**
- *Task assignment*: structural display showing the assignment of jobs to tasks and activity groups.
- *Job usage*: structural display showing the assignment of a job to positions.

**Person**
- *Organizational assignment*: structural display showing the assignment of a person over positions to higher-level organizational units.
- *Task assignment*: structural display showing the assignment of persons to tasks and activity groups.
- *Report hierarchy*: structural display showing the higher- and lower-level assignments of positions and their holders.

**User**
- *Organizational assignment*: structural display showing the assignment of a user over positions to higher-level organizational units.
- *Task assignment*: structural display showing the assignment of users to tasks and activity groups.

**Task**, **Standard Task**, **Workflow Template**, and **Task Group**
- *Task hierarchy*: structural display showing the assignment of a task to other tasks and activity groups.
- *Agent*: structural display showing the assignment of a task to other organizational objects.

You can display the exact evaluation path for the current display using  

To switch between displays, select the required object and choose  

To move up a level within a displayed structure, choose  

- In the **overview area**, double-click on an object that is displayed in a structure or list. 

The characteristics of the selected object are displayed in the **detail area**.

**Scrolling**
In the **overview area** you can scroll through the displays using  and  

If you have already made changes to data but have not yet saved them, you can use  and  to undo or recreate the last change.

In the **detail area**, you can use  and  or  and  to scroll through the periods that exist for an object.

**Editing the Characteristics of an Object**
In the **detail area** you can edit the characteristics of the object you selected.

7. Select a tab page.
8. Make the appropriate entries. For more information, see
   - [Editing Organizational Units](#)
   - [Editing Positions](#)
9. Save your entries by choosing [✓].

On tab pages containing characteristics there is a [✓].
Creating Objects

Planstelle bearbeiten [Extern]

Stelle bearbeiten [Extern]

Aufgabe bearbeiten [Extern]

Use

You depict the functional structure of your enterprise as an organizational plan, by creating organizational objects (organizational units, positions etc.) in the overview area and assigning them to objects that already exist.

Prerequisites

To create a root organizational unit for a new organizational plan, you must be in Create mode (Organization and Staffing Create or Organization and Staffing (Workflow) Create).

Create additional organizational objects

• in Create mode, if you are going to create additional objects straight away once you have created the root organizational object, without leaving the mode.

• in Change mode (Organization and Staffing Change or Organization and Staffing (Workflow) Change), if you have left create mode.

You are familiar with Validity [Seite 1251]. Select a key date and preview period.

Which objects you can create depends on which display type is currently in the overview area. If necessary, check which assignments can be displayed in your particular case, using . If necessary, use to switch to a display type in which you can carry out the required assignment.

For more information on the display types, see Displaying/Editing Objects [Seite 1263]

Procedure

Creating Root Organizational Units

You have chosen create mode.

3. In the Create a root organizational unit dialog box, enter the required validity period of the root organizational unit and confirm with .

   The system creates a root organizational unit with the provisional name “New Organizational Unit”.

6. In the detail area, replace the preset entries for object name and object description on the Basic data tab page with the name of your choice.

7. Save your entries by choosing .

   You have created your root organizational unit.

Creating Organizational Objects in the Structural Display

9. In the overview area, select the object to which you want to assign the object being created.

10. Choose .

   The Choose Relationship dialog box appears.
11. Select an object type.
   The system creates the object.

12. Over the tab pages in the **detail area**, edit the characteristics of the new object.
   For further information according to object type, see:
   - Editing Organizational Units [Extern]
   - Editing Positions [Extern]
   - Editing Jobs [Extern]
   - Editing Tasks [Extern]

**Creating Positions in Staff Assignments (List)**

3. Choose 📋.
   The system creates the new position.

4. Over the tab pages in the **detail area**, edit the characteristics of the new object.

You can also create new objects by copying. For more information, see Copying Objects [Seite 1268].

You can always assign objects to other objects – not just when you create them. For more information, see Assigning Objects [Seite 1269].
Copy objects

Use
You can create a new object by copying an object which already exists. When you do this, all the characteristics of the object are copied.

Prerequisites
You are familiar with Validity [Seite 1251]. Select a key date and preview period.

Procedure
5. In the overview area, select the object which you want to copy.
6. Choose:
   
   The Copy Object dialog box appears.
5. Specify the number of copies you wish to make and the validity period which is to apply to the new objects.
6. Choose:
   
   The new object is created. The object's tab pages will be filled with the characteristics of the template. The characteristics can be changed or enhanced as required.
7. Save your entries by choosing.
Assigning/Moving/Repositioning Objects

Use
An organizational plan depicts the functional relationships in your enterprise. To depict such relationships (the hierarchy of organizational units or the staffing of positions by persons, for example), you assign objects to each other. Objects can be
- newly assigned if they are not yet assigned.
- moved, that is, you end the object’s current assignment in a structure and create a new assignment within the same structure.
- repositioned within a hierarchy level in a structure.

The following can be target positions:
- An **organizational unit** can be assigned to objects of the following types:
  - Organizational Unit, Task, Standard Task, Workflow Template, Task Group
- A **position** can be assigned to objects of the following types:
  - Organizational Unit, Task, Standard Task, Workflow Template, Task Group
- A **person (employee)** or **user** can be assigned to objects of the following types:
  - Position, Task, Standard Task, Workflow Template, Task Group
- A **position** can be assigned to objects of the following types:
  - Position, Task, Standard Task, Workflow Template, Task Group
- A **task**, a **standard task**, a **workflow template** or a **task group** can be assigned to objects of the following types:
  - Organizational Unit, Position, Job, Person, User, Task, Standard Task, Workflow Template, Task Group.

Prerequisites
You are familiar with **Validity** [Seite 1251]. Select a key date and preview period.
- You can only assign standard tasks, workflow templates and task groups in the Organization and Staffing (Workflow) view.

Which objects you can assign/move/reposition depends on which display type is currently in the **overview area**. If necessary, check which assignments can be displayed in your particular case, using . If necessary, use to switch to a display type in which you can carry out the required assignment.

For more information on the display types, see **Displaying/Editing Objects** [Seite 1263]

Procedure
Assigning Objects
9. Search for an object that already exists, but does not yet belong to the current structure. For more information, see **Finding Objects** [Seite 1259].

10. In the **selection area**, select the object that you want to assign to another object.
11. Holding down the left mouse button, drag the object to the target position. The target position can be an object in a structural display or a field in the staff assignments (list) display.
   The system assigns the object.
12. Save your entries by choosing \( \square \).

**Moving Objects**
7. In the **overview area**, select the object that you want to move.
8. Hold down the left mouse button and drag the object to the object to which you want to assign it.
   The object has moved and is therefore newly-assigned.
9. Save your entries by choosing \( \square \).

**Repositioning Objects**
7. In the **overview area**, select the object that you want to reposition.
8. Choose \( \leftarrow \) or \( \rightarrow \) to reposition the object in a higher or lower position within the hierarchy level.
   The object has been repositioned. Therefore, the sequence of objects on a hierarchy level has changed.
9. Save your entries by choosing \( \square \).

**Assigning/Moving Using a Pushbutton**
As an alternative to assigning or moving with drag & drop, you can also use the \( \square \) pushbutton.
9. In the **overview area**, select the object that you want to assign to another object.
10. Choose \( \square \).
   The **Restrict allowed values** dialog box appears.
11. Search for the object that you want to assign, select it and then choose \( \square \).
12. Save your entries by choosing \( \square \).
Terminating/Deleting Objects or Assignments

Use
You can:

- Terminate organizational objects if you want to limit their validity.
- Terminate the assignment of an organizational object to another organizational object if you want to limit the validity of this assignment.
- Delete an organizational objects, if you want to delete them completely, including their history.
- Delete the assignment of an organizational object to another organizational object, if you want to delete this assignment and its history completely.

Prerequisites
You are familiar with Validity [Seite 1251]. Select a key date and preview period. Choose key date mode in the detail area.
Which objects/assignments you can terminate/delete depends on what is displayed in the overview area. Check which assignments can be depicted in a particular case by choosing . If necessary, switch to a display in which you can terminate/delete the object or assignment by choosing .

For more information on the displays, see Displaying/Editing Objects [Seite 1263].

Procedure

9. In the overview area, select the object which you want to terminate/delete or whose assignment you wish to delete.

10. Choose if you want to terminate, or if you want to delete.

11. Choose whether you want to terminate/delete the object or the assignment.
   
   If you want to terminate, enter the date from which the object/assignment is to be terminated in the dialog box, and choose .
   
   The object / the assignment is terminated/deleted.

12. Save your entries by choosing .
Undoing/Recreating Changes

Use
If you have already made changes to data in the overview area, but have not yet saved them, you can use and to undo or recreate them.

Prerequisites
You have changed data but you have not yet saved.
A change (no assignment) in the detail area: You have confirmed the change with Return, or you carried out another activity after the change, chosen another tab page, for example.

Procedure
3. Choose to undo the last change.
   The change is undone.
4. Choose to recreate the last change.
   The change is recreated.
5. Save your entries by choosing when you want to finally confirm your changes.
Confirmation Prompt

Use
Once you have carried out a specified number of changes to data (specified in Customizing), a dialog box appears asking whether you want to save these changes. Thus you can avoid losing data.

Prerequisites
You define the number of steps that are carried out before the confirmation prompt appears in Customizing. For further information, see the Implementation Guide (IMG) under Personnel Management → Organizational Management → Framework → Set Up Confirmation Prompt.

Activities
Save data as required.
Configuring Columns

Use
You can decide which columns should be displayed in different screen areas.

Prerequisites
The columns that are available are determined in Customizing. For further information see the Implementation Guide (IMG) under Personnel Management → Global Settings in Personnel Management → Column Framework or the IMG for Organizational Management.

Procedure
7. Choose \(\text{Choose}\).
   The Column configuration dialog box appears.
8. Select the columns that you want to display.
   Some entries represent column groups, that is, more than one column will be displayed if you select one of these entries.
9. Confirm your entries by choosing \(\text{Confirm}\).

Result
The columns you selected are displayed.

You can reset all of a user’s column configurations using the report RH_DELETE_COL_USER_SETTINGS. Note that the reset takes place across all applications. For more information, see the report documentation.
Switching Maintenance Interfaces

Use
You can replace the Organization and Staffing and the Organization and Staffing (Workflow) views with Simple Maintenance. You may want to switch if, for example, you have already used Simple Maintenance and you want to work with the maintenance interface you are used to. For more information on Simple Maintenance, see Simple Maintenance [Extern].
In addition, you can jump to Infotype maintenance from a selected object. For more information, see Infotype Maintenance [Extern].

Procedure

**Situation A:** You are in Organization and Staffing or Organization and Staffing (Workflow) view and want to switch to Simple Maintenance.
Choose Settings → Maintenance Interface. You access Simple Maintenance.
If you do not change this setting, you will automatically access Simple Maintenance when you call up a maintenance interface in the future.

**Situation B:** You want to reverse this setting. You are in Simple Maintenance.
Choose Settings → Maintenance Interface. The Organization and Staffing interface appears.
When you call up a maintenance interface in future, the Organization and Staffing view will appear.

**Situation C:** You are in neither of the two maintenance interfaces; you want to select an interface.
In the SAP menu choose Human resources → Organizational management → Settings → Set maintenance interface. The Set maintenance interface dialog box appears. Select a maintenance interface and confirm by choosing ✅. If you do not change this setting, you will automatically access the interface you select when you call up a maintenance interface in the future.

**Situation D:** You have selected an object in the Organization and Staffing maintenance interface and displayed its characteristics in the detail area. You want to go to the infotype maintenance for this object.
Choose Goto → Detail object → Enhanced object description. To get back to the Organization and Staffing view, choose 🔄.
Role Resolution

Use
Role resolution enables you to restrict the number of possible agents for a work item. Role resolution determines which responsible agents have a property described by a role [Seite 1277]. This improves the ability of SAP Business Workflow [Seite 1711] to get the right task to the right person at the right time.

Integration
The tools used to define roles [Seite 1281] are part of the Organizational Management component. At runtime, the SAP Business Workflow component uses the roles defined here in conjunction with the organizational plan for role resolution to determine agents for work items. (Decision as to which tasks must be assigned to which agents)

Prerequisites
To facilitate role resolution, roles must be defined [Seite 1281].

Features
The system performs role resolution. It is first performed at runtime depending on and using information from the process currently running.

The example refers to the role "orders administrator for customer <customer> as of order amount <order amount>".

If customer "Miller Ltd." and order amount "$34,569.34" are determined for a specific order, the agents are determined at workflow runtime who are the "orders administrator for customer Miller Ltd. as of order amount $34,569.34".

The principle of role resolution is always the same:

- The contents of the role container are read.
- The "regulations" or "rules" resulting from the role type are applied to this data.
- The agents so determined are returned as the role resolution result to an internal table with the structure SWHACTOR. This table contains the agents as Organizational Management objects (user, person, position, job, organizational unit) in the required "mix".

The type of role determines how role resolution is performed exactly. You can determine the type of role when defining a role.
Role

Definition
Object used by the SAP Business Workflow component to determine possible agents for a work item.

Use
You use roles to specify an agent (or agents) for a task if the set of possible agents is too large, or not specific enough. By assigning work items to organizationally suitable employees, responsibilities and authorizations are managed efficiently, and bottlenecks are avoided.

You want to forward Mr. Smith’s notification of absence to his head of department. All heads of department at your enterprise are possible agents for a notification of absence. However, you do not want every head of department to receive Mr. Smith’s notification of absence. At runtime, the role used to determine a manager [Seite 1311] enables you to evaluate assignments (relationships) within an organizational plan. The system uses relationships to determine that Ms. Miller is Mr. Smith’s head of department. The task is forwarded to Ms. Miller.

Further examples:
- Role to Determine Design Office [Seite 1304]
- Role to Determine Organizational Unit of a User [Seite 1316]

The agent for a role does not have to be a user. All of the objects in the Organizational Management component can be agents for a role. The workflow uses the values in the role container to select a subset of possible agents. Role resolution, which is performed at runtime to determine the agent for a workflow step, is therefore an intelligent, efficient, and flexible tool.

Roles as Default Roles for Defining Single-Step Tasks
When defining single-step tasks [Seite 1175], you can specify particular agents or recipients by their role in the following instances:

- Agent for task
- Recipient for completion
- Recipient for missed latest end
- Recipient for missed start
- Recipient for missed end

In this context, reference is made to default roles for a task. Specifying default roles for a single-step task is always optional. If default roles are specified, you may need to define binding from the task container to the role container. Resolution is performed for default roles before the single-step task is executed. (If the single-step task is used as a step in a workflow definition, resolution is only performed for default roles if the workflow definition contains no other information with regard to responsibilities or recipients.)

As a general rule, a single-step task can only be executed by its possible agents (or a subset thereof) when it is processed.
Specifying a role **restricts** these agents to those you have selected. This method cannot be used to authorize new agents to execute a customer task/standard task.

**Roles in Workflow Definition**

When the following steps of a Workflow Definition [Seite 1018] are defined, the agents responsible and the recipients can be specified by their role:

- Activities
- Wait steps
- User decisions

These specifications only have local validity for the respective workflow definition, and they are optional. (Specifying a role is just one of several methods that can be used to specify the agents responsible and the recipients. It is also possible to specify responsibility by using a suitable organizational object (job, position, organizational unit) or by using an expression [Seite 1235] with reference to the workflow container.)

**Structure**

There are various ways of defining roles. You can use the following:

- **Function modules**
  You use a function module to define standard roles if the agent for a task must be found according to extremely complex selection criteria. If you define roles using function modules, the system finds agents by executing the function. How data is obtained varies from function to function. You can use predefined functions, or create your own functions.

- **Organizational data**
  You use organizational data to define standard roles if your business processes are managed on the basis of your organization model. If you define roles using organizational data, role resolution traces the possible agents for a task by using the relationships between the task, the objects in Organizational Management, and the SAP organizational objects.

- **Responsibilities**
  You use responsibilities to define standard roles if you need more precise selection criteria to find agents, but do not want to use function modules. You can also use the organization model to find possible agents using jobs, positions, etc.

All three methods offer certain advantages. However, it is preferable to use responsibilities because you do not require ABAP coding, and can easily display and change agent assignments. You can define as many criteria for a role as required, even if you only want to use some of these criteria for the responsibilities. Each role has a role container [Seite 1280] that includes the values on which role resolution is based. Roles are always defined across clients, and they are always connected to the transport system as cross-client transport objects. At this time, the definition of client-specific roles is not supported.

When saved, each role is assigned an 8-digit number by the system that is preceded by AC, which is used for identification purposes.
Role Container

Use
The role container contains a role’s parameters. Each role has just one role container. At runtime, the role parameters contain the current, context-specific information that forms the basis of role resolution. Therefore, the role parameters constitute “input” for role resolution. The role parameters are provided with values from the workflow container via binding.

Features
Depending on the role resolution procedure, the role container includes the following information:

<table>
<thead>
<tr>
<th>Role resolution procedure</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Based on <strong>responsibilities</strong></td>
<td>The role container includes object references or field values (with names as required) that must be provided with values from the calling component via binding.</td>
</tr>
<tr>
<td>Including <strong>organizational data</strong></td>
<td>The role container only includes the <code>Org_Object_ID</code> element. This element is defined with a reference to the SAP organizational object whose assigned agents must be determined. The role container is created automatically, and is <strong>not</strong> visible during role definition.</td>
</tr>
<tr>
<td>By executing a <strong>function module</strong></td>
<td>The role container includes object references or field values as required (with names as required) that are read by the function module for role resolution and processed accordingly. Prior to role resolution, the container elements are provided with values from the task container via binding.</td>
</tr>
</tbody>
</table>
| Based on **evaluation paths** | The role container includes the following elements:  
  - `OType`, data type reference `OBJEC-OTYPE`  
    Type of object in Organizational Management according to the first step in the evaluation path  
  - `ObjID`, data type reference `OBJEC-REALO`  
    Identification of organizational object.  
  - `Org_Agent`, data type reference `WFSYST-AGENT`  
    C14 field as combination of organizational object type and organizational object.  
    When defining the role, create **either** the first two elements, **or** the last element, as role parameters in the container. |

Activities
You maintain the role container on the **Container** tab page when maintaining the roles.
Role Definition

Use
You use this function if you want to define more roles in addition to the roles delivered by SAP.

Prerequisites
Before you start defining roles, you must start the workflow process.

1. Decide what you want the workflow to achieve, and break it down to the smallest meaningful units of work. These units of work become single-step tasks in your workflow.
2. Select the objects you require. To do so, create a business object or use one of the existing objects. You should be able to find an object that you can use in the Business Object Repository.
3. Each task accesses an object method. Make sure that the methods you require already exist.
4. Create the tasks, or use the existing tasks. Define the possible agents for the task in the task definition.
5. Group the tasks together in the workflow.
6. Assign agents to the work items in the workflow definition. They must be a subset of the possible agents for the task. If this is not the case, the work item is not executed.

Features
When defining a role, you determine

- Which information must be available so that role resolution can be performed when the workflow is executed. This information constitutes the role parameters. They are defined as elements of the role container.
- The rules or regulations in role resolution that are used to determine the appropriate employees.

The role resolution procedure is determined by the role type.

The Responsibilities Role Type
When role resolution is performed, an assignment table is evaluated in which Organizational Management objects (jobs, positions, users, organizational units) are assigned to the various characteristics of the role parameters. This assignment table was explicitly created during role definition.

The Organizational Data Role Type
When role resolution is performed, the system evaluates SAP organizational objects, such as materials controller, planner group, shipping point, or sales office, which are maintained in the master data of an application object.

This type of role resolution requires the use of a separate maintenance transaction independent of role definition to create assignments between the SAP organizational objects and the organizational objects in Organizational Management (jobs, positions, users, organizational units) with which they are related.
For more information, see Define Roles Using Organizational Data [Seite 1289].

**The Function to be Executed Role Type**

When role resolution is performed, a function module is accessed that then facilitates evaluations as required. A table that is maintained in Customizing is evaluated by the function module. The function module must adhere to a given interface, and is specified during role definition.

For more information, see Define Roles Using Function to be Executed [Seite 1305].

**Role Resolution Based on Evaluation Paths**

When role resolution is performed, the system uses information that is available in Organizational Management on the basis of relationships between individual objects that are maintained in an organizational plan.

This information can be used, for example, if you need to determine the head of an organizational unit, or the remaining members of the organizational unit, starting from the initiator of the workflow.

From a technical perspective, this role resolution is very similar to role resolution by executing a function module. The RH_GET_STRUCTURE function module must be used; an evaluation path is also specified.

For more information, see Role Resolution Using Evaluation Paths [Seite 1312].

**Activities**

If you want to use role resolution to restrict the number of possible agents for a work item in a workflow, you must:

6. Choose SAP menu → Tools → Business Workflow → Development → Definition tools → Standard roles to select the tools used to define roles

7. Decide how you want to find agents in the system, that is, using function modules, organizational data, or responsibilities

8. Create a container definition (not for the organizational data role type)

9. Binding is automatically suggested for the workflow. Confirm that the fields in the role container are compatible with the fields in the workflow container.

10. Start the workflow.

At runtime, role resolution provides you with a table containing a set of possible agents. These agents are determined at runtime using the values assigned to the role container elements.
Define Roles Using Responsibilities

Use
A responsibility is an organizational object in which you group criteria together that are required by the workflow at runtime to assign work items to possible agents. This type of criteria definition does not require Customizing settings or ABAP coding.
If you use responsibilities for role definition, you enjoy numerous advantages. You can
- Use criteria that you select as required
- Work with value ranges or individual values
- Display (and change) user assignments
- Group several criteria together in a responsibility

Prerequisites
Before defining roles, you must define workflow steps using the procedure described in SAP Business Workflow [Seite 1711]. You then define the tasks that must be executed. Finally, you use roles to assign and find a list of possible agents for the task at runtime.

Procedure
Responsibilities can only be created for existing standard roles.
To define roles using responsibilities, proceed as follows:
4. Create a container definition [Seite 1285]
   A container is a generic structure that transfers data at runtime. You must create the container definition first because the responsibility depends on the data that you select in the container. You must define a container element for each criterion that you select for the evaluation at runtime.
5. Create a responsibility [Seite 1287], and define criteria (individual values or value ranges) in the responsibility editor for the container elements.
   At this point, you can also process the values and display the container elements.
6. Assign users or organizational objects to the responsibility [Seite 1288]
   If the values in the role container are compatible at runtime with the criteria defined for a responsibility, the organizational objects assigned to this responsibility are identified as possible agents for the work item.

Result
If the values in the role container are compatible with the criteria defined for the responsibilities, SAP Business Workflow identifies the possible agents for a work item. As a result, the right person receives the right work item at the right time. This ensures that business processes at your enterprise run efficiently and on schedule.
Define Roles Using Responsibilities
Creating a Container Definition

Prerequisites
A container definition informs the system about the data type processed by the workflow. You must decide whether you want to create a container definition that references table fields or object types. When the container definition references object types, you use values in the key fields of objects to restrict the list of possible agents. When the container definition references Data Dictionary fields, you can use specific data such as the amount of an order or the customer name on an invoice to restrict the list of possible agents.

To call the transaction for processing roles, access the SAP menu and choose Tools → Business Workflow → Development → Definition tools → Standard roles → Create. The Maintain Standard Role screen appears.

Creating a Container Definition with Table Fields

11. Choose Create.
   The Standard Role: Create screen appears.
12. Enter a name for your standard role in the Basic Data box.
13. Select the role type Responsibilities.
14. Choose Container Definition in the Role Definition box.
   The Standard Role: Process Container screen appears.
15. Choose Create.
   The Create Element dialog box appears.
16. If you want to use Data Dictionary fields, choose Yes.
   The Create with Data Dictionary Field Defaults dialog box appears.
17. Specify the table from which you want to select fields and Choose Continue.
   The table appears.
18. Select the fields that you want to appear in the container definition and confirm your entries by choosing Continue.
   The system prompts you to create the texts for the container elements in the Create dialog box.
   The Element <.....> dialog box appears. If you select the required indicator, an error will occur in the workflow if no binding has been defined for the element. If you select the multiple lines indicator, you can specify multiple values for one container element. You can select both indicators. Choose Continue.
20. Choose Back.
   The Standard Role: Create screen appears.

Creating a Container Definition with Object Types

Carry out steps 1 to 5, and then proceed as follows:
10. If you want to use object types, choose No.

   The Standard Role: Process Container screen appears.

11. Enter a name for the element. If you select the required indicator, an error will occur in the workflow if no binding has been defined for the element. If you select the multiple lines indicator, you can specify multiple values for one container element. You can select both indicators.

12. Select the object type you want to reference, and choose Continue.

   The Standard Role: Process Container screen appears.

13. Choose Back.

   The Standard Role: Create screen appears.

**Result**

You have created a container definition with either Data Dictionary fields or object types as elements. You use these elements to define the criteria for role resolution.
Creating a Responsibility

Prerequisites
Before you create responsibilities, you must Create a Container Definition [Seite 1285]. The values in the role container must either agree with the criteria defined for the responsibility or be in their area. The reason for this is that role resolution compares the two values in order to return a list of possible agents. If you have created a container definition, the Responsibilities button appears on the Add Standard Role screen in the Role definition group box. If you are no longer on the Standard Role Definition screen, then in the SAP Standard Menu, choose Tools → Business Workflow → Development → Definition Tools → Standard Roles → Change and enter the required role. Choose Change.

Creating a Responsibility
On the Standard Role: Create screen:
6. Choose Responsibilities from the Role definition group box.
   The Responsibilities: Change screen appears.
7. Select the superior object (the role you are creating) and choose Create.
   The Create Responsibility screen appears.
8. Enter the name and the validity period of the responsibility and confirm your entries.
   The Change responsibilities for standard role screen appears. This is the responsibility editor.
9. Define values (either single values or value areas) and choose Save.
10. If you do not want to check all criteria for a particular responsibility, leave this line blank.
   The LED display turns yellow, to make you aware that some of the criteria of the standard role will not be checked for this responsibility.

Responsibility Editor
On the Change responsibilities screen, select the responsibility that you wish to edit and choose Change.
The Change responsibilities for standard role screen appears. You can:
- Edit the values for each element
- Create a description for a responsibility
- Display a container definition
- Display details on each element

Result
You have created a responsibility with container elements.
Assigning Agents

Prerequisites
You must first create a container definition [Seite 1285] and then create a responsibility [Seite 1287], in which you define criteria for the container values. You then assign an agent to each responsibility.
If you are no longer on the Standard Role Definition screen, then in the SAP Standard Menu, choose Tools → Business Workflow → Development → Definition Tools → Standard Roles → Change and enter the required role. Choose Change and on the Standard Role: Change screen, choose Goto → Responsibilities.

Procedure
On the Responsibilities: Change screen:

3. Select the responsibility you want to assign to an agent and choose Edit → Agent assignment → Create.
   The Selection dialog box appears.

4. Select the organizational object type you want to assign as an agent and choose Continue.
   The system asks you to enter a search term and to create the relationship between the responsibility and the selected object. Confirm your entries.
   In the Overall View, you can also specify a validity period for the relationship.

Result
At runtime, the workflow evaluates the possible agents and ensures that the right tasks are routed to the right agents at the right time.
Defining Roles Using Organizational Data

Use

A Business Object is often related with an organizational entity (such as MRP controller, laboratory, sales group, purchasing organization, or planner group) by virtue of its master data. From a technical perspective, organizational entities are represented by object types in the Business Object Repository. The indicator organizational type in their basic data defines such object types as SAP Organizational Objects. The attributes of an application object type can be defined with a data type reference to a SAP organizational object type.

These can be evaluated for role resolution purposes to locate the agent of a step. For this to function correctly, you must assign specific agents in the form of positions or organizational units from Organizational Management to the abstract SAP organizational objects. At runtime, these assignments are evaluated in a role resolution if the required input data is available.

The role resolution finds a valid organizational object from Organizational Management as output data.

There are some changes required to be made to material master data.

This task should be routed to employees in the Laboratory/Design Office stored in their material master data. SAP delivers the sample role LABOR from the SAP Organizational Objects for agent determination purposes. For more information, see Role to Determine a Design Office.

Someone should contact the supplier of a certain material.

This task should be routed to employees in the purchasing group stored in the material master data.

Procedure

Relationship Between Organizational Plan and SAP Organizational Objects

You have to set up a relationship between SAP Organizational Objects and the corresponding organizational units or positions in the organizational plan. This step must always be performed because the organizational plan is set up by each customer specific to the enterprise. For more information, see Assign SAP Organizational Objects.

Definition of a Role for Evaluating this Relationship

You define a role according to a predefined schema that can evaluate the above relationship between the organizational plan and SAP Organizational Objects. This step is only required if you cannot use any of the roles delivered in the standard system. For more information, see Defining Roles.

Entering the Role for Specifying Responsibility

You enter the role as agent of an activity or a single-step task and maintain the role container binding. For more information, see Defining Binding.

The single step task Material Master: Maintain Design Data should always be processed by the MRP controller responsible for the material.
Definition time: The role Determine laboratory/design office is specified as the role of the agent of the step Material Master: Maintain Design Data. The binding definition is &Material.Labor& (Workflow Container) -> ORG_OBJECT_ID (Role container).

Run time: The material H4 Lamp, for example, is processed by the task. When the material is known, the agent determined for the work item could be the user that has the role of MRP controller for this material.

Result
At runtime, the workflow determines what organizational objects have the actual values of the key fields of the SAP organizational objects assigned to them. These organizational objects become the agents for the work items.
For more information, see Agent as Attribute of SAP Organizational Object [Seite 1302].
SAP Organizational Object

Definition
Instance of an SAP Organizational object type defined in the Business Object Repository.

Use
You can route tasks to the appropriate users by creating relationships between SAP Organizational Objects – which reside in the Business Object Repository (BOR) – and the Organizational Management objects. The system finds an agent by tracing the relationships between the task, the SAP Organizational Object and the Organizational Management object.

Before you can edit object assignments, you must choose the objects with which you want to work. You can then create relationships between the two kinds of objects. This allows you to use roles to identify potential agents for tasks in SAP Business Workflow. Once you have created these object assignments, you can edit, delete, delimit, and view them.

SAP Organizational object types represent organizational units on the object type level in the Business Object Repository. These units are used to form and describe employee groups.

Examples of organizational units and corresponding SAP organizational object types are:

<table>
<thead>
<tr>
<th>Object Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>BUS0005</td>
<td>Shipping point</td>
</tr>
<tr>
<td>BUS0007</td>
<td>Purchasing organization</td>
</tr>
<tr>
<td>SAP_40132</td>
<td>Work scheduler group</td>
</tr>
<tr>
<td>T024</td>
<td>Purchasing groups</td>
</tr>
<tr>
<td>T024D</td>
<td>MRP controller</td>
</tr>
<tr>
<td>T024L</td>
<td>Laboratory/office for material</td>
</tr>
<tr>
<td>TVKGR</td>
<td>Sales group</td>
</tr>
</tbody>
</table>

What does SAP deliver?
The attribute relationships between Business object types and SAP organizational object types mentioned above are, as a rule, already defined in the Business Object Repository.

Structure
A relationship between an SAP organizational object and a business object type is defined in such a way that the SAP organizational object is available as an attribute of an application object.

For the object type BUS1001 (material), the attribute Laboratory is defined by a data type preference from object type T024L (laboratory/design office for material).

Integration
SAP Organizational Objects must be entered in table T7791, in order for these assignments to Organizational Management objects to be possible. This table is preset in the appropriate format, and you can add new entries.
Assigning SAP Organizational Objects

Use

So that you can define roles using organizational data, you must create and edit assignments between SAP organizational objects and Organizational Management organizational objects.

For example, you want to restrict the task of buying certain materials to certain individuals within a specific organizational unit. This is achieved by creating an assignment between a purchasing group (purchasing groups are SAP Organizational Objects) and an organizational unit.

You can create assignments between any object classified as an SAP Organizational Object, and the Organizational Management objects – organizational units, positions, jobs, and work centers. You apply a validity period to these assignments, so that changes in responsibility can be shown.

Procedure

2. From the SAP menu, choose Tools → Business Workflow → Development → Definition tools → Organizational Management → SAP Org. Objects → Create Assignments.

   The Assignment to SAP Organizational Objects: Initial screen appears.

3. In the Organizational unit and Selection period fields, identify the Organizational Management objects you want to edit.

   Entries in these fields allow you to confine the objects you edit to:
   - A specific area of the organizational plan
   - Objects in the organizational plan that are valid during a specified time frame

   Enter a superior organizational unit if you want to create a relationship between an SAP organizational object and a position. You will subsequently be able to navigate from the selected organizational unit to the object.

4. In the View dialog box, select the SAP Organizational Objects you want to edit. You can choose either:
   - Organizational object type
     You confine your work to specific SAP Organizational Object Types. This reduces the number of steps you perform later, if creating assignments.
   - All organizational object types
     You can work with all types of SAP Organizational Objects. Select an object.

5. You can also create [Seite 1295], delimit [Seite 1298] or delete [Seite 1299] object assignments.

Assignments between SAP organizational objects and organizational objects from Organizational Management can also be created in Infotype 1208 [Extern].
Creating Object Assignments

Prerequisites
You want to create a relationship between an organizational unit or a position and an SAP organizational object, in order to create the link between positions and MRP controllers or organizational units and design offices.

Prerequisites
You are in the Assigning SAP Organizational Objects: Initial screen and have selected the organizational unit and SAP organizational object, which you want to edit. (See Assigning SAP Organizational Objects [Seite 1293])

Procedure
2. Choose Assignment → Change.

Another screen appears, displaying the Organizational Management objects you selected.

The tree structure can display additional information, including existing assignments with SAP Organizational Objects, and other Organizational Management objects in the organizational plan. To adjust the data displayed so that it meets your requirements, choose View.

6. Select the Organizational Management object you want to assign.

7. Choose Assignment → Create.

8. The procedure now varies, depending on the selections made in step 2:

9. If you are working with a specific SAP Organizational Object type, the appropriate dialog box appears, in which you can identify a specific object (for example, a specific purchasing group).

   Make the appropriate selections from the dialog boxes.

If you are working with all SAP Organizational Object types, a series of dialog boxes appears, allowing you to identify the type of SAP Organizational Object type you want to work with, and then a specific object.

   Make the appropriate selections from the dialog boxes.

You want to create a relationship between the organizational unit 50001285 and the SAP organizational object Laboratory, which is described by the key field 002.

To do so, create a new relationship between the organizational unit and the SAP organizational object T024L (Laboratory). A dialog box appears in which you can specify the key field for this SAP organizational object.

Result
The system saves the assignment and displays it in the tree structure.
Function Module

The following function module enables you to set up a relationship between an SAP organizational object and a position or organizational unit:

**RH_SAP_ORG_OBJEC_RELATE**

Assigns an SAP organizational object to an object in Organizational Management

**Interface**

**Export Parameters**

- **ACT_OBJTYPE**, reference field P1208-OBJTYP
- **ACT_OBJKEY**, reference field P1208-OBJKEY

**Exceptions**

- **SAP_OBJECT_KEY_NOT_VALID**

When the function module is accessed, you specify the SAP organizational object type (name is taken from the Business Object Repository) and the object type-specific key that is used to uniquely identify an object of this type.

You assign an organizational unit to this SAP organizational object in the input field. This relationship is transferred to Organizational Management.
Delimiting Object Assignments

Use
You can delimit object assignments to change the validity period applied to the relationship between an Organizational Management object and an SAP Organizational Object, so that the end date occurs sooner than stated.
It can be necessary to delimit the object assignment, for example, if you plan to redirect responsibility for a task at a specific time in the future.

Prerequisites
You are in the Assigning SAP Organizational Objects: Initial screen and have selected the organizational unit and SAP organizational object, which you want to edit. (See Assigning SAP Organizational Objects [Seite 1293])

Procedure
2. Choose Assignment → Change.
   Another screen appears, displaying the Organizational Management objects you selected.
   The tree structure can display additional information, including existing assignments with SAP Organizational Objects, and other Organizational Management objects in the organizational plan. To adjust the data displayed so that it meets your requirements, choose View.
3. Select the assignment that should be delimited:
   c) Locate the two objects in the assignment in the tree structure
   d) Choose the object that is at the lower level of the tree structure
6. Choose Assignment → Delimit.
   A screen appears, displaying the assignment information.
7. In the second Validity field, enter the appropriate end date for the validity period.
8. Choose SAP OrgObjects → Delimit.
   A message appears confirming the system has delimited the assignment.
Deleting Object Assignments

Use
You should only delete assignments between Organizational Management objects and SAP Organizational Objects only if you want to erase all record of an assignment from the database.

⚠️
Deletions should only be necessary if positions have been created incorrectly or by accident. If you want to indicate that responsibilities have changed, use the delimit feature instead.

Prerequisites
You are in the Assigning SAP Organizational Objects: Initial screen and have selected the organizational unit and SAP organizational object, which you want to edit. (See Assigning SAP Organizational Objects [Seite 1293].)

Procedure
2. Choose Assignment → Change.

Another screen appears, displaying the Organizational Management objects you selected.

➡️
The tree structure can display additional information, including existing assignments with SAP Organizational Objects, and other Organizational Management objects in the organizational plan. To adjust the data displayed so that it meets your requirements, choose View.

3. Select the assignment, which you want to delete:
   c) Locate the two objects in the assignment in the tree structure
   d) Choose the object that is at the lower level of the tree structure

4. Choose Assignment → Delete.
   A message appears asking you to confirm that you want to delete.

5. Choose Yes.
   The system deletes the assignment.
Define Role

Prerequisites
You want to define a role whose resolution refers to an SAP organizational object type.

Procedure
5. Create a new role. To call the transaction for processing roles, access the SAP menu and choose Tools → Business Workflow → Development → Definition tools → Standard roles → Create.
6. Select the Organizational data checkbox.
7. Specify an SAP organizational object type.

Result
The role container is created automatically. It includes just one element, Org_Object_ID, in which the object reference to the SAP organizational object is stored.
In this instance, you do not need to specify a function module or define a role container when defining the role.
Define Binding

Prerequisites

You have used your role container to create a role whose resolution is based on the evaluation of organizational data. For more information, see Define Role [Seite 1300]. To "provide" the role container with the object reference to the SAP organizational object, you now define binding for the role container:

- If you use the role for a single-step task, you define binding from the task container.
- If you use the role for a step, you define binding from the workflow container.

Procedure

Assign the following to the role container element: an expression that references the SAP organizational object as an attribute of the application object to be processed.

\[
\text{Org\_Object\_ID} \leftarrow \&<\text{object reference}>.<\text{object reference}>\&
\]

At runtime, the object reference to the processed object of the type material is included in the workflow container under the name Material. You assign the laboratory attribute of this object to the role container in the binding definition. By doing so, you take advantage of the fact that an attribute has been created for the material object type under the name Laboratory that includes an object reference to the design office. You define binding as follows:

<table>
<thead>
<tr>
<th>Org_Object_ID</th>
<th>\leftarrow</th>
<th>&amp;\text{Material}_\text{Labor}&amp;</th>
</tr>
</thead>
<tbody>
<tr>
<td>(Role container)</td>
<td></td>
<td>(Workflow container)</td>
</tr>
</tbody>
</table>
Agent as Attribute of SAP Organizational Object

Use

Organizational entities, which are specified in the master data of a business object, are usually supported by positions or organizational units, which may need to be addressed as the recipient of a work item. The following description illustrates the procedure used to determine these agents on the basis of organizational entities, and how they are indirectly available as attributes of the business object.

SAP Organizational Object Types

Prerequisites

Business Object Type and SAP Organizational Object Type

From a technical perspective, organizational entities are represented by SAP organizational objects [Seite 1291] in the Business Object Repository. The attributes of an application object type can be defined with references to these SAP organizational object types.

SAP Organizational Object Type and the IFSTROBJCT Interface

Each SAP organizational object type (example: design office) should support the IFSTROBJCT interface. As a result, this object type inherits the agent attribute. This attribute returns the position or organizational unit that is related to the SAP organizational object.

Many SAP organizational object types supplied by SAP already support the IFSTROBJCT interface. You only need to enhance a subtype of the corresponding object type with the IFSTROBJCT interface if this is not the case. Programming is not required.
Activities

Relationship Between Organizational Plan and SAP Organizational Objects

Each SAP organizational object must be related to the corresponding position or organizational unit. This step must always be performed because the organizational plan is initially determined for specific customers. For more information, see Assign SAP Organizational Objects [Seite 1293].

Responsibility for a Step

When responsibility is determined for a step as part of a workflow definition, the agent can be derived from the workflow container and specified using a multi-level expression of the following type: &<business object type>.<SAP organizational object type>.agents&.

On the entry screen for Responsibility, select Container and enter &Material.Labor.Agents&.

This procedure means there is no need to define and use a corresponding role.

Error Handling

When this step is performed, the following error situations can arise after the expression has been evaluated:

- The agent that is determined does not belong to the possible agents for the underlying single-step task.

- The relationship between the organizational plan and organizational object (see above) is not maintained, or not maintained in full.

If one of the above error situations arises, the step is instantiated as a work item and addressed to all of the possible agents for the single-step task.

Reference to Related Topics

An alternative concept exists that also makes use of the link between organizational entities and business objects and requires the definition of a role. This concept, which appears initially to be less easy to use, is required, for example, if you need to enter a role as a default role for a single-step task, or if a different method must be used to solve errors.
Role to Determine Design Office

Definition
A role delivered by SAP for determining the employees in a particular design office.

Use
This role enables you to use material to be processed to address the controller responsible for the material. As a result, the object reference to the material to be processed (object type BUS1001) is usually included in the WI_Object_ID element of the task container, and/or in a Material element (or similar) of the workflow container.

This BUS1001 object type has the Laboratory attribute, so that an expression of type & WI_Object_ID.Laboratory& and/or &Material.Laboratory& must be specified as the source of binding for the Org_Object_ID element of the role container.

Structure

<table>
<thead>
<tr>
<th>Role:</th>
<th>30100012</th>
</tr>
</thead>
<tbody>
<tr>
<td>Abbreviation:</td>
<td>Laboratory</td>
</tr>
<tr>
<td>Name:</td>
<td>Determine laboratory/design office</td>
</tr>
</tbody>
</table>

Integration
Individual positions from the organizational plan must be related with the corresponding SAP organizational objects of the T024L Laboratory/design office for material type. These relationships are evaluated for role resolution.
Defining Roles Using Function to Be Executed

Use
You use function modules for role definition when very complex selection criteria are required for agent determination, in other words, when it is not possible to use responsibilities to model agent selection. Function modules provide you with a very powerful tool for determining the agent of a task in Workflow.

Prerequisites
3. You must specify what job you expect the workflow to perform. Depending on this, you decide whether you want to use a standard function module delivered by SAP or your own function module that suits your requirements exactly. To avoid having to display the coding of the function module during the procedure, you should be absolutely sure about what container elements are required by the function module beforehand. If you do need to check the coding of a function module, on the Standard Role: Display screen, choose Goto → Function module.

3. You created your own function module, where relevant. The function module must enable the following process:

d. The role container transferred as the table parameter AC_CONTAINER is read using the macro commands SWC_GET_ELEMENT and SWC_GET_TABLE.

If you want to have the macro commands available, you must integrate the Include <CNTN01> as a sub-report for shared use. <CNTN01> mainly contains the macro command definitions for creating and processing a container instance.

For a complete list of all macro commands, refer to:
Macro Instructions for Processing a Container Instance in a Program [Seite 1133]
Macro Instructions for Accessing Objects, Attributes and Methods [Seite 1146]

e. The role parameters are used to determine the relevant agent. At its simplest, this sub-program consists of a loop on a (Customizing) table, from which the agent is selected.

f. The table ACTOR_TAB is filled.

(See also Example [Seite 1307])

4. You have created a role. To call the transaction for processing roles, access the SAP menu and choose Tools → Business Workflow → Development → Definition tools → Standard roles → Create.

Procedure
On the Standard Role: Create screen:

11. Select Function to be executed.

12. Enter the name of the function module. For example, if you want to use a function module for reporting on the organizational structure, you can enter RH_GET_STRUCTURE.

13. Save the role.

   Depending on what function module you used, the field Evaluation path appears.

14. Enter the relevant evaluation path and choose Save.
15. Choose Container Definition.
   The Standard Role: Process Container screen appears.
17. If you want to use Data Dictionary fields, choose Yes.
   The Create with Data Dictionary Field Defaults dialog box appears.
18. Specify the table from which you want to select fields and Choose Continue.
   The table appears.
19. Select the fields that you want to appear in the container definition. Bear in mind that these are the elements required by the function module.
   The system prompts you to confirm the texts for the container elements.
20. Choose Back.
   The Standard Role:Create screen appears.

**Result**
Workflow executes the function module and, depending on the container data, returns a list of possible agents.
Function Module for Role Resolution

The following excerpts from a fictitious function module for role resolution, which determines the agent responsible on the basis of a release code and object to be released, can be used as an example. The ReleaseCode and ReleaseObject elements are defined in the role container as role parameters.

```
FUNCTION GET_REL_RESPONSIBLE.

*----------------------------------------------------------
*"lokale Schnittstelle:
*"TABLES
*"ACTOR_TAB STRUCTURE SWHACTOR
*"AC_CONTAINER STRUCTURE SWCONT
*"EXCEPTIONS
*"NOBODY_FOUND
*----------------------------------------------------------
INCLUDE <CNTN01>.

* define variables stored in container
DATA: RELEASE_OBJECT TYPE SWC_OBJECT.
DATA: RELEASE_CODE LIKE RM06B-FRGAB.

* local data
DATA: BEGIN OF RELOBJECTKEY,
   NUMBER LIKE EBAN-BANFN,
   POSITION LIKE EBAN-BNFPO,
END OF RELOBJECTKEY.

REFRESH ACTOR_TAB.
CLEAR ACTOR_TAB.

* convert persistent container to runtime container
SWC_CONTAINER_TO_RUNTIME AC_CONTAINER.

* read elements out of container
SWC_GET_ELEMENT AC_CONTAINER 'ReleaseCode' RELEASE_CODE.
SWC_GET_ELEMENT AC_CONTAINER 'ReleaseObject' RELEASE_OBJECT.

* separate object key
SWC_GET_OBJECT_KEY RELEASE_OBJECT RELOBJECTKEY.

* loop and select table <TABLE> with
  *RELEASE_CODE
  *RELOBJECTKEY-NUMBER and RELOBJECTKEY-POSITION
  ....

* end of selection
* exception and parameter handling
```
IF SY-SUBRC NE 0.
  RAISE NOBODY_FOUND.
ELSE.
  ACTOR_TAB-O_TYPE = 'TABLE'-ACTOR_TYPE.
  ACTOR_TAB-O_TYPE = 'TABLE'-ACTOR_TYPE.
  APPEND ACTOR_TAB.
ENDIF.
ENDFUNCTION.
Interface of Function Module for Role Resolution

The interface of a function module for role resolution is described by the following parameters:

**Table Parameters**

**AC_CONTAINER, Reference Structure SWCONT**
Role container with role-specific parameters that must be available as input values for role resolution.

**ACTOR_TAB, Reference Structure SWHACTOR**
Table with results of role resolution as return values.

The **SWHACTOR** structure has the following logical appearance:

<table>
<thead>
<tr>
<th>Field name</th>
<th>Type</th>
<th>Length</th>
<th>Short text</th>
</tr>
</thead>
<tbody>
<tr>
<td>Otype</td>
<td>CHAR</td>
<td>2</td>
<td>Object type in Organizational Management</td>
</tr>
<tr>
<td>ObjID</td>
<td>CHAR</td>
<td>12</td>
<td>ID of object in Organizational Management</td>
</tr>
</tbody>
</table>

The structure consists of two fields:

- **OType** includes a 2-character character field that contains the identification of the object type in Organizational Management. At this time, the following object types are permitted as the result of role resolution:
  
  - O Organizational unit
  - S Position
  - C Job
  - A Work center
  - US User name
  - P Person (PD master data)

  The entries for **OType** are checked against table T779O [Seite 1530].

- **ObjID** includes a 12-character character field that contains the identifying name of a user and/or the unique ID (8-digit number) of an object in Organizational Management.

**Exceptions**

**NOBODY_FOUND**
If the function module for role resolution is exited via the **NOBODY_FOUND** exception, the status of the **Cancellation for role resolution without result for further procedure** indicator is decisive.

- The indicator is set:
  
  The work item and/or workflow from which role resolution was requested is assigned the incorrect status.

- The indicator is not set:
  
  The work item and/or workflow from which role resolution was requested continues. To determine the agent, the possible agents are evaluated.

  The cause of the error, which may have been output as a message when the exception was triggered, is logged in the history of the work item and/or in the workflow step log.

  The message type is not relevant to error handling.
Error Handling for an Empty Table

If the function module for role resolution is not exited via its NOBODY_FOUND exception, and if the ACTOR_TAB table with the agents is still returned empty, the above information applies accordingly.
Role to Determine Manager

Definition
A role delivered by SAP for determining the manager of an agent, position, or organizational unit.

Use
Oftentimes, you will use this role to find the manager of the initiator of a workflow, or the manager of the current agent of a step. The _WF_Initiator element of the workflow container and _WI_Actual.Agent element of the task container are used to store the user names in a 14-character character field in accordance with the ROBJECTS-OBJECT reference. Binding, therefore, must be defined for the Org.Object element of the role container.
As an example, role 00000168 is also used in the example demo for processing a notification of absence. For more information, see Example Demo: Process Notification of Absence [Seite 52].

Structure

| Role:       | 00000168 |
| Abbreviation: | Manager |
| Name:       | Manager of... |

The SWX_GET_MANAGER function module is used to define this role.

Integration
The organizational plan of the enterprise is used for role resolution.
The link between an employee and his or her manager can be depicted in the organizational plan by two different relationships:

- Indication of the chief position for an organizational unit (position "manages" organizational unit, relationship A/B012).
- Direct reporting structure between positions (position "reports to" position, relationship A/B002).
Role Resolution Using Evaluation Paths

Use
The organizational situation of employees within an enterprise is depicted in the organizational plan. Using this existing information, you can perform role resolution starting from one particular employee to determine one other employee, or several other employees, along specific relationships in the organizational plan. The following role evaluates a function module to perform resolution. Please note that this function module is provided by SAP as a default.

Starting from a particular employee, you want to determine his or her organizational unit so that you can address a work item to all of the employees in this organizational unit.

The role that includes this functionality is available in your system and can be used as an explanation of this documentation. For more information, see the role used to determine the organizational unit of a user [Seite 1316].

Integration
From a technical perspective, this role resolution is very similar to Role Resolution Using a Function to be Executed [Seite 1305]. The RH_GET_STRUCTURE function module must be used; an evaluation path is also specified.

Activities
2. Maintain an organizational plan with the appropriate relationships.
3. An evaluation path describes how the relationships between organizational objects are processed in a particular logical order, as required for the role resolution described above.
4. Define a role that evaluates these relationships. This role uses the RH_GET_STRUCTURE function module to perform resolution.
   
   This step is only required if you cannot use any of the roles delivered in the standard system. For more information, see Define Role [Seite 1300].
5. If you enter the role as an agent for an activity or as a default role for a single-step task, maintain binding for the role container. For more information, see Define Binding [Seite 1314].
Define Role to Evaluate Evaluation Paths

Use
You want to define a role whose resolution is based on the evaluation of evaluation paths.

Procedure
8. Create a new role. To call the transaction for processing roles, access the SAP menu and choose Tools → Business Workflow → Development → Definition tools → Standard roles → Create.
5. Select the Function to be executed checkbox.
6. Enter RH_GET_STRUCTURE as the function module for role resolution.
   The named function module is provided by SAP for this purpose. Further programming is not required.
   If this function module is entered as a function module for role resolution, you can specify the evaluation path in an additional dialog box.
7. Define the role container.
   The role container for a role based on the RH_GET_STRUCTURE function module must only contain the following elements:
   Element OType, data type reference OBJEC-OETYPE: type of object in Organizational Management according to the first step in the evaluation path.
   Element Org_Agent, data type reference WFSYST-AGENT: C14 field as combination of object type in Organizational Management and object in Organizational Management.
   Create the first two elements, or the last element, as role parameters.

   The object type in Organizational Management (example: US) and the object in Organizational Management (example: SCHMIDT) can be transferred either to two separate fields or to one field (example: USSCHMIDT). Data specified in one field is evaluated first.

   You define the manager_of role (manager of a user). To do so, you refer to the RH_GET_STRUCTURE function module and specify US_CHEF as the evaluation path.
   You create a new element with the name Org_Agent in the role parameter container. You define this container element as an obligatory element with a data type reference to dictionary table field WFSYST-AGENTS.
Define Binding

Prerequisites
You have created a role based on the RH_GET_STRUCTURE function module using your role container. To "provide" the role container with the agent that represents the starting point of role resolution, you now define binding for the role container:

- If you use the role as a default role for a single-step task, you define binding from the task container.
- If you use the role as a role for a step, you define binding from the workflow container.

The workflow container and task container include system fields that are always available. In the standard system, they are filled by the workflow system. These container elements often include the information required as role parameters and can, therefore, be used as a source for binding. The following container elements include information on users in a C14 field in the <USName> structure.

- The _WF_Initiator element of the workflow container
- The _WI_Actual_Agent element of the task container

Procedure
The procedure is explained using the example of an "approve leave" single-step task. This task must always be completed by the manager of the person who submits the request. Therefore, you want to enter the manager_of role as the default role, and then integrate this customer task as an activity in a workflow definition.

5. Create an element in the task container called applicant with reference to the WFSYST-AGENTS ABAP Dictionary field.
6. Declare the manager_of role as the default role of this customer task.
7. Define the following binding from the task container to the role container:

<table>
<thead>
<tr>
<th>Org_Agent</th>
<th>&amp;Applicant&amp;</th>
</tr>
</thead>
<tbody>
<tr>
<td>(Role container)</td>
<td>(Task container)</td>
</tr>
</tbody>
</table>

8. In the description of the appropriate activity within the workflow definition, define the following binding from the workflow container to the task container:

<table>
<thead>
<tr>
<th>APPLICANT</th>
<th>&amp;_WF_Initiator&amp;</th>
</tr>
</thead>
<tbody>
<tr>
<td>(Task container)</td>
<td>(System variable in workflow container)</td>
</tr>
</tbody>
</table>
Binding Definition for Role Resolution

workflow container

_task container

role container

 WF_INITIATOR

APPLICANT

ORG_AGENT
Role for Determining Organizational Unit of a User

Definition
A role delivered by SAP for determining the organizational unit to which a particular employee belongs.

Use
This role enables you to determine an employee’s organizational unit on the basis of the employee. A distinction is made between the following scenarios:

- The role is used in conjunction with a single-step task, which is classified as a general task [Extern].
  The generated work item can be viewed by all of the employees from the organizational unit that has been determined (and can be processed by a user from this organizational unit).

- The role is used in conjunction with a single-step task whose possible agents [Extern] are specified by one or more positions.
  The work item can only be viewed by employees with a position that belongs to the organizational unit that has been determined and is part of the possible agents.

- The role is used in conjunction with a single-step task whose possible agents [Extern] are specified by a job.
  The work item can only be viewed by employees with a position that belongs to the organizational unit that has been determined and is described by the job.

  The possible agents of a single-step task result from the relationship with the job of secretary. If the described role is used, a work item can always be directed to the secretary of the organizational unit in question.

Structure

<table>
<thead>
<tr>
<th>Role</th>
<th>30000011</th>
</tr>
</thead>
<tbody>
<tr>
<td>Abbreviation</td>
<td>FindOwnOrgUn</td>
</tr>
<tr>
<td>Name</td>
<td>Organizational unit of a user</td>
</tr>
</tbody>
</table>

This role uses the RH_GET_STRUCTURE function module in conjunction with the WF_ORGUN evaluation path (organizational unit of a user/person).

Integration
Each user whose organizational unit must be found must, of course, belong to an organizational unit via his or her position.
The Cancellation for Role Resolution Without Result Indicator

Definition
Indicator that determines how the system reacts if role resolution fails to find a valid agent.

Use
It is possible for role resolution to fail to find a valid agent. This is the case

- If role resolution runs with errors and does not provide any results at all
  (From a technical perspective: the function module for role resolution is exited via its NOBODY_FOUND exception, and/or returns an empty agent table.)

- If role resolution provides agents that do not belong to the possible agents [Extern] of the single-step task.
  (Of course, if recipients or persons responsible for workflow are expressed by specifying a role, the latter cannot occur.)

<table>
<thead>
<tr>
<th>If this indicator is...</th>
<th>Then...</th>
</tr>
</thead>
<tbody>
<tr>
<td>SET</td>
<td>The workflow is cancelled if role resolution fails to find an agent. This protects the data.</td>
</tr>
<tr>
<td>NOT SET</td>
<td>All of the possible agents in the system become actual agents for the work item if role resolution fails to find an agent. The task is completed.</td>
</tr>
</tbody>
</table>

The error situation is recorded in the step log.

Role resolution is used to distribute invoices to various buyers at your enterprise. Only Mr. Miller can execute this work item if the invoice amount exceeds $5,000. If you set the Cancellation for role resolution without result indicator, and if role resolution fails to find an agent for an invoice that exceeds $5,000, the workflow is cancelled. Only the workflow administrator can restart it. If you do not set the Cancellation for role resolution without result indicator, and if role resolution fails to find an agent for an invoice that exceeds $5,000, all buyers can process the invoice.
Selecting Objects from the Organizational Plan

Use
As well as defining roles, it can be necessary to describe certain relationships between employees and areas of responsibility or the borders between areas of responsibility in system tables. This information is then available for role resolution.

Create a table by assigning certain companies and invoice amounts to administrators via their positions.

<table>
<thead>
<tr>
<th>Position</th>
<th>Company</th>
<th>Amount ($)</th>
</tr>
</thead>
<tbody>
<tr>
<td>50001234</td>
<td>A - H</td>
<td>0 - 50.000,-</td>
</tr>
<tr>
<td>50001235</td>
<td>A - H</td>
<td>&gt; 50.000,-</td>
</tr>
<tr>
<td>50001236</td>
<td>I - Z</td>
<td>0 - 60.000,-</td>
</tr>
<tr>
<td>50001237</td>
<td>I - Z</td>
<td>&gt; 60.000,-</td>
</tr>
</tbody>
</table>

If the name of the company and the invoice amount are known, the details in this table are used for role resolution, in order to find an administrator.

The following function modules are available, to ensure that you have access to objects from Organizational Management which you need to fill the table and that you can program possible entries.

Features

**RH_DETERMINE_ORG_OBJECT**
Determines the ID of any Organizational Management object.

**Interface**

**Import Parameter**
- ORG_OBJECT, Reference structure SWHACTOR

**Exceptions**
- NO_ACTIVE_PLVAR
- NO_OBJECT_ID_SELECTED
- NO_OBJECT_TYPE_SELECTED

This function module can be called without transfer parameters.

When you access this function module, as dialog box is displayed from which you can select an object type from Organizational Management.

Once you have selected an object type from Organizational Management, one of the Organizational Management objects stored for this type can be selected and displayed using the search function.

The object type and ID of this Organizational Management object is returned as an export parameter in the SWHACTOR structure and can be used in your application.

**RH_DETERMINE_ORG_OBJID**
Determines the ID of an Organizational Management object from a predefined Organizational Management object type.

**Interface**

**Export parameter**
Selecting Objects from the Organizational Plan

- **ORG_OBJECT_TYPE**, Reference field OBJEC-OTYPE

Import Parameter
- **ORG_OBJECT_OBJID**, Reference field SWHACTOR-OBJID

Exceptions
- **NO_ACTIVE_PLVAR**
- **NO_OBJECT_ID_SELECTED**

This function module forms only the “second half” of the function module described above. When you access it, you transfer the Organizational Management object type whose possible values you want to display using the possible entries function. You transfer this Organizational Management object type as a 1 or 2 character ID.

You can select and display one of the entries, which is stored for the Organizational Management object type using the search function.

The object type and ID of this Organizational Management object is returned as an export parameter in the variable ORG_OBJECT_OBJID and can be used in your application.
Specify Agent, Recipient, or Administrator by His or Her Role

Use
You want to specify an agent, the workflow administrator, or a recipient by his or her role. To do so, you can use the roles you defined yourself or the standard roles provided by SAP.

Procedure
The procedure for entering a role is always the same:
3. To enable you to specify the agent/administrator/recipient as a role, select Role.
4. Enter the unique, 8-digit number of the role in the appropriate input field. If you do not know what this number is, use the input help function. In the standard system, the input help function displays the abbreviation and description of the role. By choosing F17, you can display the plan version and number of the role instead of the description.
4. Define binding for assigning values to the role container:
   When a role is specified in the task definition or workflow definition, the role container must be filled with values from the appropriate container (task container or workflow container) via a binding definition. The binding definition editor is available for this purpose. It enables you to define the appropriate assignments to the elements of the role container.
Event

Definition
Occurrence of a status change in an object.
An event is created from any application program (the event creator) and "published" system-wide. Any number of receivers can respond to the event with their own “response mechanisms”.

Object: Invoice, vendor Miller
Event: Entered

Use
You can use an event:
- As a triggering event [Extern] of a task or a workflow.
  When the event occurs, the task or workflow is started as a response.
- As a terminating event [Extern] of a task.
  When the event occurs, the task is terminated as a response.
- In workflow definition steps of the types event creator [Extern] and wait for event [Extern].

Event receiver linkage at definition time
An event is published without the creating application being informed as to whether a receiver reacts to this event. The system enters potential receivers in a linkage table [Extern], which is evaluated by the event manager [Extern].

Events at runtime
In its event container, each event has information about the context of its creation, which is available to the event receiver. The information can be used for event-driven control and communication mechanisms. The most important event parameter is the reference to the object whose status change the event announces. (The information about the type of status change is part of the name of the event.)

Integration
An event is always defined as a component of an object type [Extern].

The actual creation of an event is not implemented in the program of the object type.

Creation of additional events
You can extend an object type on a customer-specific basis using the delegation concept by adding more events. You must then also cater for the creation [Seite 1322] of these additional events.
Event Creation

Use

Events are created when the relevant status change occurs. The event creation must be implemented by SAP or yourself, or catered for by appropriate table settings. Usually, the event creation is implemented in the application programs for the events supplied by SAP.

Customer-specific events

You can also create additional events yourself for specific status changes, which are not provided by SAP in the standard version. The creation of these events is catered for by table settings.

Features

When an event is triggered, the workflow system checks the event receiver linkage table to see whether receivers are entered for this event. If any are found, these receivers are called using the function module also located in the linkage table.

There are various ways in which events can be created:

<table>
<thead>
<tr>
<th>Feature</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Function module</td>
<td>Event creation by calling function module SWE_EVENT_CREATE or SAP_WAPI_CREATE_EVENT. An event can be created from any program by calling the function module SWE_EVENT_CREATE or SAP_WAPI_CREATE_EVENT. Knowledge and experience of the programming environment are required. The ID of the event, the ID of the triggering object type and the object type-specific key are passed to this function module as import parameters, and the event container as a table parameter. For further information, refer to Creation of Events by Calling a Function Module [Seite 1340].</td>
</tr>
<tr>
<td>Change documents</td>
<td>Event creation when change documents are written. You can set the following system behavior by making entries in the relevant control tables of change document management: Status changes of objects which are logged as change documents are reported automatically as events as well. For further information, refer to Creation of Events When Change Documents Are Written [Seite 1325].</td>
</tr>
<tr>
<td>General status management</td>
<td>Event creation upon status changes. You can set the following system behavior by making entries in the relevant control tables of status management: The system automatically creates an event for objects that use general status management when an object status changes. For further information, refer to Creation of Events Upon Status Changes [Seite 1334].</td>
</tr>
<tr>
<td>Message control</td>
<td>Event creation as message type. Events can be created via a connection to Message Control. For further information, refer to Creation of Events via Message Control [Seite 1337].</td>
</tr>
<tr>
<td>Logistics Information System (LIS)</td>
<td>Event creation when an exception situation occurs (LIS exception). The occurrence of an exception situation defined within the LIS as an exception is to cause an event to be created. For further information, refer to Creation of Events Upon LIS Exceptions [Seite 1338].</td>
</tr>
<tr>
<td>Business Transaction Events</td>
<td>For further information, refer to Creation of Events via Business Transaction Events [Seite 1339].</td>
</tr>
<tr>
<td>HR master data</td>
<td>Event creation when HR master data changes. To create events when HR master records change, you must carry out the following Personnel Management Customizing activities: Activate event linkage [Extern].</td>
</tr>
</tbody>
</table>
Define event types for customer-specific business objects [Extern]

Redefine event types for SAP business objects [Extern]

These Customizing activities can be found in the SAP Reference IMG under Personnel Management → Global Settings in Personnel Management → Business Workflow Events.

Activities

You can use a wizard [Seite 1324] for the creation of certain events. If you cannot use a wizard, you must adhere to the following procedure when creating customer-specific events:

4. Define the event that you want to create as a component of an object type which you have created in the customer namespace.

   Usually you create this object type as a subtype of an object type supplied by SAP. This extends the functionality inherited from the supertype. But you can also create your own object type in the Business Object Repository without using inheritance from a supertype.

   This procedure is necessary because customer-specific events cannot be defined for object types supplied by SAP. These object types have program character and you cannot change them.

   For further information on creating an object type as a subtype, refer to Creating Object Types [Seite 1102].

5. Make this subtype into a delegation type [Extern] of the initial object type.

   For further information, refer to Adapting Object Types on a Customer-Specific Basis [Seite 1159].

6. Maintain the relevant table settings to create your event.

   For further information, refer to Tutorial: Event Creation Upon Status Changes [Seite 1647].
Wizards for Event Creation

Use
You can use these wizards to create events, which can then be used within the definition of workflows.

Integration
The wizards do not cover all aspects of event creation [Seite 1322]. For alternative ways of creating events, choose Tools → Business Workflow → Development → Definition tools → Events → Event creation.

Features
Wizards are available for the following:
- Change documents
- Logistics Information System
- Business Transaction Events
The wizards are mostly self-explanatory and have documentation. To display the relevant documentation, select Click here for further information...

Activities
To execute the individual wizards, choose Tools → Business Workflow → Development → Definition tools → Events → Event creation → Set up with wizard.
Then select the wizard that you want to execute.
Creation of Events When Change Documents are Written

Use
Event creation can be connected to the writing of a change document without modifying an existing application.

Integration
Many business objects are changed frequently. It is often useful and even necessary to be able to trace the changes made. This logging is carried out with change documents. These can be created for changes already made as well as for planned changes.

To be able to log changes to a business object in a change document, an appropriate change document object must be defined in the system. In its definition, a change document object has tables, which represent a business object in the system. Changes to table fields designated as change document-relevant are logged by writing a change document.

Changes made to the material master data in the field MARA-WRKST (basic material) are logged as standard on the part of the application in a change document for the change document object MATERIAL.

For further information about change documents, refer to the documentation Change Documents [Extern].

Features
The assignment between a change document object and a business object type/event must be maintained in a system table. The system then creates the event whenever a change document is written for the change document object. The change document is written when the change is updated. The procedure described, putting the event after the logging, ensures that the event is not created until the change has actually been made.

The system fills the event container of the event created with information about the change document. For further information and advice on how to use these event parameters, refer to Creation of Additional Event Parameters With Event Creation [Seite 1332].

Activities
- Assign a change document object to an object type/event pairing and determine the action (create, change or delete) on the application object for which the event is to be created. You can classify the event in order to create three different events for a change document, if appropriate from a business point of view.

  For further information, refer to Maintaining the Assignment Between Change Document and Event Without Field Restrictions [Seite 1327].

- You may restrict the event creation with regard to whether the change document or change meets certain conditions.

  For further information, refer to Maintaining the Assignment Between Change Document and Event With Field Restrictions [Seite 1329].
You may specify a function module with which you can still perform operations on the event container immediately before the event manager is called, that is, before possible event receivers are determined.

This function module is only required under specific circumstances.
Maintaining the Assignment Between Change Document and Event Without Field Restrictions

Prerequisites
The event that you want to create is defined as a component of an object type.

Procedure
You make the following settings to create an event each time a particular change document is written.


9. In the first view Change "Events for Change Document": Overview, create a new entry. To do this, select the function New entries.

10. Specify a change document object in the field Chng.doc.obj.

11. In the field Business obj. type, enter the object type whose status change is to be indicated by the event created. Use the F4 input help.

The object types offered to you by the system in the F4 input help usually have the same key structure as the change document object.

It may, however, make sense from a business point of view to create the event for an object type whose key structure is different from the key structure of the change document object. In this case, you must first enter a structure for the object type key and a function module in the workflow-relevant settings for change documents. The function module "translates" the key of the change document into the key of the business object. For further information, refer to Maintenance of Workflow-Relevant Settings for Change Documents [Seite 1330].

12. Enter the ID of the event to be created in the field Event. The event must be defined for the specified object type. The name is taken automatically from the object type definition.

13. Select whether the event is to be created upon change, create or delete actions.

Changes to fields in various tables are generally logged with a change document object. Only one of these tables is the main table in the sense that the relevant change document is written with its key. The create, change and delete actions always associated with the main table.

Depending on how the change documents are used in the application, it may be that no change document is written upon create or delete actions.

When an item in a sales and distribution document (structure VBAP) is created, the header data of the document (structure VBAK = main table of the change document object) is changed. Therefore, in this case the change document written logs a change and not a create action.
Maintaining the Assignment Between Change Document and Event Without Field Restrictions

For further information on how you can react to the creation or deletion of non-main tables, refer to Maintaining the Assignment Between Change Document and Event With Field Restrictions [Seite 1329].

14. Save your entries and exit view maintenance.

Result

If you have catered for the creation of an event in the manner described above, you can use this event as a triggering event [Extern] of a task or workflow in the normal way.

If other conditions are to be checked for the linkage between the triggering event and the task to be started, use a check function module [Extern]. You must enter this manually in the type linkage table [Extern].

Event container of created event

In addition to other elements, the event container of the created event contains:

- The reference to the changed object in the element _Evt_Object.
- The reference to the user logged-on in the element _Evt_Creator.
Maintaining the Assignment Between Change Document and Event With Field Restrictions

Prerequisites
To restrict the event creation with regard to whether the change refers to specific fields or fulfills specific conditions, proceed initially as described under [Maintaining the Assignment Between Change Document and Event Without Field Restrictions][1327].

Procedure
4. Select a change document-event assignment and call the dependent view Complex field restrictions.
   
   This displays the screen for changing a condition. You process the condition in the upper half of the screen. An existing condition may already be displayed there.

5. Determine the condition that is to trigger the event.
   
   Refer to the documentation on the condition editor [Seite 1012].

6. Exit the condition editor and choose Save.

   Via the dependent view Field restrictions, you can determine a more simple field restriction (comparison of an "old" value with a "new" value). From there you can also branch to the condition editor via the function expert mode.
Maintenance of Workflow-Relevant Settings for Change Documents

Use

Certain settings must be made for each change document object before it can be used for event creation. This information is required, for example, for correct assignment between change document object and object type.

For all change document objects supplied by SAP, the settings required are made.

- If you want to use the functionality of existing change documents, the entries displayed provide additional information on the existing change document objects.
  
  Do not change these entries!

- If you create your own change document object and want to prepare it for the purposes of flexible event creation, you must also make the relevant specifications in the view maintenance described below.

Features

The following information is stored in the table for each change document object.

Main table in change document

A change document usually refers to several tables. Changes to one of these tables (the "main table" of the change document object) determine the character of the change.

Whether the change document is written when an entry is created, changed or deleted is hence derived from the changes to the main table in the change document.

Change document key with structure

The change document is written with the (key) fields of the table or structure specified in the field Change document key with structure.

Using this, the system can check whether the key fields of the event-triggering object can be derived consistently (= domain same) from the key fields with which the change document is written.

It is also used to set up the F4 input help for entering a business object type when creating events.

Indicator Action: Create, change, delete

Change documents are often written when table entries are changed, and less often when they are created or deleted. To construct a full reference, the actions for which a change document is written are specified. The creation of entries in a table (for example order items), which is not the main table, may lead to a change in the main table (for example order header) and the event only being created if a change action is executed.

Indicator Action: Create instance

Events created from the relevant change document when an object is created can only be used as terminating events if this indicator is set.

Structure

Structure (or table) whose fields define the key composition of an object.

This structure must be specified if the key of the event-triggering object cannot be derived from the key of the change document object, but is instead determined with the function module (see below).
Maintenance of Workflow-Relevant Settings for Change Documents

**Function module**
Function module called by the system with the import parameters *key of change document* and *object type*, and which returns the key of the event-triggering object. This function module is provided by the application for each change document object, if applicable. Its interface corresponds to the function module `SWE_TEMPLATE_OBJECT_KEY_SET`, which can be used as a template.

**Activities**
To maintain the workflow-relevant settings of change documents, choose Tools → Business Workflow → Development → Definition tools → Events → Event creation → Change documents → Define workflow properties.
Creation of Additional Event Parameters With Event Creation

Use
The system adds the old and new values of all changed fields to the event container of the event created. These values can then be read and evaluated in the workflow started by the event in order to influence the process flow of this workflow.

Prerequisites
Writing values into the event container
In order for the system to write the old and new values of a changed field into the event container, this field must be defined as a database field attribute [Extern] for the object type in the Business Object Builder.

Using values in the binding
To enable you to define a binding, with which this information is brought from the event container to the workflow container, the relevant fields must be defined as multiline event parameters [Extern] for the event in the Business Object Builder.

Features
After a change document has been written and the system has recognized from the assignment table that an event of a particular object type is to be created for it, the system checks whether the changed fields are defined as database field attributes of this object type. If this is the case, the changed value and the new value are written under the name of the attribute as a two-line element into the event container. (New value with index 0001, old value with index 0002.)

The values are put into the container as described even if they are not defined as event parameters. The definition as event parameters is required so that you can access it in a binding definition.

Activities
If you want to use the function of additional event parameters with event creation, you must proceed as follows:

- Define an appropriate database field attribute for each relevant field, a change in which can create an event. Ensure that this attribute refers to a table field for which a change is actually logged with the change document. Adopt the name proposed by the system.
  
  For further information, refer to Definition of Attributes for an Object Type [Seite 1123].

- In the same manner, define a multiline event parameter for the event to be created upon the change for each of the fields above. Adopt the name proposed by the system.
  
  For further information, refer to Definition of Events for an Object Type [Seite 1129].

- Define a binding between the event container and the container of the event receiver (usually the workflow container of the workflow to be started).
Creation of Additional Event Parameters With Event Creation

To test what the content of the event container is upon "arrival" at the event receiver, you can send the event container to yourself as a mail. To do this, you must enter a function module as event receiver, which converts the container content into a mail and sends it.

- Add a new entry to the type linkage table for the event receiver linkage.
- Enter the object type and the event that you assigned to the change document object.
- Enter your user name as receiver type.
- Enter SWE_EVENT_MAIL as receiver function module.
- Set the indicator type linkage activated.

For further information, refer to Evaluation and Maintenance of the Type Linkage Table [Seite 1358].

You have created a subtype for the object type BUS1007 (customer). You have added the following components to this subtype:

- Attribute PostalCode with database field reference KNA1-PSTLZ
- Attribute City with database field reference KNA1-ORT01
- Attribute District with database field reference KNA1-ORT02
- Event changed with event parameters PostalCode, City and District (database field references as with attributes).

The container of the event created when a customer’s address changes (relocation from Trier to Cologne) has not only the standard system elements but also the following multiline elements, each of which has a two-line value.

<table>
<thead>
<tr>
<th>Element</th>
<th>Value</th>
<th>Index</th>
</tr>
</thead>
<tbody>
<tr>
<td>PostalCode</td>
<td>50999</td>
<td>0001</td>
</tr>
<tr>
<td></td>
<td>54294</td>
<td>0002</td>
</tr>
<tr>
<td>City</td>
<td>Cologne</td>
<td>0001</td>
</tr>
<tr>
<td></td>
<td>Trier</td>
<td>0002</td>
</tr>
<tr>
<td>District</td>
<td>Rodenkirchen</td>
<td>0001</td>
</tr>
<tr>
<td></td>
<td>Pallien</td>
<td>0002</td>
</tr>
</tbody>
</table>
Creation of Events Upon Status Changes

Use
Event creation can be connected to changes in system status or user status without modifying an existing application.

Integration
General status management makes it possible to document the current processing state of an application object using statuses.
A status is an indicator that can have only the states set (active) or not set (not active). Statuses that can have more than these two states are not supported.
These statuses can be set by the system (system status) and by the user (user status).
- System statuses are indicators that are only set by the system. Their purpose is to document the current state of an object from the system viewpoint.
- User statuses are indicators defined by the user, which document the processing state of the object from the viewpoint of the user.
User statuses are only defined within a status profile. A status profile should be regarded as a group of user statuses.

Activities
If the status change of an object is logged as a change of a user status or system status, you can "report" this in a simple manner using an event. Certain entries must be made in a control table:
- To create an assignment between a status object type and an application object type/event.
- To initiate event creation when the status is either set or reset.
Maintaining the Assignment Between Status Change and Event

Prerequisites
The event that you want to create is defined as a component of an object type. The key fields of this object type must be compatible (= domain same) with the key fields of the status object type whose status change is to cause the creation of an event. The relevant information is located in the system table TBO00, which is maintained and supplied by SAP. This function is not available in a separated Basis system.

Procedure
Make the following settings to create an event if a status change occurs:


10. In the next dialog box, decide whether you want to implement the event creation by connecting to a system status (system settings) or a user status (customer settings). The basic procedure described below is not affected by this decision.

   The screen View: Change "Events for ...Status": Overview is displayed. If you are not in change mode, select Display -> Change.

11. Select New entries.

12. Enter a status object type in the field StatusOT.

13. Only for event creation via user status (customer settings): Enter the status profile for the user status in the field Profile.

14. In the field BusinessOT, enter the object type whose status change is to be indicated by the event created. Use the F4 input help.

15. Enter the ID of the event to be created in the field Event. The event must be defined for the specified object type. The name is taken automatically from the object type definition.

   The event entered is created upon each status change for this status object type.

16. If the event is not to be initiated for every status change but only for activation or deactivation of certain statuses, the view must be maintained for this status event.

   In the view "Status restrictions", you can define the statuses of the status object type, which must be set for the event to be triggered. If you select the field Inactive, the event is triggered when the relevant status is reset.

   If you accessed maintenance of the assignment between status change and event via customer settings, you can enter either a system status or a user status (for the status profile in the first view) in each input line for the status restrictions.

   You can also specify several status restrictions. In this way, you get an "AND" operation for the various restrictions.
You can get "OR" operations by defining several events for a status object type.

**Result**

If you have catered for the creation of an event in the manner described above, you can use this event as a triggering event [Extern] of a workflow or a task in the normal way.

**Event container of created event**

In addition to other elements, the event container of the created event contains:

- The reference to the changed object in the element _Evt_Object.
- The reference to the user logged-on in the element _Evt_Creator.
Creation of Events via Message Control

Use
One of the options for customer-specific creation of events is fully integrated in Message Control. For this, you must create a separate output type and declare 9 (workflow event) as transmission medium.

Prerequisites
This function is not available in a separated Basis system.

Features
Message Control is used for the exchange of information between different partners.

Delivery system
EVEN ("trigger event") has been created as output type.
The program RVNSWE01 with the form routine CREATE_EVENT is the standard processing program for sending the message.
The event ID and the object type that characterize the event created are assigned to a message application in the table NAST.
For further information, refer to the documentation Message Control [Extern].
Creation of Events Upon LIS Exceptions

Use
The processing following an exceptional situation defined as an *exception* recognized by the Early Warning System can be undertaken by a workflow. This *exception* must trigger an event entered as a triggering event of a workflow.

Integration
The Early Warning System is based on the key figures of the Logistics Information System (LIS). The Early Warning System can be used in decisions to make selections and check weak points within Logistics. The Early Warning System enables you to search for exceptional situations, thus helping you to recognize and rectify potential problems at an early stage.

You define the exceptional situations as *exceptions* within the Early Warning System. An *exception* consists of the specification of characteristics and/or characteristic values (such as vendor, material) and conditions. Conditions can be created as threshold values (such as materials/vendors with an order value greater than $5000), as trends (such as a positive trend in the order value) or as planned/actual comparisons.

For further information about the LIS and the Early Warning System, refer to the documentation [Early Warning System: Overview](#)

Prerequisites
This function is not available in a separated Basis system.

Features
The assignment between an *exception* and a business object type/event must be maintained in a system table. Then the system creates the event when an *exception* is recognized.

In connection with the event creation, the system fills the event container with the following information about the exception at runtime:

- The characteristic values when the *exception* was triggered
- The key figures when the *exception* was triggered

To enable you to define a binding, with which this information is brought from the event container to the workflow container, you must define the characteristics and key figures as event parameters for the event in the *Business Object Builder*.

Activities
You make entries in a *control table* to define which *LIS exception* is assigned to which object type/event. You may also specify a function module with which operations can be performed on the event container.

The system calls this function module immediately before scanning the *type linkage table* [Seite 1358] for possible receivers.

For further information, refer to [Wizards for Event Creation](#).
Creation of Events via Business Transaction Events

Use

G/L accounting and accounts receivable/payable accounting provide publish and subscribe interfaces, which inform other application components or external interested parties (partners, customers) that particular events (such as document entered) have taken place in financial accounting. The data this creates can be used by the interested parties for their own subsequent processing. But no data can be returned to G/L accounting or accounts receivable/payable accounting.

SAP Business Workflow uses this interface to create events of object types defined in the Business Object Repository from its events. These events can be used as triggering events of a workflow, for example.

Features

When a Business Transaction Event occurs the system calls a function module which creates the BOR event. To do this, this function module "converts" the data from the Business Transaction Event into the format suitable for the event and creates the appropriate event for the Business Transaction Event. The function module hence has a parameter interface and implementation which are appropriate for the Business Transaction Event.

For every Business Transaction Event that can be linked to an event, SAP provides a suitable function module with appropriate interface and implementation. The following relationships are currently possible:

<table>
<thead>
<tr>
<th>Business Transaction Event</th>
<th>Event</th>
</tr>
</thead>
<tbody>
<tr>
<td>00001030 POST DOCUMENT: Update standard data</td>
<td>BKPF Accounting document CREATED</td>
</tr>
<tr>
<td>00001040 REVERSE CLEARING: After standard update</td>
<td>BKPF Accounting document CLEARINGREVERSED</td>
</tr>
<tr>
<td>00001050 POST DOCUMENT: FI/CO interface</td>
<td>BKPF Accounting document CREATED</td>
</tr>
<tr>
<td>00001110 CHANGE DOCUMENT: Save standard data</td>
<td>BKPF Accounting document CHANGED</td>
</tr>
<tr>
<td>00001320 CUSTOMER MASTER DATA: Save</td>
<td>BUS3007 Customer account CREATED</td>
</tr>
<tr>
<td>00001420 VENDOR MASTER DATA: Save</td>
<td>BUS3008 Vendor account CREATED</td>
</tr>
<tr>
<td>00001520 CREDIT MANAGEMENT: Save</td>
<td>BUS1010 Customer credit account CREATED</td>
</tr>
</tbody>
</table>

Activities

The Business Transaction Events and the events are already defined. You may create the linkage between them. You can use a wizard for event creation. Finally, you must set the status of the event whose creation you have just configured to released.

To do this, call the Business Object Builder and change the release status of the event for the relevant object type.
Creation of Events by Calling a Function Module

Use
You must ensure that the event is triggered when the status change occurs. It may be necessary to trigger the event using a function module in your program.

Prerequisites
Note that events say something about object status changes that have actually occurred. Therefore ensure that the event is not created until the relevant status change has taken place. For this, the function module for creating an event should be called in the same logical unit of work (LUW) as the one in which the status change is made.

Features
You can create any event from any application or system program by calling the relevant function module:

This function module is called `SWE_EVENT_CREATE`.
For the following special cases, there are other function modules, which actually use the functionality of the above function module internally:

- `SWE_EVENT_CREATE_IN_UPD_TASK`
  This function module makes it possible to create events in an update task. In contrast to the function module `SWE_EVENT_CREATE`, it can be called with the addition of `IN UPDATE TASK`.

- `SWE_EVENT_CREATE_FOR_UPD_TASK`
  The event is created in the update task. (The function module is not called with the addition of `IN UPDATE TASK`.)
  In the case of methods for creating an object, the requester used for linking instance-related events is transported into the update session so that an event can be used as a terminating event in the workflow as well. (The particular problem here is that when the instance linkage for the terminating event is entered, the object instance is still unknown.)

When the function module `SWE_EVENT_CREATE` is called, the following operations are performed synchronously:

- **Find requester**
  If an event is created by an application that is executed as an asynchronous object method within a workflow, the work item that called this method can be established via internal queries.

- **Determine all supertypes of the triggering object type**
  The type linkage is not only taken into account for the triggering object type itself, but also for all supertypes of this object type.
  
    A type linkage is entered for an event of the object type `person`.
    If this event is created by the subtype `applicant`, the type linkage of the `person` should also be evaluated.

The evaluation of the linkage tables taking into account the import parameters of the function module for determining a possible receiver follows.
Creation of Events by Calling a Function Module

It is not possible to state whether the receivers were called successfully.

For further information, refer to Evaluation and Maintenance of the Type Linkage Table [Seite 1358].

Activities

Call the function module SWE_EVENT_CREATE or one of its modified versions mentioned above in a program of your application. The function module has the following interface (selection):

<table>
<thead>
<tr>
<th>Import parameters</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>OBJTYPE</td>
<td>SWETYECOCU-OBJTYPE</td>
</tr>
<tr>
<td>OBJKEY</td>
<td>SWEINSTCOU-OBJKEY</td>
</tr>
<tr>
<td>EVENT</td>
<td>SWETYECOCU-EVENT</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Export parameters</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>EVENT_ID</td>
<td>SWEDUMEVID-EVTID</td>
</tr>
</tbody>
</table>

It is not possible to state whether the receivers were called successfully.

If no receiver could be established, zero is returned as the event number.

<table>
<thead>
<tr>
<th>Table parameters</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>EVENT_CONTAINER</td>
<td>SWCONT</td>
</tr>
</tbody>
</table>

Since the asynchronous RFC for calling the receiver function module is not triggered until after the next COMMIT WORK, you must initiate the command COMMIT WORK in your application after the function module for creating an event is called in order for the events to actually be created.

The database commit performed automatically with a screen change does not trigger the asynchronous RFC.
Event parameters

If the event that you want to create has other event parameters defined in addition to the event parameters defined by the system, you must carry out the following before the function module is called:

4. Declare the event container  
5. Initialize the event container  
6. Assign values to the event container.

Macro instructions are available for carrying out these steps. For further information, refer to Macro Instructions for Processing a Container Instance [Seite 1133].

The container is then passed as a table parameter to the function module SWE_EVENT_CREATE. The name you give to the event container in your application is up to you.
Definition of Event Container

Use
The event container contains the **event parameters** as container elements. If you define **triggering** events or **terminating** events for a task, you can specify the binding from the event container to the task container. If you define triggering events for a workflow, you can specify the binding from the event container to the workflow container.

Features
The event container contains **workflow system elements** as standard. You can add more container elements to the event container within object type definition in the *Business Object Builder*. This is not usually necessary, however.
Event Receiver

Definition
Program in which an event is evaluated. From a technical point of view, the event receiver is started by an asynchronous RFC call of the receiver function module in the event receiver linkage table.

Use
Any application that wants to react to an event must provide a receiver function module and make an appropriate entry in the linkage table.

If you use events in the environment of SAP Business Workflow, special components in the workflow system are the receivers of the event.

You do not then need to concern yourself with the programming of the receiver function module and the entry in the linkage table.

Integration
The receiver function module is selected and called by the event manager once it has evaluated the type linkage table.
Receiver Function Module

Definition
Function module with defined interface for receiving events. The event manager calling the receiver function module initiates the event receiver’s reaction to the event.

Use
The container passed in the interface of the function module is read with macro instructions provided. If object references were passed in the container, they must first be converted into runtime handles. For further information and a full list of all macro instructions, refer to:
- Macro Instructions for Processing a Container Instance in a Program [Seite 1133]
- Macro Instructions for Accessing Objects, Attributes, and Methods [Seite 1146]

The workflow system uses the following function modules as standard:

<table>
<thead>
<tr>
<th>Function module</th>
<th>Use</th>
</tr>
</thead>
<tbody>
<tr>
<td>SWW_WI_CREATE_VIA_EVENT</td>
<td>Is entered for triggering events.</td>
</tr>
<tr>
<td>SWW_WI_COMP_EVENT_RECEIVE</td>
<td>Is entered for terminating events.</td>
</tr>
<tr>
<td>SWW_EI_EVENT_RECEIVE</td>
<td>Is entered for wait steps.</td>
</tr>
</tbody>
</table>

Structure
The interface of the receiver function module is described with the following parameters:

<table>
<thead>
<tr>
<th>Import parameters</th>
<th>Type of the triggering object.</th>
</tr>
</thead>
<tbody>
<tr>
<td>OBJTYPE</td>
<td>OBJTYPE</td>
</tr>
<tr>
<td>OBJKEY</td>
<td>SWEINSTCOU-OBJKEY</td>
</tr>
<tr>
<td>EVENT</td>
<td>SWEINSTCOU-EVENT</td>
</tr>
<tr>
<td>RECTYPE</td>
<td>SWETYPECOU-RECTYPE</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Table parameters</th>
<th>Persistent event container of the event.</th>
</tr>
</thead>
<tbody>
<tr>
<td>EVENT_CONTAINER</td>
<td>SWCONT</td>
</tr>
</tbody>
</table>

There is a template function module for receiver function modules
(SWE_CD_TEMPLATE_REC_FB) in the function group SWE_TEMPLATE.

Integration
The receiver function module must be provided by the potential event receiver. The interface of this function module is standard and predefined.

The receiver function module expects the ID of the event, the event container and the name of the receiver type as input parameters. Return parameters are not passed by the receiver function module.
Return parameters and exceptions cannot be passed by the receiver function module. In addition, it cannot initiate exceptions (error messages and warnings) which lead to the function module being aborted.

The receiver function module must be flagged as RFC-enabled.
Check Function Module

Use

If the check function module [Extern] terminates with an exception, the receiver function module is not called and the linkage between the event and the receiver therefore not established.

Using a check function module gives the event receiver the opportunity to decide whether the receiver should actually be called for general or non-specific events before the call actually takes place.

The event Notification created should only actually lead to the receiver being called, if the notification is a service notification.

A check function module exception has no effect on the event creator.

The container passed in the interface of the function module is read with macro instructions provided. If object references were passed in the container, they must first be converted into runtime handles.

For further information and a full list of all macro instructions, refer to:

- Macro Instructions for Processing a Container Instance in a Program [Seite 1133]
- Macro Instructions for Accessing Objects, Attributes, and Methods [Seite 1146]

Integration

Creator and Receiver Context

The check function module must be provided by the potential event receiver, if applicable.

The check function module has the same interface as the receiver function module. This allows the contents of the event container to be checked.

Features

The interface of the check function module is described with the following parameters:
### Import parameters

<table>
<thead>
<tr>
<th>OBJTYPE</th>
<th>SWETYPECOU-OBJTYPE</th>
<th>Type of triggering object</th>
</tr>
</thead>
<tbody>
<tr>
<td>OBJKEY</td>
<td>SWEINSTCOU-OBJKEY</td>
<td>Concatenated, object type-specific key of the triggering object</td>
</tr>
<tr>
<td></td>
<td></td>
<td>The reference to the triggering object is created internally from this information and written to the event container under the element ID _Evt_Object.</td>
</tr>
<tr>
<td>EVENT</td>
<td>SWEINSTCOU-EVENT</td>
<td>ID of event</td>
</tr>
<tr>
<td></td>
<td></td>
<td>The event must be defined for the triggering object type.</td>
</tr>
<tr>
<td>RECTYPE</td>
<td>SWETYPECOU-RECTYPE</td>
<td>Receiver type.</td>
</tr>
</tbody>
</table>

### Table parameters

| EVENT_CONTAINER | SWCONT | Persistent event container of the event. |

### Exception

If the check function module terminates without an exception, the receiver function module is called by the event manager.

If the check function module terminates with any exception, the receiver function module is not called.

Exceptions are not treated in different ways.

> There is a template function module for check function modules (SWE_CD_TEMPLATE_CHECK_FB) in the function group SWE_TEMPLATE.
Receiver Type

Definition
Additional input parameter for the receiver function module, which determines the type of object that is manipulated (in the broadest sense) by the event receiver.

Use
The receiver type, like the receiver function module, is entered in the type linkage table by the event receiver.
A value does not have to be entered in the type linkage table for the receiver type. Accordingly, the receiver type is an optional import parameter of the receiver function module.

Integration
The event receiver can provide a generic receiver function module that then branches with the input parameter receiver type.
The receiver type, like the receiver function module, is entered in the type linkage table by the event receiver. Specification of a receiver type is optional.
Receiver Instance

Definition
Instance of an event receiver.

Use
The ID of a receiver instance (receiver ID) is entered in the instance linkage table at runtime by the potential event receiver.
When the instance linkage is implemented by the event manager at runtime, the event manager writes the receiver ID to the event container and passes it to the receiver function module declared in the type linkage table for further evaluation.

In the workflow environment (wait steps, terminating events), the number of a workflow is generally used as a receiver instance at runtime.
The work item manager [Extern] makes the entry in the instance linkage table.
Receiver Type Function Module

Definition
Function module for establishing the receiver type.

Use
It is possible that there is to be a receiver type, but that it cannot be established until runtime from the information accompanying the event. It is therefore possible to call a receiver type function module to evaluate the event container and determine the receiver type.

A workflow is to be started in response to an event.

The receiver function module is the generic function module for starting workflows, which expects the ID of the workflow task to be started as an input parameter in the receiver type.

The ID may not be known when the linkage is entered. This may be the case if the ID of the workflow task to be started depends on information from the event-creating application (for example an amount).

The relevant information is located in the event container. The ID of the workflow task is then determined by the receiver type function module evaluating the event container.

The container passed in the interface of the function module is read with macro instructions provided. If object references were passed in the container, they must first be converted into runtime handles.

For further information and a full list of all macro instructions, refer to:
- Macro Instructions for Processing a Container Instance in a Program [Seite 1133].
- Macro Instructions for Accessing Objects, Attributes, and Methods [Seite 1146]

Structure
The interface of the receiver type function module is described with the following parameters:

<table>
<thead>
<tr>
<th>Import parameters</th>
<th>Export parameters</th>
</tr>
</thead>
<tbody>
<tr>
<td>OBJTYPE</td>
<td>OBJTYPE</td>
</tr>
<tr>
<td>OBJKEY</td>
<td>OBJKEY</td>
</tr>
<tr>
<td>EVENT</td>
<td>GENERIC_RECTYPE</td>
</tr>
<tr>
<td>GENERIC_RECTYPE</td>
<td>GENERIC_RECTYPE</td>
</tr>
<tr>
<td>RECTYPE</td>
<td>RECTYPE</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>OBJTYPE</th>
<th>OBJKEY</th>
<th>EVENT</th>
<th>GENERIC_RECTYPE</th>
<th>RECTYPE</th>
</tr>
</thead>
<tbody>
<tr>
<td>OBJTYPE</td>
<td>OBJTYPE</td>
<td>OBJKEY</td>
<td>OBJKEY</td>
<td>OBJKEY</td>
</tr>
<tr>
<td>EVENT</td>
<td>EVENT</td>
<td>EVENT</td>
<td>EVENT</td>
<td>EVENT</td>
</tr>
<tr>
<td>GENERIC_RECTYPE</td>
<td>GENERIC_RECTYPE</td>
<td>GENERIC_RECTYPE</td>
<td>GENERIC_RECTYPE</td>
<td>GENERIC_RECTYPE</td>
</tr>
<tr>
<td>RECTYPE</td>
<td>RECTYPE</td>
<td>RECTYPE</td>
<td>RECTYPE</td>
<td>RECTYPE</td>
</tr>
</tbody>
</table>

Type of triggering object
Concatenated, object type-specific key of the triggering object
The reference to the triggering object is created internally from this information and written to the event container under the element ID Evt_Object.
ID of event
The event must be defined for the triggering object type.
Receiver type.
The receiver type, if entered, is passed to the receiver type function module. Its value, however, is overwritten by the export parameter RECTYPE of the receiver type function module.
Receiver Type Function Module

<table>
<thead>
<tr>
<th></th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>RECTYPE</td>
<td>Result of the evaluation by the receiver type function module.</td>
</tr>
</tbody>
</table>

**Table parameters**

<table>
<thead>
<tr>
<th>Event Container</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>EVENT_CONTAINER</td>
<td>SWCONT</td>
</tr>
<tr>
<td>Persistent event container of the event.</td>
<td></td>
</tr>
</tbody>
</table>

There is a template function module for receiver type function modules (SWE_CD_TEMPLATE_RECTYPE_FB) in the function group SWE_TEMPLATE.

**Integration**

The evaluation of the event container with the aim of establishing the receiver type is performed in the receiver type function module. This function module is called provided it is entered in the type linkage table. It returns the receiver type established. The check and receiver function modules are then called with this receiver type.
Maintenance of Linkage Tables

Use
The linkage tables contain the assignment of events to the event receivers interested in the events. The entries in the linkage tables must be made by the potential receiver of an event. This can be done using function modules provided or a maintenance transaction.

⚠️ It is only necessary to maintain linkage tables manually if you use events within separate developments. If events are used exclusively within the SAP Business Workflow environment, all necessary entries (including transport) are made by the workflow system.

Integration
A distinction is made between type linkage and instance linkage. They are contained in separate tables.

Features
If there are appropriate entries in the linkage tables when the event is created, the event manager creates the linkage by calling the receiver function module.

If there are no appropriate entries in the linkage tables when the event is created, the event has no effect. Events are not buffered so that they can be produced at a later point in time.

🚀 It is, however, possible to temporarily store events for which a linkage is entered in the event queue.

The linkage between the creator and the receiver can be activated and deactivated by making a selection in the table (indicator Type linkage activated) at definition time and even at runtime for a specific situation or customer.

Activities
The maintenance transactions for linkage tables can be accessed by choosing Tools → Business Workflow → Development → Administration → Event manager → ...
Evaluation and Maintenance of Instance Linkage Table

Use
The instance linkage describes the assignment of a *receiver instance* to a particular combination of *object type*, *object key* and *event*. This linkage is entered into the instance linkage table by the receiver instance itself using a function module provided.

The entries in the instance linkage table are always made at event receiver runtime.

- If events are used in *SAP Business Workflow* (as triggering and terminating events, for wait steps), the entries required in the linkage tables are made by the workflow system.

- An entry is made in the instance linkage table by the workflow system if work items wait for a terminating event. This can apply to the following work item types:
  - Dialog work items (type W) and background work items (type B) if an asynchronous object method is executed.
  - Wait step work items (type E)
  - Work queue work items (type A)

Integration
The instance linkage enhances the type linkage such that the reaction to the event is only produced if the event is derived from a particular object. This object is specified in the instance linkage table using its object key.

The unique ID (receiver ID) of a *receiver instance* can also be specified in the instance linkage table. This receiver ID is then written to the event container by the event manager and forwarded to the event receiver when the receiver function module is called.
Evaluation and Maintenance of Instance Linkage Table

Instance Linkage

The reaction to an event is made dependent on a particular object, as an instance of an object type

The instance linkage table is only evaluated by the event manager following the evaluation of the type linkage table if the linkage is not designated a global event linkage in the type linkage table. The process flow is as follows:

5. The event manager checks whether the object type, object key and event in the instance linkage table are in accordance with the event parameters of the created event.

   If this is the case, the ID of the receiver instance (receiver ID) is entered into the event container from the instance linkage table.

6. The check function module is called if entered in the type linkage table.

7. If this function module terminates without an exception (or if no check function module is specified), the receiver function module in the type linkage table is called with the transfer parameters event, receiver type and event container.

8. If it was possible to establish the instance linkage, the entry is deleted from the instance linkage table. The entry in the type linkage table remains.

The specifications made in the type linkage table at definition time are used again for the actual assignment of the event to the receiver. This double construction is necessary since the generic selection of the receiver function module must be defined before runtime, but the instance values are not yet known at definition time.

The linkage logic does not allow an actual receiver instance to wait for an event whose triggering object is not yet known. The key of the triggering object must also always be known for an instance linkage.

This problem occurs when an object is created in a work item by an asynchronous method and this is reported to the workflow system by an event. The work item, as an actual receiver instance, is to wait for an event whose triggering object is not yet known when the work item is entered in the instance linkage table.
For this event, there is the workflow requester via which a relationship can be created between the unknown object and the waiting work item.

**Features**

**Structure of the instance linkage table**

The following fields are available in the instance linkage table:

- **Object type**
  Type of the object that creates the event.

- **Event**
  Event created.

Object type and event must be defined in the *Business Object Repository*.

- **Receiver type [Extern]**

- **Object key**
  Concatenated key of the object that creates the event.

- **Receiver instance**
  ID or instance of a receiver type.

The instance linkages (with associated type linkages) required for the wait steps and the terminating events of an activity in the workflow definition are entered automatically. Do not change these entries!

The following are entered in the type linkage table:

- **Object type and event ID**

- **WORKITEM** (for terminating events) or **EVENTITEM** (for wait steps) as **receiver type**

- **Function module** `SWW_WI_COMP_EVENT_RECEIVE` (for terminating events) or `SWW_EI_EVENT_RECEIVE` (for wait steps) as **receiver function module**

- **Indicator global** is not set

- **Indicator enabled** is set

The following are entered in the instance linkage table:

- **Object type and event ID**

- **Key of the object processed in the task**

- **Number of the work item to be terminated or waiting as ID of receiver instance**

If it was possible to establish the instance linkage, the entry is deleted from the instance linkage table. The entry in the type linkage table remains.

**Evaluation of the instance linkage table**

For the purpose of troubleshooting and error analysis, it may be useful to display the instance linkages [Extern] entered. To do this, you can display a section from the instance linkage table according to certain selection criteria.
Evaluation and Maintenance of Instance Linkage Table

Maintenance of the instance linkage table
You maintain the instance linkage table if you use event control within your own developments outside workflow.
The instance linkage table can be maintained using the following function modules:

- **SWE_EVENT_REC_INST_ENTER**
  You can insert entries into the instance linkage table using the parameters of this function module.

- **SWE_EVENT_REC_INST_DELETE**
  You can delete an entry from the instance linkage table using this function module.

- **SWE_EVENT_REC_INST_DELETE_ID**
  You can delete all entries for a receiver from the instance linkage table using this function module.

If you make entries in the instance linkage table, ensure you delete them again when an instance linkage is no longer expected.
You should also delete the associated entries in the type linkage table. Ensure that there are no (more) instance linkages for this type linkage.

Activities
To maintain the type linkage table, choose **Tools → Business Workflow → Development → Utilities → Events → Instance linkages.**
Evaluation and Maintenance of Type Linkage Table

Use
The type linkage describes the assignment of a receiver function module [Extern] and a receiver type [Extern] to a particular combination of object type and event. This linkage is entered into the type linkage table.

The table must be completed if a response is always to be made to an event of a particular object type, even if the linkage is to be instance-based. The type linkage table is evaluated at runtime by the event manager. The event receiver should make the entry in the type linkage table using a function module provided.

If events are used in SAP Business Workflow (as triggering and terminating events, for wait steps), the entries required in the linkage tables are made by the workflow system.

If you enter an event as a triggering event of a task, for example, the following entries are made automatically:

- Object type and event ID
- Task to be started as receiver type
- Function module SWW_WI_CREATE_VIA_EVENT as receiver function module
- Indicator global set in background

When the linkage is activated within task definition, the indicator enabled is also set in the type linkage table.

Integration

Transport and client copy
Entries in the client-specific type linkage tables are written automatically into a Customizing transport request if the client is configured (in table T000) for changes to be recorded automatically. All entries are then transported including activation indicators.

In the case of client copy, you should note that the event receiver linkages are copied into the target client, but are always deactivated in the target client.

The activation indicator of each individual event receiver linkage is only copied with client copy if explicitly requested (parameter option for copying tables of class A).

Process Flow
The event manager begins the evaluation of the type linkage table when it is notified of the ID of a created event. For event handling to take place, either the event created and its triggering object type or the created event and a supertype of the triggering object type must be entered in the type linkage table.

Only those entries in the type linkage table for which the indicator Event linkage activated is set are included by the event manager in the evaluation.
Type Linkage

An appropriate workflow is always started in response to an event of a particular object type.

If the event manager finds an entry in the type linkage table for the event created, the following processing steps take place:

- The receiver type function module [Extern] for determining the receiver type is called if specified in the type linkage table.

- If the event linkage is designated a **global event linkage**, the following occurs:
  - The check function module [Extern] is called if specified in the type linkage table.
  - If this function module terminates without an exception (or if no check function module is specified), the receiver function module is called with the transfer parameters `event`, `receiver type` and `event container`.
  - The instance linkage table is **not** evaluated.

- If the event linkage is **not** designated a **global event linkage**, the instance linkage table [Seite 1354] is evaluated.

If an error occurs within the function module `SWW_WI_CREATE_VIA_EVENT` (task missing or has errors, problems with tRFC), a mail is sent to the workflow system administrator.

**Features**

**Structure of the type linkage table**

The following fields are available in the type linkage table:

- **Object type**
  Type of object for which the event is created.

- **Event**
  Event created.
Object type and event must be defined in the Business Object Repository.

- **Receiver type**
- **Receiver function module**
- **Check function module**
- **Receiver type function module**
- **Destination**

Standard name of a logical RFC destination in which the receiver function module is called. If no destination is specified, the receiver function module is called locally in the destination `WORKFLOW_LOCAL_<Client>`.

The workflow runtime system only uses events locally and does not complete this field. Do not enter a destination.

For further information, refer to [Destination Types [Extrem]].

- **Indicator: Global**

  If the event linkage is shown as *global*, the receiver function module specified is called each time the object type/event pair occurs, irrespective of the triggering object instance.

  The instance linkage table is not evaluated for this event.

  The indicator is set automatically by the workflow system:

  - The indicator is set for triggering events.
  - The indicator is not set for terminating events and wait steps.

  This indicator can only be set manually with the following function modules:

  - `SWE_EVENT_REC_TYPE_GLOBAL`
  - `SWE_EVENT_REC_TYPE_NOT_GLOBAL`

- **Indicator: Type linkage activated**

  If you set this indicator, the type linkage between the event creator and the event receiver is activated.

- **Indicator: Enable event queue**

  This indicator is only evaluated if the event queue is active [Seite 1519].

  If the indicator is set, the event receiver is started via the event queue. If the indicator is not set or the event queue is not active, the receiver is started immediately.

- **Behavior if error feedback**

  The system presetting maintained on the tab page Basic data [Seite 1517] in event queue administration can be adopted or revised.

- **Receiver status**
Evaluation and Maintenance of Type Linkage Table

The status of the event receiver affects the starting of the receiver. In general, the receiver linkage must be active to start the receiver. The status of the receiver linkage has the following effects:

- **No errors**
  
  If the receiver linkage has the status *no errors*, the receivers are started by the system. They can be started directly or via the event queue.

- **Errors**
  
  If the receiver linkage has the status *errors*, the receiver is not started immediately, but is put into the event queue automatically instead. There the entry is assigned the status *Waiting because of errors*. So the receiver is not started, but there is an opportunity to start the receiver again after the error has been removed. Triggered events do not get lost even with linkages that have errors. The event receivers can be started again using the tab page Linkages with errors [Seite 1525].

Maintenance of type linkage table

You have to process the type linkage table manually if:

- You use event control within your own non-workflow developments.
- You want to analyze the entries made by the workflow system.
- You want to add a check function module at a later date.
- You want to activate an event receiver linkage.
- You want the events of a particular linkage to use the event queue [Seite 1515].

Otherwise, the maintenance of the type linkage table (entering linkages for triggering or terminating events) is carried out by the workflow system.

It is possible to create a full entry in the type linkage table using the following function module:

```
SWE_EVENT_REC_TYPE_ENTER.
```

APIs are also available for processing individual fields in the type linkage table.

<table>
<thead>
<tr>
<th>Name of function module</th>
<th>Explanation</th>
</tr>
</thead>
<tbody>
<tr>
<td>SWE_EVENT_REC_TYPE_CHECK_FB</td>
<td>Changing the check function module for the event receiver</td>
</tr>
<tr>
<td>SWE_EVENT_REC_TYPE_DISABLE</td>
<td>Deactivating a type linkage</td>
</tr>
<tr>
<td>SWE_EVENT_REC_TYPE_ENABLE</td>
<td>Activating a type linkage</td>
</tr>
<tr>
<td>SWE_EVENT_REC_TYPE_FB</td>
<td>Changing the receiver function module</td>
</tr>
<tr>
<td>SWE_EVENT_REC_TYPE_GET_FB</td>
<td>Changing the function module for the type determination of the event receiver</td>
</tr>
<tr>
<td>SWE_EVENT_REC_TYPE_GLOBAL</td>
<td>Setting a global event linkage</td>
</tr>
<tr>
<td>SWE_EVENT_REC_TYPE_NOT_GLOBAL</td>
<td>Deleting a global event linkage</td>
</tr>
<tr>
<td>SWE_EVENT_REC_TYPE_READ</td>
<td>Reading a type linkage</td>
</tr>
</tbody>
</table>

Use the following function module to delete an entry from the type linkage table:

```
SWE_EVENT_REC_TYPE_DELETE
```

If you make entries in the type linkage table, make sure you delete them again when a type linkage is no longer expected. Ensure that there are no instance linkages for this type linkage.
Activities
To maintain the type linkage table, choose Tools → Business Workflow → Development → Utilities → Events → Type linkages.

The overview displays the existing type linkages with details of object type, event and receiver type. Choose Edit → New entries to add new entries to the type linkage table. Choose Goto → Details to revise entries.
Event Simulation

Use
The creation of an event [Extern] is simulated and the result specifies which tasks and workflows would be started as receivers of this event.

Prerequisites

Authorization
You require appropriate authorization for the functions shown. This is the authorization (authorization object S_PROGRAM) for executing a program for authorization group SWU_DIAG.

Features
The result of the test also shows whether errors were diagnosed in connection with the simulated start of the task, which may result in the task or workflow not being started.
The respective error causes are designated with a letter. The legend can be consulted for a short explanation of the error causes. The most common error causes are:

- There is no workflow definition for the workflow to be started (error cause a).
- The workflow is not activated (error causes b, c).
- The linkage between the triggering event and the task or workflow has not been activated (error cause e).
- The binding definition from the event container to the workflow or task container has errors (error causes h, i). The task or workflow may nevertheless be started but will probably not be processed as expected.

To remove the error in all of the above cases, position the cursor on an entry and choose Goto → Task. This takes you to the definition of the workflow or task in question, where you can take appropriate measures.

You activate workflows in the Workflow Builder.

- The linkage between the event and the task/workflow is entered in the linkage table but the task does not exist (error cause g).
  
  This error occurs if you delete a task/workflow defined with triggering events. You must then use transaction SWE2 to process the type linkage table manually and delete the relevant entry.

- The linkage between the triggering event and the task/workflow to be triggered has been created for the task/workflow, but has not been entered in the linkage table (error cause d).
  
  This error indicates an inconsistency between two tables. Call the task/workflow (Goto → Task) and save it again.

- The actual execution of the task to be started still depends on a check function module [Extern] being run without errors. It therefore cannot be finally assessed at this point (error cause f).

- Calls with errors or non-executable calls to tasks/workflows can, for technical reasons, also affect other calls, which may themselves be successful (error causes j, k).
Activities

Executing simulations

Choose Tools → Business Workflow → Development → Utilities → Event → Simulate event to simulate an event.

3. Describe the event to be triggered with object type and event ID.

4. Choose to simulate the event.

You can also simulate an event directly from a step of the type event creator.
Creation of Events for Test Purposes

Use
You can use this function to create an event for test purposes. This can be used, for example, to start a task or a workflow via an event, and hence check the definition and functionality of triggering events.

⚠️ This function is purely a test function that you do not use in normal operation. The object status change that is normally reported by the event did not occur.

Activities
To create an event for test purposes, choose Tools → Business Workflow → Development → Utilities → Events → Create event.

5. Describe the event to be triggered with object type and event ID.
6. Enter an object key so that the event is generated for a unique object.
7. Decide how the receiver function module is to be called:
   - **Trigger receiver FM with delay**
     The system calls the receiver function module as a tRFC with a delay of 10 minutes. In the meantime, you can call the tRFC log under your name (Environment → RFC queue) in order to call the receiver function module in the debugger from there. The call is then made in a separate logical context.
   - **Trigger receiver FM synchronously**
     The system calls the receiver function module synchronously. You have the opportunity to trace the determination of the receiver (search in linkage table) and the call to the receiver function module in the debugger, and to branch to the receiver called.

Activate the debugging mode via the OK code /h before triggering the event.

8. Select Create event.

If the expected reaction to the event created does not occur, check whether the event is entered as a triggering event of a task and whether the linkage is activated.
Event Trace

Use
All events created correctly are logged in the event trace irrespective of whether potential receivers exist. If a receiver is entered in the event trace, it does not necessarily mean that this receiver was called successfully.

Prerequisites
The event trace is only written if logging has been activated. To trace any events which may not have been created, you must first activate the event trace and then create the event again.

Features
The following data is logged in the event trace:

Event data
- Triggering object type, triggering object
- Event ID
- Triggering program
- Trigger date and time

Receiver or linkage data
- Receiver function module
- Receiver type
- Receiver instance
- Linkage status

Activating/deactivating the event trace
As well as for activating or deactivating the event trace, you can also use this function for specifying selection criteria so that only certain events are logged.

Displaying the event trace
The event trace display can also be restricted using a selection screen. In addition to the criteria for event data, receiver data and event receiver linkage data, the display can also be restricted to linkages with errors.

Deleting the event trace
The deletion of events can also be controlled using selection criteria. The function does not therefore always delete the entire event trace automatically. If you mark the field Only display list, the events you select are not deleted, but displayed in a list.

Activities
- To activate/deactivate the event trace, choose:
Event Trace

Tools → Business Workflow → Development → Administration → Event manager → Event trace → Switch event trace on/off

- Select the function Selective tracing on the dialog box that follows to choose selection criteria.
- To display the event trace, choose:
  Tools → Business Workflow → Development → Administration → Event manager → Event trace → Display event trace
- To delete the event trace, choose:
  Tools → Business Workflow → Development → Administration → Event manager → Event trace → Delete event trace
Business Workplace: Workflow Functions

Use
You use this part of the Business Workplace if you want to use the functions of SAP Business Workflow.

The Business Workplace is the main interface between an end user and the workflow system. All dialog and missed deadline work items to which the user is assigned as a recipient are displayed in the user’s workflow inbox.

As a head of department, you are responsible for approving leave requests. The relevant approval process is implemented using a workflow in your enterprise.

The requests (in the form of work items) appear in your worklist (workflow inbox) and must be rejected or approved there.

The rejected or approved requests (executed work items) are not only returned to the applicants after processing, but are also put into your workflow outbox (under Work items executed by me). You can therefore check the requests you have processed.

Features

Workflow settings
You can configure the workflow functions in the Business Workplace using the personal workflow settings.

Business Workplace screen areas
The Business Workplace has three screen areas, which are used in the following manner by SAP Business Workflow:
Overview tree

The following workflow functions are available under the Inbox node, which is under the initial node Workplace:

- Workflow
  - Grouped according to task
  - Grouped according to content
  - Grouped according to content type
  - Grouped according to sort key
- Overdue entries
- Deadline messages
- Incorrect entries

For information on these functions, refer to Workflow Inbox [Seite 1408].

The following functions are available under the Outbox node, which is under the initial node Workplace:

- Started workflows
- Work items executed by me
- Forwarded work items

For information on these functions, refer to Workflow Outbox [Seite 1441].

The Resubmissions node is located under the initial node Workplace and contains the:

- Workflow resubmissions [Seite 1443]
**Worklist**

The worklist is displayed in the upper right corner of the Business Workplace screen. Depending on whether you are in the workflow inbox, the workflow outbox or the workflow resubmissions, you have various functions available to you, which are described at the respective locations.

**Work item preview**

In the lower right corner of the Business Workplace screen, a work item selected in the worklist is displayed in a preview. Not all the functions of the work item display or the workflow log are available.

A user exit can be used to configure the work item preview to suit your individual requirements.

**Support for context menus**

All workflow functions can be called using the relevant context menu.

**Workflow Toolbox**

*SAP Business Workflow’s Workflow Toolbox* enables the user to access workflow functions even during a workflow-driven application transaction.

**E-mail notification for new work items**

The report RSWUWFML can be used to inform an employee by mail that there is a new work item in their Business Workplace inbox.

This function is therefore beneficial to all employees who do not work with their Business Workplace on a daily basis.

**Activities**

To access the Business Workplace from the *SAP Easy Access* screen, choose one of the following options:

-  
-  
-  

**Menu → Business Workplace**

**Tools → Business Workflow → Development → Runtime Tools → Business Workplace.**
Work Item

Definition
Object that represents a task or action in the workflow system at runtime.

Use
Work items are subdivided into a specific work item type according to their assignments. The internal processing procedures are controlled via this work item type. The work item type determines which statuses and transitions are valid.
Depending on the work item type, some of these work items are displayed in a user's work list. Other work items, on the other hand, are only used and processed internally.

Structure

Work item types displayed in the Business Workplace

<table>
<thead>
<tr>
<th>Work item type</th>
<th>Short text</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>W</td>
<td>Work item with dialog [Seite 1372]</td>
<td>Runtime representation of single-step tasks that require interaction with the user.</td>
</tr>
<tr>
<td>D</td>
<td>Missed deadline [Seite 1377]</td>
<td>Work item for notification of missed deadline</td>
</tr>
<tr>
<td>A</td>
<td>Work Queue [Seite 1390]</td>
<td>A work queue is a list of objects to be processed once and together in a limited time frame.</td>
</tr>
</tbody>
</table>

Other work item types

<table>
<thead>
<tr>
<th>Work item type</th>
<th>Short text</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>F</td>
<td>Workflow [Seite 1380]</td>
<td>Runtime representation of a multistep task</td>
</tr>
<tr>
<td>B</td>
<td>Batch item [Seite 1385]</td>
<td>Runtime presentation of a single-step task that runs in the background</td>
</tr>
<tr>
<td>E</td>
<td>Wait step work item [Seite 1392]</td>
<td>Runtime representation of a wait step in the workflow definition</td>
</tr>
<tr>
<td>C</td>
<td>Container anchor</td>
<td>This type of work item is required as a special development in the EDI environment [Extern]. It does not normally appear in the workflow environment. Work items of this type should be regularly deleted or archived.</td>
</tr>
</tbody>
</table>
Dialog Work Items (Type W)

Definition
Work item that represents a task at runtime that requires interaction with the user.

Since the user decision is also represented internally by a task, a dialog work item can also represent a user decision.

When a dialog work item is executed, the underlying object method of the task is called. The deadlines for executing dialog work items are monitored.

Use
A dialog work item is displayed with ready status in the workflow inbox of the Business Workplace. It is removed from the integrated inboxes of the other agents when the recipient reserves, executes, or processes this work item with other functions.

The database oriented approach used in SAP Business Workflow allows a work item to be seen by several recipients equally authorized in organizational terms in their inboxes and executed from there. However, only one recipient can actually reserve this work item for processing and execute it. The work item is then no longer available to any other recipients.

Integration
A task represented by a dialog work item can be

- a step in a workflow definition:

  In the workflow definition, reference is made to tasks in the activity and user decision steps.

- started as single steps via an event or in dialog:

  Tasks can be started as elementary activities directly in dialog or via a triggering event. These tasks are then also represented by a dialog work item in the workflow inbox.
## Status of a Dialog Work Item

The valid statuses for dialog work items (type W) are listed in the following table.

<table>
<thead>
<tr>
<th>Work item status</th>
<th>Description</th>
</tr>
</thead>
</table>
| waiting           | The work item has been scheduled for its *requested start*. A work item has this status:  
|                   | - if it already exists but the *requested start* specified in the workflow definition has not been reached yet.  
|                   | - if it has been set to resubmission Work items in the *waiting* status are not displayed in the workflow inbox. |
| ready             | The work item has been released for execution and appears in the workflow inbox of all recipients. |
| reserved          | The work item has been received by one of its recipients with the result that its status has changed from *ready* to *reserved*. A work item in the *reserved* status is then displayed to this recipient only. It is no longer displayed in the workflow inboxes of the other recipients. |
| In process        | The work item is currently being processed by a recipient or in a different mode. A work item also has this status:  
|                   | - if the work item is waiting for its terminating event.  
|                   | - if the user cancelled the method.  
|                   | - if the method was ended with a temporary exception for which no subsequent steps have been modeled. The point at which processing is completed cannot be detected by the workflow system in this status. As long as the status of the work item is set to *in process*, database changes have not been made. |
| Executed          | The work item is awaiting explicit confirmation of its completion. The work item only has this status if it is necessary to confirm that it has been completed. A work item with *executed* status can be executed or forwarded several times until it is set to the status *done* in the Business Workplace. In this way, groupware components are realized in SAP Business Workflow. |
| completed         | The execution of the work item is completed. The result of the task represented by the work item is correct, i.e. the result modeled in the workflow definition. Work items in the *completed* status are not displayed in the workflow inbox of the Business Workplace. |
| Logically deleted | Execution of the work item is no longer meaningful or required by the workflow logic. A work item changes to the *logically deleted* status in the following way:  
|                   | - Termination in parallel processing branches  
|                   |   When the required number of processing paths has been executed in a fork, the work items in the other paths that have not yet reached the *completed* status are automatically set to the *logically deleted* status.  
|                   | - Intervention by an administrator  
|                   |   An administrator can only set a work item to the *logically deleted* status if it has not yet reached the *completed* status and is not part of a higher-level workflow. |
### Status of a Dialog Work Item

<table>
<thead>
<tr>
<th>Status</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Work items in the <em>logically deleted</em> status are not displayed in the workflow inbox of the Business Workplace.</strong></td>
<td></td>
</tr>
<tr>
<td><strong>error</strong></td>
<td>Execution of the work item was terminated with an error.</td>
</tr>
</tbody>
</table>

A work item with the *logically deleted* status may have caused database changes or other actions (generate event, send notification). These changes are not canceled automatically (compensated).

In addition to the statuses given above, a work item can be **locked against execution**. This is possible in any status and involves an administrative function which is selected for a work item via the *Change* option.
Status Transitions of a Dialog Work Item

The diagram below shows the possible status transitions that a dialog work item (type W) can undergo:

Status and Status Transitions of Work Items (W)

The arrows are labeled with the functions used by a user to trigger the respective status transition.

Comments
For further information, refer to Status of a Dialog Work Item [Seite 1373].

Transition from status waiting
The transition from the status waiting to the status ready is performed automatically by the work item manager when the requested start is reached. A workflow system administrator can set a work item to the status ready manually.

Transition from status ready
From this status, the work item passes either to the status reserved or via the status in process to the status completed.

Transition from status reserved
A work item with the status reserved can be reset to the status ready.

Transition to and from status in process
A work item with the status in process can be reset to the status ready. A workflow system administrator can reset a work item manually. This function is available when changing the work item.

Transition from status executed
After confirmation of end of processing, the work item assumes the status completed.
Status Transitions of a Dialog Work Item

**Transition from status completed**
Work items with the status *completed* can no longer be set to another status even if a workflow system administrator intervenes.

**Transition from status error**
A workflow system administrator can intervene and set work items with errors to the status *in process* or the status *logically deleted* (possibly after eliminating the error).

**Transition from status logically deleted**
Work items with the status *logically deleted* can no longer be set to another status even if a workflow system administrator intervenes.
Missed Deadline Work Item (Type D)

Definition
Notification of a deadline recipient if the runtime system detects that the deadline for a certain work item has been exceeded.

Use
This work item informs its recipients that a deadline (start or end deadline) of a monitored work item has been exceeded. The recipients are informed by means of a missed deadline work item (type D) in the workflow inbox of the Business Workplace. When it is executed, this work item displays information on the monitored (and now late) work item. The text for notifying the recipient is set by default. When a deadline is monitored for an activity [Seite 1024] or user decision [Seite 1074], it is also possible to enter an individual text in the respective task definition.
# Status of a Missed Deadline Work Item

The valid statuses for missed deadline work items (type D) are listed in the following table:

<table>
<thead>
<tr>
<th>Work item status</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Ready</td>
<td>The work item appears in the Business Workplace of the recipients.</td>
</tr>
<tr>
<td>In process</td>
<td>One of the recipients has executed the work item.</td>
</tr>
<tr>
<td>Completed</td>
<td>The execution of the work item is completed. Work items in the <em>completed</em> status are not displayed in the workflow inbox of the Business Workplace.</td>
</tr>
</tbody>
</table>
Status Transitions of a Missed Deadline Work Item

The following status transitions are possible for missed deadline work items (type D):

**Transition to and from status in process**

The status of a missed deadline work item changes from *ready* to *in process* when it is executed by a user. Execution of a missed deadline work item displays the most important information on the monitored work item.

The missed deadline work item remains in the *in process* status until end of processing is confirmed explicitly. Until then, the work item can be executed or forwarded several times.

**Transition to completed status**

The status of the missed deadline work item changes to the *completed* status when completion of processing has been confirmed explicitly.
Workflow Work Item (Type F)

Definition
Work item that represents a multistep task at runtime.

Use
For every multistep task started there is one type F work item. The workflow log and the workflow container can be accessed:
- For error diagnosis and error correction if no work items are displayed
- For information on steps of a workflow (including their current agents, notes, and ad hoc objects) already processed
- For modifying an ongoing workflow by changing the workflow container
- For reporting on completed processes

Type F work items are not displayed in the Business Workplace but can be found using the work item selection [Seite 1490].

Structure
A workflow consists of a sequence of work items that are executed by agents or the system. The work items represent the steps in the workflow definition that refer to a particular task. These are steps of the types activity [Seite 1024] or user decision [Seite 1074].

Integration

Workflow Execution

The Work Item Manager manages the processing of work items and monitors deadlines. To automate workflow processes, activities in the workflow can also refer to object methods which run in the
background. If this is the case, the work item manager initiates the calling of the background processes. The work items whose execution requires dialog can be accessed by the selected agents (determined from the organizational model and role resolution) from their worklists in order to select them for processing. This worklist is displayed and managed in the workflow inbox of the Business Workplace [Seite 1368].
# Status of a Workflow Work Item

The valid statuses for workflow work items (type F) are listed in the following table:

<table>
<thead>
<tr>
<th>Work item status</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Waiting</strong></td>
<td>The (sub) workflow is scheduled but its requested start has not yet been reached.</td>
</tr>
<tr>
<td><strong>Ready</strong></td>
<td>Temporary interim status</td>
</tr>
<tr>
<td><strong>In process</strong></td>
<td>Execution of the (sub) workflow has begun.</td>
</tr>
<tr>
<td><strong>Completed</strong></td>
<td>The end of the (sub) workflow has been reached.</td>
</tr>
<tr>
<td><strong>Logically deleted</strong></td>
<td>Execution of the (sub) workflow is no longer required or meaningful. A status of a workflow changes to <em>logically deleted</em> in the following way:</td>
</tr>
<tr>
<td></td>
<td>• Intervention by an administrator.</td>
</tr>
<tr>
<td></td>
<td>An administrator can only set a workflow item to the <em>logically deleted</em> status if it has not yet reached the <em>completed</em> status.</td>
</tr>
<tr>
<td></td>
<td>• Termination in parallel processing branches</td>
</tr>
<tr>
<td></td>
<td>When the required number of processing paths has been executed in a fork, the work items in the other paths that have not yet reached the <em>completed</em> status are set to the <em>logically deleted</em> status.</td>
</tr>
<tr>
<td></td>
<td><img src="image1" alt="Diagram" /></td>
</tr>
<tr>
<td></td>
<td>A workflow set to the <em>logically deleted</em> status is recursively scanned for dialog and workflow work items (type W or F) that do not yet have the status &quot;completed&quot;. These work items are then also set to the status <em>logically deleted</em>.</td>
</tr>
<tr>
<td></td>
<td><img src="image2" alt="Diagram" /></td>
</tr>
<tr>
<td></td>
<td>A work item changed from the <em>executed</em> status to the <em>logically deleted</em> status may already have caused database changes or other actions (generate event, send notification). These changes are <em>not</em> canceled automatically (compensated), but are recorded in the log.</td>
</tr>
<tr>
<td><strong>Errors</strong></td>
<td>An error occurred in workflow control.</td>
</tr>
</tbody>
</table>

Workflows and their statuses are **not** displayed in the Business Workplace. Information about the workflow can be accessed via the subordinate dialog work items.
Status Transitions of a Workflow Work Item

The diagram shows the possible status transitions.

Status and Status Transitions of Work Items (F)

The statuses of a workflow work item and the permissible status transitions always concern an entire workflow.
A step (activity) in the workflow definition can reference both a task and another workflow. A workflow can therefore also contain subordinate subworkflow items.

Comments

Transition from status waiting
The work item manager automatically carries out the transition from the waiting status to the ready status when the requested start date/time of the workflow has been reached.
A workflow system administrator can set a work item to the status ready manually.

Transition from status ready
The ready status is a temporary interim status because it only exists until the first work item of the workflow has been created.

Transition to and from status in process
The status of the workflow changes to in process as soon as the first work item of this workflow has been created. The workflow remains in this status until the entire workflow definition has been processed.

Transition to incorrect status
An error occurs during workflow control or coordination.

Role resolution for determining an agent does not return a result which can be used.
A workflow system administrator can intervene and set workflows with errors to the status in process or the status logically deleted (possibly after eliminating the error). If a workflow is incorrect, the responsible workflow system administrator specified either globally in Customizing or in the basic data of each workflow definition is notified by mail.

This status does not mean that a dialog work item of this workflow has the incorrect status.

**Transition to completed status**

A workflow is set to the status completed when the last step of the relevant workflow has been completed.
Background Work Item (Type B)

Definition
Work item that represents a single-step task at runtime whose execution does not require a dialog and, therefore, can be controlled automatically by the system.

Integration
Type B work items are not displayed in the Business Workplace.
## Status of a Background Work Item

The valid statuses for background work items (type B) are listed in the following table.

<table>
<thead>
<tr>
<th>Work item status</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Waiting</strong></td>
<td>The work item has been scheduled for its requested start. A work item has this status after it has been created until the requested start specified in the workflow definition is reached.</td>
</tr>
<tr>
<td><strong>Ready</strong></td>
<td>Temporary intermediate status of a background work item. The system calls the associated object method as soon as a background work item can be processed. The status of the background work item then changes to in process immediately.</td>
</tr>
<tr>
<td><strong>In process</strong></td>
<td>The work item is currently being processed. A work item also has this status • if the method was left with a temporary exception. In this case, special Error Handling for Background Work Items with Temporary Errors [Seite 1388] is carried out. • if the work item is waiting for its terminating event. The point at which processing is completed cannot be detected by the workflow system in this status.</td>
</tr>
<tr>
<td><strong>Completed</strong></td>
<td>The execution of the work item is completed. The result of the task represented by the work item is correct, i.e. the result modeled in the workflow definition.</td>
</tr>
<tr>
<td><strong>Logically deleted</strong></td>
<td>Execution of the work item with regard to the process logic is no longer meaningful or necessary for the process to continue. A work item changes to the logically deleted status in the following way: • Termination in parallel processing branches When the required number of processing paths has been executed in a fork, the work items in the other paths that have not yet reached the completed status are automatically set to the logically deleted status. • Intervention by an administrator The administrator can only set a work item to the logically deleted status if it has not yet reached the completed status and is not part of a higher-level workflow. A work item with the logically deleted status may have caused database changes or other actions (generate event, send notification). These changes are not canceled automatically (compensated).</td>
</tr>
<tr>
<td><strong>error</strong></td>
<td>Execution of the work item was terminated with an error.</td>
</tr>
</tbody>
</table>

In addition to the statuses given above, a work item can be locked against execution. This is possible in any status and involves an administrative function which is selected for a work item via the Change option.
Error Handling for Background Work Items

Use

Error handling for background work items must be carried out by the workflow system because background work items (with errors) are not displayed in the Business Workplace and are therefore detected too late, if at all. Because background work items that are temporarily incorrect, for which another execution attempt can (theoretically) be successful, cannot be restarted by a user. (Temporarily incorrect work items are work items whose method was terminated with a temporary exception for which no subsequent step has been defined.)

Features

In Customizing for the workflow system, you can define how often the workflow system attempts to restart a temporarily incorrect work item. You can also define the interval between two repetition attempts and activate the monitoring function. Monitoring is also activated as part of automatic Customizing [Extern] (Tools → Business Workflow → Development → Utilities → Customizing).

You can also specify the maximum number of repetition attempts (“repetition counter”) separately for each individual background step in the respective workflow definition. This setting overrides the value in Customizing if it is greater than zero.

If an error (method exception) occurs when background work items are executed by the workflow system, the system responds as follows depending on the type of exception and error modeling settings:

<table>
<thead>
<tr>
<th>In the workflow definition...</th>
<th>The exception is defined for the object method as a...</th>
<th>System or application error</th>
</tr>
</thead>
<tbody>
<tr>
<td>a subsequent step is modeled for the exception.</td>
<td>The step has status completed and the modeled subsequent step is executed. Background work items for which a repetition counter is explicitly modeled in the workflow definition are restarted first by the system. If all of the attempts are unsuccessful, the status of the work item changes to completed and the modeled subsequent step is executed.</td>
<td>The step has status completed and the modeled subsequent step is executed.</td>
</tr>
</tbody>
</table>
### Error Handling for Background Work Items

| No subsequent step is modeled for the exception | The step is not yet completed. The respective work item retains the status in process. Background work items are restarted by the system. The number of repetition attempts is determined either by the repetition counter in the step definition or - if this is equal to zero - by the repetition counter set in Customizing. If all of the attempts are unsuccessful, the work item status changes to incorrect. | Workflow and work item assume the error status. |

### Processing Incorrect Work Items

The workflow system sends a mail to the relevant system administrator for every background work item with the status incorrect.

### Processing Work Items That Have Been Started

The workflow system determines all of the background work items that have been in process for longer than 30 minutes. An error message is then sent to the workflow system administrator for all of these background work items, since the system assumes that processing has been cancelled. However, this does not necessarily mean that an error has occurred.
Work Queue Work Item (Type A)

Definition
Work item that represents a work queue.
A work queue comprises a list of objects that are to be processed once together within a specified period.
The work queue serves as a framework for the individual entries to be processed and manages the list of
objects to be processed including their statuses and the tasks to be performed on them.
Work queue work items are displayed in the Business Workplace. The work item status indicates the
overall processing status of the work queue.

Use
Once you have created the work queue work item and you know its work item ID, you have the following
options:
- Process the work queue within a workflow.
- Control the release, processing and status evaluation of the work queue with function
  modules.
- Control the release and processing of the work queue by processing the work queue work
  item directly.

Integration
To create the work queue work item from the list, you call the function module SWZ_AI_CREATE.
## Status of a Work Queue Work Item

A work queue work item can have the following statuses:

<table>
<thead>
<tr>
<th>Work item status</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><img src="image" alt="Waiting" /> waiting</td>
<td>The work queue has been created but cannot be released yet.</td>
</tr>
<tr>
<td><img src="image" alt="Ready" /> ready</td>
<td>The work queue work item (type A) is displayed in the Business Workplace of the recipients passed in the table AGENTS of the function module SWZ_AI_CREATE.</td>
</tr>
<tr>
<td><img src="image" alt="Reserved" /> reserved</td>
<td>A person with release authorization has reserved the work queue.</td>
</tr>
<tr>
<td><img src="image" alt="In process" /> in process</td>
<td>Work queue processing has begun.</td>
</tr>
<tr>
<td><img src="image" alt="Completed" /> completed</td>
<td>All lines in the work queue have the status COMPLETED or CANCELLED and have reported back accordingly to the work queue. A work queue work item (type A) that assumes this status automatically creates the event created.</td>
</tr>
<tr>
<td><img src="image" alt="Logically deleted" /> logically deleted</td>
<td>Further processing of the work queue is invalid (and therefore no longer possible).</td>
</tr>
<tr>
<td><img src="image" alt="Error" /> error</td>
<td>At least one line of the work queue has the status error.</td>
</tr>
</tbody>
</table>

If the work queue is to be processed via a workflow and therefore not appear as a separate work item in any Business Workplace, a non-existent user must be specified in the table when the work queue is created. The work queue must be reserved for further processing.
Wait Step Work Item (Type E)

Definition
Work item that represents a wait step or a workflow at runtime, which is waiting for an event to occur.

Use
Type E work items are not displayed in the Business Workplace.
Work Item with Express Notification

Use
When the system creates a work item with priority 1, each of its recipients receives an express notification (dialog box with appropriate text) on the screen. The recipient can call the Business Workplace directly from the express message. Excluded agents do not receive a message.

Constraints
The system does not create an express message

- if the user processes the work item immediately due to advance with immediate dialog [Seite 1453].
- if the work item represents a general task that is not restricted to certain agents.
- if the work item was forwarded.

The system only creates express notifications when a work item is created. An express notification is not created for the new recipients of priority 1 work items that are forwarded.

If one of the selected agents processes the work item, the other agents still receive an express message.

Features
Express messages are only sent for dialog steps as soon as the system has created them with the ready status. If the work item is created first with the waiting status because its requested start has not been reached yet, the express message is not sent until the status changes from waiting to ready.

Activities

How is the priority set?
The priority of a work item can be determined for steps that require a dialog with the user. It is defined in the tab page Miscellaneous of the step definition.
Workflow Settings

Use
You use the workflow settings to maintain the special workflow functions in the Business Workplace.

Features
The following functions are available:

- Personal settings [Seite 1395]
- Display organizational assignment [Seite 1397]
- Refresh organizational environment [Seite 1397]
- Adopt substitution [Seite 1400]
- End substitution [Seite 1400]
- Maintain substitute [Seite 1400]
- Activate substitute [Seite 1400]
- Adopt view [Seite 1404]
- Exit view [Seite 1404]

Activities
You can access the workflow settings within the Business Workplace [Seite 1368] via Settings → Workflow settings.
Personal Settings

Use
The personal settings for workflow enable you to adapt the runtime system to suit your requirements.

Features

Work item display

<table>
<thead>
<tr>
<th>Setting</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>User view with ActiveX (32-bit)</td>
<td>Both of these settings define the work item display as the default. Note that the ActiveX variant is only available on 32-bit platforms.</td>
</tr>
<tr>
<td>User view without ActiveX</td>
<td>This setting defines the technical work item display as the default.</td>
</tr>
</tbody>
</table>

Workflow log

<table>
<thead>
<tr>
<th>Setting</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>User view with ActiveX (32-bit)</td>
<td>User view of the workflow log without any technical aspects. This view uses ActiveX controls and is therefore only available on 32-bit platforms.</td>
</tr>
<tr>
<td>User view without ActiveX</td>
<td>User view of the workflow log without any technical aspects. This view uses the hierarchical list display of the ABAP List Viewer.</td>
</tr>
<tr>
<td>Technical view</td>
<td>In addition to the main semantic information, the technical view of the workflow log also displays technical numbers and texts that may not be available in the logon language of the user. The technical view is intended primarily for system administrators.</td>
</tr>
</tbody>
</table>

Further settings

<table>
<thead>
<tr>
<th>Setting</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Display work item texts in logon language</td>
<td>All work item texts in the Business Workplace are always displayed in the user’s logon language. If the user’s logon language is different to the language in which the work item was created, additional database accesses are required that can delay display of the Business Workplace.</td>
</tr>
<tr>
<td>Enable forwarding of work items to several users</td>
<td>The work item can be forwarded to several agents. The term “agent” is used in a broad sense in this context. You specify the agent as an organizational object (organizational unit, job, position, user) when forwarding the work item. This organizational object can consist of several persons. For further information, refer to Forwarding.</td>
</tr>
<tr>
<td>(Double)-clicking on an object displays the object in the same window.</td>
<td>You can choose an object in the work item display, which is then displayed in the current</td>
</tr>
</tbody>
</table>
### Personal Settings

<table>
<thead>
<tr>
<th>Setting</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>session window and replaces the work item display. When you exit the object display, you return to the work item display. If you do not set the indicator, a new session is created to display the object.</td>
<td>This setting does not apply if you work with the SAP GUI for HTML.</td>
</tr>
<tr>
<td>No tips &amp; tricks in workplace</td>
<td>The Business Workplace [Seite 1368] includes tips and tricks for working with work items and workflows in the work item preview. If you set this indicator, the tips and tricks are not displayed.</td>
</tr>
<tr>
<td>No HTML in execution of decision tasks.</td>
<td>You use this indicator to decide whether the execution of decision tasks is displayed in HTML or not. Deactivating the HTML display may make sense, for example, if you have problems displaying HTML documents generally because of the settings in your local HTML browser.</td>
</tr>
</tbody>
</table>

### Activities

To change the personal settings for workflow, you choose **Settings → Workflow settings → Personal settings** in the Business Workplace.

The settings are saved as user parameters and take effect the **next time** the work item display or workflow log is **called**.

You can change the standard values for these personal workflow settings in Customizing for SAP Business Workflow. These standard values are always used if users have not maintained settings of their own.
Organizational Assignment of a User

Use
The worklist displayed is formatted specifically for the user who is logged on. You can query this user's organizational assignment.

Prerequisites
The functionality described depends on a setting in the PD system table [Seite 1533].

Features
The organizational assignment of a user is buffered and read at each new logon or once a day (but not each time the Business Workplace is called). If the organizational assignment of a user changes while the user is logged on and must be updated, you choose Settings → Workflow settings → Refresh organizational environment in the Business Workplace.

Activities
To view current information on the assignment of the user logged on in the organizational plan of the enterprise, choose Settings → Workflow settings → Display organizational assignment in the Business Workplace.

The following information is displayed for the user logged on:
- The name maintained in the user address
- The organizational unit
- The position the user occupies
- The job describing the position

By double-clicking on an entry, you can display the staff assignments or the job description.
Substitutes for Workflows

Use

*SAP Business Workflow* implements a substitution arrangement to deal with a user's duties in their absence.

Prerequisites

The automatic appearance of work items in the substitute's worklist is dependent on a setting in the PD system table [Seite 1533]. It is possible to define which objects can be entered as position-related substitutes in Administration of Organizational Management [Seite 1530].

Features

Who is involved?

Substitution always involves two users:

- One user specifies a substitute: User B
- The other user (the substitute) 'adopts' the substitution: User A

The substitute can process "foreign" work items for the duration of the substitution. It is irrelevant whether they are assigned to the underlying single-step tasks as a possible agent.

How does substitution work?

Substitution works in two ways:

11. User A adopts substitution for user B and for the duration of the substitution sees only the work items seen previously by B in their workflow inbox.

   User B must have entered user A as their substitute for a particular period without activating the substitution. During this period, A can adopt substitution for B at any time without having to confer with B.

   Instead of their own worklist, user A is displayed user B's worklist. B's name is in the column Substitution for in the worklist, as long as this column is included in the current configuration of B's Business Workplace.

12. User A automatically sees not only the work items previously seen by employee B, but also their own work items in their Business Workplace. This also applies to all work items generated for B in the future ("automatic forwarding").

   User B must enter user A as substitute and activate the substitution. No further action on the part of user A is required for this kind of substitution.

   A can continue working as usual. They see a Business Workplace to which B's work items are added dynamically. A can recognize these work items by the fact that B's name is in the column Substitution for, as long as this column is included in the current configuration.

In both cases, user A can perform operations on these "foreign" work items within the scope of the substitute profile assigned to them by B. It is irrelevant whether A is assigned to the underlying single-step tasks as a possible agent.

User B can continue working without any restrictions in both cases.
Activities
You can access substitute maintenance in the Business Workplace [Seite 1368] by choosing Settings → Workflow settings. For information on the individual functions, refer to Maintaining and Activating Substitutes [Seite 1400]
Maintaining and Activating Substitutes

Maintaining substitutes

6. In the Business Workplace [Seite 1368], choose Settings → Workflow settings → Maintain substitute.

The Maintain substitute dialog box appears.

To facilitate maintenance of the substitution, the Personal substitutes entry and the positions you occupy are displayed in the hierarchy on the Maintain substitutes screen.

You specify your substitute either as a personal substitute or as a position-related substitute.

- Personal substitute
  A personal substitute can see and execute all your work items, including those assigned to you via a personal agent assignment.
  You can only specify one other user as a personal substitute.

- Position-related substitute
  A position-related substitute can only see and execute the work items you have received on the basis of agent assignment at the level of position, job or organizational unit.
  You can enter another position or a user as a position-related substitute.

27. Position the cursor either on the entry Personal substitutes or on the relevant position, and select Create substitute.

You can maintain existing entries by double-clicking on the relevant entry.

If you want to maintain a position-related substitution, decide whether you want another position or another user as a substitute. Select either position or user as the substitute type and specify the number or the user name of the substitute.

28. Specify the validity period for the substitution on the detail screen for substitution.

Only within this period can the substitute adopt the substitution.

29. Specify a substitute profile in the dialog box Detail screen substitution.

Irrespective of whether you have created your substitute as a personal or position-related substitute, you can limit the scope of the work items displayed to your substitute by specifying a substitute profile.

30. Select the field Substitution active if applicable.

You must activate the substitution if you want work items to be visible for the substitute automatically from now on. If you do not activate the substitution, the substitute must adopt the substitution explicitly to see your work items.

31. Save your entries and exit substitute maintenance.
Activating substitutes

   The Activate substitute dialog box appears.
12. Select the substitutes that you want to activate, and choose the function Activate.

Deactivating substitutes

   The Activate substitute dialog box appears.
12. Select the substitutes that you want to deactivate, and choose the function Deactivate.

Adopting substitution

   The Choose substitution dialog box appears.
17. Select the user(s) for which you want to adopt substitution.
18. Exit the dialog box.

Ending substitution

6. In the Business Workplace, choose Settings → Workflow settings → End substitution.
   The substitution is ended.
Views

Use
The Business Workplace [Seite 1368] provides various views on the work items displayed in the workflow inbox.

Using an appropriate view, a superior can "see" and process the work items of their employees.

Choosing a particular view also gives you the opportunity to see work items of other users in your workflow inbox and process them with full functionality, although the underlying tasks are not organizationally assigned to you.

Prerequisites
Views only require action by an employee: This employee chooses a view from a catalog of defined views. The employee must have the authorization required to choose a particular view.

Authorizations
To choose a particular view, you require a corresponding authorization based on authorization object S_WF_LVIEW.
To maintain a particular view, you must have the relevant authorization. This is an authorization based on the authorization object S_TABU_DIS to maintain tables for authorization group SWES.

Features
Views are always based on an evaluation path [Seite 1534] starting from the employee who wishes to adopt a view and leading to the employees whose inboxes could be viewed. The employee adopts a view by selecting another employee from the result list of the evaluation.

Activities
The activities associated with this function include:

- Adopting and Exiting Views [Seite 1404]
- Maintaining Views [Seite 1403]
Maintaining Views

Prerequisites
A view is always based on an evaluation path. This evaluation path describes which relationships are traced from the user who wants to adopt the view to the users whose Business Workplaces can be viewed.
You can use one of the evaluation paths available in the system. If there are no suitable evaluation paths, you can define an evaluation path of your own.

⚠️ Different views can only be maintained as a customer setting.

Procedure

36. To call table maintenance for views, choose Tools → Business Workflow → Development → Definition tools → Worklist client → Maintain views.
   This displays the screen Change View "View for Maintaining Views": Overview.

Maintaining views is an activity that is described in the Implementation Guide and can be performed in Customizing.

37. Create a new view. To do this, choose the function New entries.

38. Assign a unique name to the view.

39. Specify an evaluation path.

40. Describe the view with a long text.

41. Specify a start evaluation path.
   This selection is optional. The start evaluation path is used to get an initial selection of objects, which is then evaluated further via the first evaluation path.

42. Select Save.
Adopting and Exiting Views

Procedure

Adopting views

16. To adopt a view (as view on your own work items), choose Settings → Workflow settings → Adopt view in the Business Workplace.

        You are now on the dialog box View: Choose Agent. Only the views for which you have an authorization are available.

17. Choose a view.

        On the basis of the evaluation path defined for the view, the system selects the positions, organizational units or users connected to you.

18. From the result list of this selection, choose the object whose workflow inbox you want to view.

Exiting views

A view is only active while the Business Workplace is displayed. The next time you call the Business Workplace, you are asked if you wish to adopt the view previously set.

To return to the standard view when working in the workflow inbox of the Business Workplace, choose Settings → Workflow settings → Exit view.
Dynamic Columns for the Business Workplace

Use
Up to 6 columns in the workflow inbox of the Business Workplace [Seite 1368] can be filled on a task-specific basis with contents that are determined dynamically at runtime.
The standard functions for filtering, sorting, and grouping are available for these columns.
Please compare with Selectable Columns for the Business Workplace [Seite 1406].

Features
If you want to include one of the “dynamic columns” into the workflow inbox display, you must specify an element from the task container for each task, from which the content of the column is established at runtime.

Work items that belong to different tasks are then also displayed with different information. Work items that belong to tasks for which this functionality is not used are displayed with a blank entry.

Activities
The column contents are maintained via Tools → Business Workflow → Development → Definition tools → Worklist client → Dynamic columns for worklist.

You can also define the column headings. These headings are displayed if all of the work items displayed in the Business Workplace refer to the same task.
Selectable Columns for the Business Workplace: Workflow

The columns displayed essentially determine the appearance and information content of the workflow inbox of the Business Workplace. Detailed knowledge of the columns is also required to make full use of the filter and grouping criteria.

You can determine the selection of columns via display variants. The following columns are available:

<table>
<thead>
<tr>
<th>Column</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>Work item is executable</td>
<td>Indicator denoting that the work item [Seite 1371] is executable. ( )</td>
</tr>
<tr>
<td>Work item title</td>
<td>Title of work item.</td>
</tr>
<tr>
<td>Status indicator</td>
<td>Status of work item. In symbol form.</td>
</tr>
<tr>
<td>Creation date</td>
<td>Date when the work item was created with the status ready or waiting for the first time. A work item is only created with status waiting if a requested start was declared for the work item and the work item is created before the requested start.</td>
</tr>
<tr>
<td>Creation time</td>
<td>Creation time of a work item.</td>
</tr>
<tr>
<td>Priority</td>
<td>Priority of the work item.</td>
</tr>
<tr>
<td>Attachments exist</td>
<td>Attachments are shown with the symbol.</td>
</tr>
<tr>
<td>End of processing must be confirmed</td>
<td>Indicator denoting that the end of processing must be confirmed explicitly. ( )</td>
</tr>
<tr>
<td>Work item overdue</td>
<td>Indicator denoting that a deadline has been missed for the work item. ( )</td>
</tr>
<tr>
<td>ID</td>
<td>Unique number of a work item, which is assigned internally by the system.</td>
</tr>
<tr>
<td>Type</td>
<td>Type of work item. In abbreviation form.</td>
</tr>
<tr>
<td>Task</td>
<td>Identification for the task represented by the work item (for example TS00008323).</td>
</tr>
<tr>
<td>Technical status</td>
<td>Status of work item. Technical name.</td>
</tr>
<tr>
<td>Work item type</td>
<td>Type of work item. Name</td>
</tr>
<tr>
<td>Task name</td>
<td>Name of task</td>
</tr>
<tr>
<td>Work item status text</td>
<td>Status of work item. Name</td>
</tr>
</tbody>
</table>
| Deadline status                | The deadline status specifies whether one of the deadlines has been missed. The possible values in this column are therefore:  
  - None  
  - Latest start  
  - Requested end  
  - Latest end  
  - Other  
| Current agent                  | Name of the user who last reserved or processed the
### Selectable Columns for the Business Workplace: Workflow

<table>
<thead>
<tr>
<th>Column</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Latest end date</td>
<td>Latest end of work item. The end is reached when the work item assumes the status completed.</td>
</tr>
<tr>
<td>Latest end time</td>
<td>Latest end time of a work item</td>
</tr>
<tr>
<td>Forwarder</td>
<td>Name of the party who forwarded the work item.</td>
</tr>
<tr>
<td>Substitution for</td>
<td>Name of the substituted party whose work item is displayed</td>
</tr>
</tbody>
</table>
| Work item content       | Column in which the default attribute of the object referenced in the container element 
                          | _ WI_Object_ID is displayed.                                                 |
| Group object            | Column in which the default attribute of the object referenced in the container element 
                          | _ WI_Group_ID is displayed.                                                  |
| Execution can be rejected | Indicator denoting whether execution of the work item can be rejected (X)   |
| Dynamic columns         | Refer to Dynamic Columns for the Business Workplace [Seite 1405].           |
Workflow Inbox

Use
The worklist of the user currently logged on to the Business Workplace is displayed in the workflow inbox.

Integration
As is the case for the workflow resubmissions [Seite 1443] and the workflow outbox [Seite 1441], the workflow inbox is an integral part of the Business Workplace.

Features
Views in the workflow inbox
A user's worklist can be displayed as an overview or according to the following grouping criteria:

<table>
<thead>
<tr>
<th>Grouped according to</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>task</td>
<td>The work items are grouped according to the tasks to which they belong.</td>
</tr>
<tr>
<td>content</td>
<td>The work items are grouped according to the object instances to which they belong.</td>
</tr>
<tr>
<td>content type</td>
<td>The work items are grouped according to the object types to which they belong.</td>
</tr>
<tr>
<td>sort key</td>
<td>The work items are grouped according to sort keys. Please refer to Grouping According to Sort Keys [Seite 1432].</td>
</tr>
</tbody>
</table>

You can also choose from the following views:
- Overdue entries
- Deadline messages
- Incorrect entries

Workflow Functions
The screen area in which the worklist is displayed has a toolbar containing the workflow functions and functions for configuring the list. For information about these functions, refer to the documentation on the ABAP List Viewer [Extern].

The workflow functions can be accessed in the Business Workplace via the toolbar or the relevant context menu (right-hand mouse button). The following functions can be executed on work items:

- **Update**
  This function updates the worklist [Extern] of the user currently logged on to the Business Workplace.

- **Execute**
  To be executed, a work item must have either the status ready or the status in process.

  When a dialog work item [Extern] is executed, the object method to which the single-step task for this work item refers is carried out.

  When a missed deadline work item [Extern] is executed, the most important information on the monitored work item is displayed.

- **Display work item**
  This function goes directly to the work item display [Seite 1411].
• **Reserve** (dialog work items only)
  This reserves a work item for execution by the end user in question. The work item must have the status *ready*. This work item is then no longer visible to the other recipients who could previously see it in status *ready*. The status of the work item changes from *ready* to *reserved*.

• **Replace** (dialog work items only)
  Replacing releases the work item from its reserved status and makes it available again to all of the original recipients. The status of the work item changes from *reserved* back to *ready*. The status of the work item must be *reserved* before it can be replaced.

• **Forward**
  Forwarding [Seite 1435] passes a work item on to another employee for execution.

• **Resubmit**
  If a user chooses this function, the selected work item is placed in workflow resubmissions [Seite 1443].

• **Display workflow log**
  This function displays the workflow log [Seite 1420].

• **Manage attachments** ... (Functions for attachment management [Seite 1429], dialog work items only)
  • Display attachments
  • Create attachments
  • Change attachments
  • Delete attachments

• **More functions**
  • Set to 'Done' [Seite 1440]
  • Reject execution [Seite 1430]
  • Execute together [Seite 1431]
  • Change priority [Seite 1433]
  • Send mail [Seite 1425]
  • Change work item [Seite 1438]

• **Environment**
  • Display objects in workflow (dialog work items only)
  • Display workflow relationships [Seite 1436]
  • Start Workflow [Seite 1450]
  • Create link [Seite 1434]
Activities

You can access the workflow inbox in the tree on the left in the Business Workplace via Workplace → Inbox.
Work Item Display: Standard View

Use
The objective of the work item display is to display all the information and activities that are relevant to an end user in the environment of the displayed work item in a clear and concise manner. The work item display supports the agent not only in the processing of the current work item but also in the compilation of an activity list, which then functions as the agent's personal worklist.

Integration
A technical work item display is available for work items of all other types. You can make this technical work item display standard for dialog work items as well. The standard view of the work item display can be shown with or without ActiveX. You make this setting for the work item display variant in your personal workflow settings.

Features
The work item display is primarily designed for displaying dialog work items. It contains details about deadlines, statuses, agents, attachments and linked objects for a work item. The work item display also enables an end user - providing they have the relevant authorization - to compile an activity list as their personal worklist.

The work item display has three tab pages (Basic data, Activities, and Available objects).

- Tab page Basic data
- Tab page Activities
- Tab page Available objects
- Customer-defined tab page for work item display

You can define another tab page, which is then displayed as the first tab page when the work item display is called.

Application toolbar functions

- **Execute**
  To be executed, a work item must have either the status ready or the status in process.

  When a dialog work item is executed, the object method to which the single-step task for this work item refers is carried out.

  When missed deadline work items are executed, the most important information on the monitored work item is displayed.

- **Display last message**

  The return code that was returned to the workflow system after the object method was executed can be retrieved for processed work items using the Messages function.

- **Forward**

  Forwarding passes a work item on to another employee for execution.

- **Resubmit**
If a user chooses this function, the selected work item is placed in workflow resubmissions [Seite 1443].

- **Change priority**
  Refer to Changing Priorities [Seite 1433].

- **Change deadlines**
  Refer to Changing Deadlines [Seite 1428].

- **Display/create/change attachments**
  For attachments, refer to Attachment Management [Seite 1429].

- **Reserve** (dialog work items only)
  This reserves a work item for execution by the end user in question. The work item must have the status *ready*. This work item is then no longer visible to the other recipients who could previously see it in status *ready*. The status of the work item changes from *ready* to *reserved*.

- **Replace** (dialog work items only)
  Replacing releases the work item from its reserved status and makes it available again to all of the original recipients. The status of the work item changes from *reserved* back to *ready*. The status of the work item must be *reserved* before it can be replaced.

- **Mail**
  Refer to Send Mail [Seite 1425].

- **Display workflow log**
  This function displays the workflow log [Seite 1420].

- **Display graphical workflow log**
  This function displays the graphical workflow log [Seite 1424].

**Additional functions in the menus**

**Work Item**
- **End resubmission**
  The work item is put back into the workflow inbox. It has the status *reserved*.

- **Create link [Seite 1434]**
- **Reject execution [Seite 1430]**
- **Set to ‘Done’ [Seite 1440]**

**Goto**
- **Methods**
  You can use this function to execute the defined secondary methods [Extern] of the work item displayed from the work item display.

  A prerequisite for this is that the work item represents an activity that is part of a workflow. At least one secondary method must be defined for this activity.
Workflow description

The description text of the superordinate multistep task, the "process description", is displayed.

The work item must be part of a workflow and a description text must be maintained for the workflow.

- Technical work item display [Seite 1418]

Extras

- Displaying tasks

This function can be used to display the definition of the task represented by the work item. Refer to Definition of a Single-Step Task [Seite 1175] and Definition of a Multistep Task [Seite 1194].

- Technical data

Technical data about the work item, for example its ID, its texts or the ID of the superordinate work items are displayed.

- Organizational Assignment [Seite 1397]
- Displaying Agents [Seite 1427]

Activities

You can access the work item display by:

- Selecting a work item in the Business Workplace and choosing 📚.
- Double-clicking one of the entries for a step in the workflow log.
- Selecting an entry that does not represent a workflow in the work item selection [Seite 1490] hit list displayed. (If you choose a workflow, the workflow log is displayed.)
Tab Page Basic Data

Use
The information on this tab page of the work item display [Seite 1411] is mostly self-explanatory. Except for the priority, you cannot make any changes here.

Features

Deadlines
These are the deadlines monitored by the runtime system. (Deadlines that are not set are displayed without a date.)
- **Start by** (latest start)
- **End by** (latest end)
Depending on whether the work item represents a step in the workflow or a task, the deadlines were either specified when this step was defined in the workflow definition or when the task was started online. To display all of the deadlines of the work item, choose Work item → Deadlines.

Further Information
- **Forwarded by**
  If the work item was forwarded to you, the name of the forwarder is entered here.
- **Priority**
  The priority of a work item is derived from the definition of this step in the workflow definition. The priority is used as a sort criterion for positioning the work item in the Business Workplace.
  The priority can be changed here.
- **Status**
  The current processing status is expressed by the work item status [Seite 1373].
- **Creation date** (*created on*) and processing date (*processed from*)
  These are the actual dates and times (when the work item was created and when processing was started).

Using the Messages function, you can display the return codes for processed work items, which were returned to the workflow system after execution of the object method.

Work Item Description
A description of the work item to be executed is provided at the bottom left of this tab page. The task description is entered in the task definition. It is used for information purposes and generally contains instructions and recommendations on processing the work item displayed.

Attachments
The titles of all the attachments added to this work item or, if the work item is part of a workflow, to the preceding work items are shown in the lower right of this tab page.
Tab Page Activities

Use

The Activities (not yet processed) list contains all the activities that are relevant for processing this work item. The tab page Activities is part of the work item display [Seite 1411].

Features

The work item text of the task represented by the work item is generally at the start of the list ("main activity"). Once this activity has been processed (and completion of processing has been confirmed, if necessary), the status of the work item changes to completed. No other actions can then be carried out in the work item display.

Activities

You can extend the activity list and in this way create a worklist. The activities added represent your "personal worklist" as end user (agent).

<table>
<thead>
<tr>
<th>Function</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Done</strong></td>
<td>An activity selected from the list is reported to be done. This functionality cannot be executed on the leading activity (first line) of this work item. The agent is responsible for reporting that an activity is done. The actual processing is not checked.</td>
</tr>
<tr>
<td><strong>Execute</strong></td>
<td>An activity selected from the list is executed. The main activity (first line) of the work item can also be executed via the menu path Work item → Execute.</td>
</tr>
<tr>
<td><strong>Create</strong></td>
<td>Another activity is added to the list. This activity can entail</td>
</tr>
<tr>
<td></td>
<td>• Executing a method on an existing object (created invoice, created material, etc.)</td>
</tr>
<tr>
<td></td>
<td>• Creating a new object (text, etc.)</td>
</tr>
<tr>
<td></td>
<td>When you create an activity, you are given step-by-step support in dialog by a &quot;wizard&quot;.</td>
</tr>
<tr>
<td><strong>Delete</strong></td>
<td>An activity is deleted from the list.</td>
</tr>
</tbody>
</table>

Scrolling in the activity list: The activity list contains extensive information on each activity, which you can view by scrolling to the right. This information includes the following:

- Description
- Creator of the activity with date and time
- Actual agent of the activity with date and time
- Method and object type
Tab Page Available Objects

Use
All objects that are related to the work item are displayed in the list of available objects.
These objects are stored in container elements of the task container as object references. You can display these objects or add new objects to the container element.
There are:

- **Ad hoc objects**
  Objects added to a work item in this or one of the preceding steps of the workflow
  (container element _Adhoc_Objects of the task container)

- **Attachment objects**
  Documents added to a work item in this or one of the preceding steps of the workflow
  (container element _Attach_Objects of the task container).

- **Process objects**
  - The object currently being processed (container element _WI_Object_ID of the task container)
  - The object added for grouping purposes (container element _WI_Group_ID of the task container)
  - Objects that are referenced in other elements of the task container

The tab page *Available objects* is part of the *work item display [Seite 1411]*.

Features

Displaying objects with their default attributes
Each object referenced in the work item container is displayed with its default attribute [Extern]. The default method [Extern] of each object can be executed upon request. If no default attribute was defined for the object type, the key fields of the object are displayed.

Adding objects
You can extend and process the list of objects. The main purpose of this is to make the relevant information available to the agents of the subsequent steps in the workflow as well.
Only object types [Extern] that support the IFFIND interface can be selected. You identify an actual object [Extern] of this type by specifying its key fields [Extern].

Activities
To execute the functions displayed, proceed as follows:

<table>
<thead>
<tr>
<th>Function</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Create</td>
<td>An object is added to the list. When you add an object, you are given step-by-step support in dialog by a &quot;wizard&quot;.</td>
</tr>
<tr>
<td>Display</td>
<td>The default method of an object selected from the list is executed.</td>
</tr>
<tr>
<td>Remove</td>
<td>An object is deleted from the list.</td>
</tr>
</tbody>
</table>
Work Item Display: Technical View

Use
The technical work item display shows all information from the environment of the work item displayed. The technical work item display is intended in particular for workflow system administrators.

The data is always formatted and displayed in a similar way irrespective of the work item type. You should note the work item type shown on the screen.

Features
General
Work item information
- Information derived from the definition of the relevant task: Work item type, work item text, way in which processing is completed.
- Status of the work item as current processing information
- Actual agent (available after dialog work items have been executed).

In the case of processed work items, the Messages function can be used to display the return codes returned to the workflow system after execution of the object method.

Deadline data
Here you can find the current dates/times [Extern] (creation date/time of work item and date/time processing started and ended) as well as the deadlines that are monitored by the runtime system (requested and latest start and end deadlines).

A monitored deadline shown with red is in the past. If the symbol is displayed as well, a defined action was triggered.

Description
This description displayed here is taken from the task definition. It is used to inform the selected agents and contains instructions and recommendations on processing the work item displayed.

If you have selected a work item using the work item selection [Seite 1490] or the workflow outbox rather than your workflow inbox, you can execute it provided you are one of the possible agents.

You do not need to be one of the recipients in this case.

Additional functions
As well as the functions available in the standard view of the work item display [Seite 1411], the following additional functions are also available:

- Display/create/delete object
  Each object referenced in the work item container is displayed with its default attribute [Extern]. The default method [Extern] of each object can be executed upon request. If no default attribute is defined for the object type, the key fields of the object are displayed instead.
You can extend and process the list of objects. The main purpose of this is to make the relevant information available to the agents of the subsequent steps in the workflow as well.

Only object types [Extern] that support the IFFIND interface can be selected. You identify an actual object [Extern] of this type by specifying its key fields [Extern].

- **Type-specific data**
  
  Type-specific data only applies to work queue and wait step work items.

  For work queue work items (type A), the objects and tasks contained in the work queue are listed.

  For wait step work items (type E), the number of events expected (information taken from the workflow definition) and the number of events that have already occurred are specified.

- **Execute for testing**
  
  In order to make it possible to check the runtime system's program execution (work item manager and workflow manager) in debugging mode after execution of an object method, internal communication within the workflow system cannot take place asynchronously. To achieve this, execute the work item for test purposes. Enter /h in the command field first to go to the debugging mode.

- **Monitored work item**
  
  The work item (type F or W) whose deadlines or completion are monitored by the workflow system is displayed.

  This function is only possible and active from missed deadline work items [Extern]. This function displays the work item. Full display functionality, including all navigation and change options, is also available here for this work item.

- **Instance linkage**
  
  You go to the relevant line of the instance linkage table, in which the expected event (identified by object type and event ID) and the object (identified by object reference) are specified.

  This function is only possible and active for those work items that wait for an event that completes them. This may apply to dialog work items (type W) and background work items (type B). This does apply to wait step work items (type E).

- **Container**
  
  This function displays the content of the task container.

- **Change work item [Seite 1438]**

**Activities**

You can go to the technical view of the work item display by choosing Goto → Technical work item display in the work item display or by having this display variant as a presetting in your personal workflow settings [Seite 1395].
Workflow Log: Standard View

Use
The workflow log [Extern] enables you to format all the information that is generated or collected during the execution of one active business process in one central location. The standard view described here is intended, in particular, for agents and process controllers who want to get an overview of the steps processed so far.

Prerequisites
To use the view described here, you must have set a view with ActiveX in your personal workflow settings [Seite 1395].

Integration
Other options for displaying the workflow log are:
- Workflow Log: Technical View [Seite 1422]
- Graphical Workflow Log [Seite 1424]

Features

Application toolbar functions
The following functions are available on the application toolbar:
- Update
- List with technical details
  This function takes you to the technical view of the workflow log.
- Graphical workflow log
- Personal workflow settings
  You can use this function to maintain your personal workflow settings.

Tab pages
The system shows the various views on the workflow log on different tab pages.

Tab page Workflow Chronicle ( \( \square \), what was processed when?)
The tab page Workflow Chronicle shows a hierarchical display of all steps in the workflow, which have been processed so far or are due to be processed. If the workflow has a subworkflow structure, the subworkflows are also displayed.
The Details function (\( \square \) symbol) lists the following information about each step in the lower part of the screen:
- Who carried out what detailed actions for these work items and with what results.
- When these actions were carried out.
- Which objects were involved.
The Agents function (\( \square \) symbol) displays the selected and possible agents of a step.
The Graphic function (\( \square \) symbol) displays the graphical workflow log.
**Workflow Log: Standard View**

**Tab page Workflow Agents (跑了，谁在处理什么？)**
The tab page *Workflow Agents* shows the employees involved in this workflow up to now. The following is displayed for each employee:

- What action was carried out in what step.
- When this action was carried out.
- Which objects were involved.

This view shows how an employee was involved in the execution of a workflow.

**Tab page Workflow Objects (处理了什么？)**
The tab page *Workflow Objects* lists the objects related to the workflow or addressed up to now in the execution of the workflow. These objects include:

- The “leading” object of the workflow.
- Any attachments and objects added in the individual steps of the workflow.

The following is displayed for each object:

- Who carried out what detailed action for what task.
- When this action was carried out.

This view shows what information was generated and processed, and how.

**Information at the click of a mouse**
You can view all the information provided in the workflow log using the mouse.
You can also go to the work item display [Seite 1411] for each dialog step. You can display address data for agents as well as the contents of work item attachments or the result of actions that have been executed.

**Activities**
You can access the workflow log from the work item display or the Business Workplace [Seite 1368] via the icon.
Workflow Log: Technical View

Use
The workflow log [Extern] enables you to format all the information that is generated or collected during the execution of one active business process in one central location.
The view described here, in particular the technical view described below, is intended for workflow system administrators.

Integration
Other options for displaying the workflow log are:
- Workflow Log: Standard View [Seite 1420]
- Graphical Workflow Log [Seite 1424]

Prerequisites
To use the view described here, you must have set the view without ActiveX in your personal workflow settings [Seite 1395].
If you have chosen technical view in your settings, the standard view with technical details is displayed (see below). Otherwise, the two workflow log views are identical.

Features
The system uses a two-level, hierarchical list from the ABAP List Viewer [Extern] to display the various workflow log views. You can adapt the appearance of the list to suit your requirements using display variants.
By single-clicking on an entry or a symbol, you can then branch to the workflow container or the work item display [Seite 1411], for example.
The following views are available:

- **Workflow chronicle**
  The first level of the Workflow chronicle view shows all the workflow steps that have already been processed or are currently pending. If the workflow has a subworkflow structure, the subworkflows are also displayed.
  The second level (detail view) shows the following for each step:
  - Who carried out what detailed actions for these work items and with what results.
  - When these actions were carried out.
  - Which objects were involved.
  This view is used to determine what activities were carried out in a workflow and in what order.

- **Workflow agents**
  The first level of the Workflow agents view shows the employees involved in this workflow up to now.
  The second level (detail view) shows the following for each employee:
  - What action was carried out in what step.
  - When this action was carried out.
**Workflow Log: Technical View**

- Which objects were involved.
  This view shows how an employee was involved in the execution of a workflow.

- **Workflow objects**
  The first level of the Workflow objects view lists the objects that are related to the workflow or that have been addressed up to now during execution of the workflow. These objects include:
  - The “leading” object of the workflow.
  - Attachments and objects added in the individual workflow steps.
  The second level shows the following for each object:
  - Who carried out what detailed action for what task.
  - When this action was carried out.
  This view shows what information was generated and processed, and how.

- **... With technical details (technical view)**
  The technical view shows technical control information for execution of a workflow, as required by workflow administrators, for instance.
  Based on the workflow chronicle, the technical view shows technical nodes and control structures, and makes additional data available, such as container elements ( ), employee data ( ), and workflow data ( ).
  The status of the work item is also displayed.

- **... With subworkflow structure**
  Here you can choose whether or not to display any subworkflows and their structure.

- **... With error indicators**
  If you activate the function View with error indicators, errors are marked in the log with the symbol. The standard indicator is .

**Activities**
You can access the workflow log from the workflow inbox of the Business Workplace [Seite 1368] via the symbol, or using the context menu (click the right-hand mouse button).
You can maintain the individual views (such as the technical view) within the workflow log via Views or Views → Other views.
Graphical Workflow Log

Use
The workflow log [Extern] enables you to format all the information that is generated or collected during the execution of one active business process in one central location.

Integration
Other options for displaying the workflow log are:
- Workflow Log: Technical View [Seite 1422]
- Workflow Log: Standard View [Seite 1420]

Features
The graphical workflow log adds to the textual information. The workflow steps already processed are shown with 🔄 in a graphical representation of the workflow definition. You can see at a glance which “route” a workflow instance has taken and which activities are processed in parallel to your own within a business process. Unlike the text version of the workflow log, the graphical workflow log also shows the subsequent flow of a workflow instance.

The screen of the graphical workflow log is split into the workflow area (left) and the overview area (right).

The following functions are available:
- 🔄 Refresh
  Refreshes the display.
- 🔄 Display node
  If you select an executed step and choose this function, the system goes to the technical work item display.
  If the step has not yet been executed, the relevant activity is displayed.
- 🔄 Align...
  The workflow is centered within the workflow area.
- 🔄 Zoom in
  The size of the workflow is increased within the workflow area.
- 🔄 Zoom out
  The size of the workflow is decreased within the workflow area.
- 🔄 Complete view
  The whole of the workflow is shown within the workflow area.

For other functions, refer to the documentation on the Workflow Builder [Seite 1002].

Activities
You can call the graphical workflow log from within the workflow log [Seite 1420] via 🔄.
Send Mail

Purpose
You want to send a mail to another user in connection with a work item that requires processing. The work item being processed is therefore also to be made accessible to this user for information purposes. The *Send mail* function enables you to send mails to any recipients with a text which can be entered freely. These mails are automatically linked to your work item in such a way that when the mail is executed by the recipient the work item is displayed.

Process Flow

Procedure as sender of mail
16. In the Business Workplace's workflow inbox, position the cursor on a work item and choose *Other functions* \(\rightarrow\) *Send mail*.
   
   If you are already in the work item display, choose *Work item* \(\rightarrow\) *Send mail*.
17. Enter the text of the mail.
18. Send the mail to any recipients who are available as mail addressees
   
   The mail text is added to the work item as an attachment. The work item remains in your Business Workplace with the same status.

You can still complete the work item despite the fact that you have sent a mail. Since this makes the mail which has been sent superfluous, the recipient of the mail receives a second mail informing him/her that the (first) mail no longer needs to be dealt with.

Reply ends resubmission
Optional: If you do not want to process the work item until the mail has been replied to, define a resubmission for the work item. To do this, position the cursor on the entry, choose *Resubmit* and enter a date in the future.

The work item is resubmitted irrespective of this date and appears in your workflow inbox when the reply to the mail is received.

Procedure as recipient of a mail
You receive a mail that can be executed.
21. Read the mail.
22. Execute the mail, if applicable. The work item referred to by the sender in their mail is displayed. For further information on the object to be processed, choose *Goto* \(\rightarrow\) *Object* to display the default attribute or execute the default method (generally *Display*) for the processed object.

   You can execute this work item if you are one of the possible agents of this task.
23. Reply to the mail. To do this, choose *Document* \(\rightarrow\) *Reply* or *Document* \(\rightarrow\) *Reply w/reference* when the mail is displayed. Then enter your reply and save your entries.
24. Choose *Document* \(\rightarrow\) *Send*. On the send screen, the work item to which the mail referred is entered as the "recipient". Do not change this entry. Send the reply to the proposed recipient.
Result
The reply sent ends a resubmission which may have been defined for the recipient work item. This work item is appended as an (additional) attachment and can be read as such by the original user and taken into account.
Display Agents

Use
This function displays information about the agents of the work item.

Features
The system displays the following agents for the work item:
- Recipient [Extern]
- Possible agents [Extern]
- Excluded agents [Extern]
You can choose a compressed display - containing only the user names (users only function) - or a complete display with additional information on the relationships used by the system to determine the agents (overall view function). You can also display the organizational assignment of each user. Note that this information is only available for work items of types for which an agent is logical and necessary.

Activities
You can display the agents from the work item display [Seite 1411] by choosing Goto → Agents → ...
Changing Deadlines

Use
You use this function if you want to change the deadlines for a work item at workflow runtime. This dialog box displays all the deadlines of work item processing. Please also refer to Current Dates/Times of a Work Item [Extern].

Features
Depending on whether the work item represents a step in the workflow or a single-step task, the deadlines were either specified when this step was defined in the workflow definition or when the single-step task was started in dialog. The dialog box is split into the following sections:

Deadlines
- Start by (latest start)
- End by (latest end)

Planned deadlines
- Start by (requested start)
- End by (requested end)

A monitored deadline shown in blue is in the past. If ( ) is also displayed, the appropriate action has been initiated. This generally involves informing the deadline recipient.

Actual dates/times
- Created on: Creation date/time of the work item
  (Technically: The work item is created with the status ready or - if the requested start date has not yet been reached - with the status waiting).
- Processed from: Start of processing
  (Technically: Transition of the work item to the status in process).
- Completed on: End of processing or date when set to ‘done’.
  (Technically: Transition of the work item to the status completed.)

Activities
To execute this function, choose in the work item display [Seite 1411].
Attachment Management

Use
One or more attachments can be assigned to each work item that appears in the Business Workplace's workflow inbox.
Attachments are documents written either with a SAPscript editor (document classes RAW, SCR) or with a PC application (document classes DOC, URL, PPT, XLS, PDF, ...) and then imported. You can enter new documents as attachments or create attachments from existing files.

Features

General
The attachment is automatically

- Added to the work item container
- Added to the container of the superordinate workflow
- Added to the containers of the subsequent work items in the workflow

You can define default documents for the individual document classes. For further information, refer to Default Documents [Extern].

Attachments can be displayed by the recipients of the subsequent steps. But they cannot be changed and, therefore, have a document character.

A superior who is to make a decision on releasing a budget can enter an attachment justifying their decision. The selected agents of the subsequent steps can display this attachment.

If a work item has attachments, this is indicated by a symbol in the Attachments column in the Business Workplace. You can also execute the function for processing an attachment by double-clicking in this column (column header AT). If an attachment already exists, it is displayed.

If a work item has attachments, this is indicated by the symbol in the work item display.

Functions on attachments

<table>
<thead>
<tr>
<th>Display</th>
<th>Displays the attachment selected in the dialog box Existing attachments.</th>
</tr>
</thead>
<tbody>
<tr>
<td>Create</td>
<td>Creates a new attachment. If you want to import an existing document from your PC to the attachment, choose Import.</td>
</tr>
<tr>
<td>Change</td>
<td>Displays the selection of Changeable attachments.</td>
</tr>
<tr>
<td>Delete</td>
<td>Displays the selection of Deletable attachments.</td>
</tr>
</tbody>
</table>

Activities
You can access attachment maintenance in the Business Workplace by choosing or the relevant context menu (right-hand mouse button).
Reject Execution

Use
You can use this function if you need to reject execution of a work item for business or technical reasons. This function is only available for work items of type W.

The table entry to be processed already exists or the material whose master data is to be changed is no longer used.

Prerequisites
This function is only available if the property processing rejectable has been selected for the related activity in the workflow definition.

Features
Processing of the work item is terminated with the reject execution function. The subsequent steps defined in the workflow definition are executed. Do not use this function if you do not want to or cannot process the work item for personal reasons (not responsible, not competent). In this case, replace the reserved work item or forward it.

Activities
You can access this function in the Business Workplace [Seite 1368] by choosing Other functions or the work item display (Work item → Other functions → Reject processing).
Execute Together

Use
You can select various work items and then execute them together. This function is only available for work items of type W.

The work items do not necessarily have to belong to the same workflow.

Prerequisites
The work items must refer to the same single-step task.
For information about other prerequisites for this function, in particular with regard to the underlying method, refer to Creating an "Execute Together" Method [Extern].

Features
Only the first of the selected work items is proposed for execution. The entries you make here also apply to the other work items to be executed together.

Activities
Select the work items that you want to execute together. Then choose Execute together via Other functions... in the Business Workplace [Seite 1368].
Grouping According to Sort Key

Use
Each work item carries the two container elements _WI_Object_ID and _WI_Group_ID in its container. Both elements have been defined to hold an object reference.

- The container element _WI_Object_ID automatically contains the reference to the object to be processed in the work item
- The container element _WI_Group_ID can contain an object reference, which must be assigned to this container element in a binding or via initial value assignment.

The object reference assigned via _WI_Group_ID is generally not identical to _WI_Object_ID, but is derived from the work item execution environment. It is used to group work items that refer to different objects or object types but are nevertheless connected.

Features

Display default attributes in workflow inbox
The default attributes [Extern] of the objects referenced in _WI_Object_ID and _WI_Group_ID are available in the Business Workplace's workflow inbox [Seite 1408] under the column headers Object or Group for grouping, sorting and filtering purposes.

Material master data is processed in several instances of a workflow. The container element _WI_Object_ID in the containers of the individual work items therefore always contains the reference to the object of type BUS1001 (material) to be processed.

The container element _WI_Group_ID is assigned the expression &Material.Labor& (laboratory/drawing office for material as object reference) in the relevant steps of the workflow definition. The default attribute of the laboratory is therefore available as a group.

Default methods
The default method [Extern] of the objects referenced in _WI_Object_ID and _WI_Group_ID is executed by double-clicking in the Object or Group column.
Change Priority

Use
The priority of a work item is a measure of its urgency. It can be used as a sort criterion for organizing the workflow inbox. This function is only available for dialog work items.

Features
End users can display and change the priority of a work item. The priority is between 1 (highest) and 9 (lowest). The change in priority can also be passed on automatically to the superordinate workflow and then to all work items created subsequently. The Pass on priority to subsequent steps indicator must be selected for this to take effect.

If a higher priority (lower number) is defined in the workflow definition for one of the subsequent steps, it is not changed.

Activities
You can access the Change priority function in the Business Workplace by choosing Other functions...
Create Link

Use
By storing links to work items in folders [Extern], you can organize your work effectively using a personal folder hierarchy.

Features
The work item is added to a personal or shared folder as a link. This work item can be displayed from this folder and executed by its recipients. Work items can have an unlimited number of links.

Activities
You can access this function in the Business Workplace [Seite 1368] by choosing Environment...
Forward Work Item

Use
Forwarding passes a work item on to another user for execution.

Integration
The range of users to whom a work item can be forwarded is determined from the task definition as follows:

<table>
<thead>
<tr>
<th>Task definition:</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Task is a general task [Extern]</td>
<td>Work item may be forwarded to all users.</td>
</tr>
<tr>
<td>The indicator General forwarding allowed is set for the task</td>
<td>Work item may be forwarded to all users.</td>
</tr>
<tr>
<td>The indicator General forwarding not allowed is set for the task</td>
<td>Work item may only be forwarded to the recipients [Extern] of the task.</td>
</tr>
</tbody>
</table>

Features
The user has the following search options to select the new recipient:

- The F4 input help can be used to determine the new recipient by performing a generic search for name components.
- The structure search can be used to determine the new recipient using the graphical display of the organizational plan.

The new recipient does not have to be an actual user. (The work item can also be forwarded to an organizational unit or a job.)

Activities
You can access this function in the Business Workplace [Seite 1368] by choosing .
Workflow Relationships

Use
You can use this function to establish the work items in which a particular object is being processed. The following work item types are taken into account:
- Dialog work items (type W)
- Background work items (type B)
- Workflow work items (type F)

The list displayed is purely runtime information.

Prerequisites

Authorization
You require appropriate authorization for the functions shown. This is an authorization based on the authorization object S_PROGRAM for executing a program for authorization group 'SWI_OA'.

Features
You can call the function for displaying linked workflows from several starting points.

Calling via SAP Easy Access
Two functions are available via SAP Easy Access:
- **Workflows for object**
  This function displays all workflows linked to a particular object (for example a specific notification of absence). You must specify this object beforehand.
- **Workflows for object type**
  This function displays all workflows linked to a particular object type (for example FORMABSENC). You must specify this object type beforehand.

  You can only specify objects whose object type definition supports the interface IFFIND.

Calling from the Business Workplace
In this case, you do not specify the object. The workflow relationships are displayed for the object processed in the selected work item.

Calling as a generic object service
In this case, you do not specify the object. The workflow relationships are displayed for the object being processed.

What is displayed?
The display of the workflow relationships has two parts. The upper part displays the workflows identified for the object or object type. The lower part displays data on the currently selected work item in the form of a simplified workflow log:
- **Steps so far**
Workflow Relationships

The workflow steps processed so far are listed under Step name. The steps are linked to the work item preview of the relevant work item. The current agents of the work item are listed under Agents. Click once to go into detailed display of the user data.

- Information objects addressed so far
  All the objects and attachments belonging to the workflow are displayed here. Click once to display the information objects.

Activities

<table>
<thead>
<tr>
<th>Calling...</th>
<th>Path</th>
</tr>
</thead>
<tbody>
<tr>
<td>... Via SAP Easy Access</td>
<td>Tools → Business Workflow → Development →</td>
</tr>
<tr>
<td></td>
<td>Runtime Tools → Workflows For Object</td>
</tr>
<tr>
<td>... From the Business Workplace [Seite 1368]</td>
<td>Display workflow relationships</td>
</tr>
<tr>
<td>... As a generic object service</td>
<td>System → Workflow → Workflow overview</td>
</tr>
</tbody>
</table>
Change Work Item

Use
The function enables you to change information and data associated with the selected work item.

Features

Changes in the initial screen
In general, the changes that are allowed depend on the type and status of the work item. The following changes are possible:

- **Work item text**
- **Priority**
- **Deadline data**

You can set and change deadline data for the start and end deadlines of the work item. You can only make specifications for deadlines that are still being monitored and are in the future. For example, dialog work items with the *ready* status can no longer be assigned a requested start.

Manual intervention into the processing of work items
If you are an administrator with the appropriate authorization, you can intervene manually into the processing and therefore into the process flow of a workflow. This is particularly helpful for dealing with errors. The following changes are possible:

- **Set to ‘ready’ manually**
  The work item status is changed from *waiting* to *ready*.
  The work item therefore appears in the workflow inboxes of the selected agents.

- **Complete manually**
  The work item status is changed to *completed*.
  In the case of dialog work items, this status change (together with the current work item container) is passed on to the workflow system for evaluation. If the object method to be executed is a synchronous method with a result, the possible result values are displayed for selection.

- **Replace manually**
  The status of the work item is reset to *ready*.
  The work item is therefore displayed again in the workflow inboxes of all selected agents.

- **Restart after error**
  Work items and, in particular, workflows with errors can be restarted after the errors have been corrected.
  The information written by the workflow system in a log when the error occurred is now taken into account.

- **Delete logically**
  The status of the work item is changed to *logically deleted*. 
In the case of dialog work items, this status change is passed on to the workflow system.

**Locking/unlocking execution**

An administrator can lock or unlock the current work item for execution. To do this, you select the appropriate function from the *Edit* menu.

**Deleting deadlines**

To delete a deadline you have changed or entered and reset it to its initial values, choose *Edit → Delete deadline*.

**Changing work item containers**

Choose *Edit → Change container* to go to the editor for changing the work item container. The contents of the container for the relevant work item are displayed. You can change the current, runtime-specific data for the particular work item. The container may still have elements that do not currently have a value. These elements are hidden as standard. The presence of elements that are not displayed is shown by an indicator. To display these elements, choose *Edit → Show elements*. To add more lines to container elements defined as multiline, choose *Edit → Additional line*.

**Activities**

You can access this function in the Business Workplace [Seite 1368] by choosing Other functions, the relevant context menu or the technical work item display [Seite 1418].
Set Work Item to Done

Use
This function is used by the recipient of a work item to confirm explicitly that processing of this work item has been completed. As long as explicit confirmation has not been provided, the work item has *executed* status and can be executed again or forwarded.

The status of the work item changes from *executed* → *completed*.

Prerequisites
This function is only possible on a work item if a setting was made in the definition of the associated single-step task stating that the end of processing must be confirmed explicitly.

Activities
You use this Business Workplace function if the status of the work item has been changed to *executed* by a terminating event.
In general, however, the work item is not set to *done* via the Business Workplace. This function is usually provided as a dialog box directly after a work item has been processed.
*Missed deadline work items [Extern]* must always be confirmed explicitly. After execution, they remain in the status *in process* until they are set to *done*.
You can access this function in the Business Workplace [Seite 1368] by choosing Other functions.
Workflow Outbox

Use
The workflow outbox lists the workflows started and the work items forwarded and executed by the current user.

Integration
As is the case for the workflow resubmissions [Seite 1443] and the workflow inbox [Seite 1408], the workflow outbox is an integral part of the Business Workplace.

Features
Views in the workflow outbox

Started workflows
This view shows work items for the tasks started by you as a user in dialog or by a triggering event whose event container contains your user name as _Evt_Creator.

Work items executed by me
The work items executed by you are displayed in this view.

Forwarded work items
The work items forwarded by you are displayed in this view.

Workflow Functions
The screen area in which the worklist is displayed has a toolbar containing the workflow functions and functions for configuring the list. For information about these functions, refer to the documentation on the ABAP List Viewer [Extern].

The following functions are available for the different workflow outbox views:

- **Update**
  You use this function to update the workflow outbox.

- **Display work item** (not in Started workflows view)
  This function goes directly to the work item display [Seite 1411]

- **Change agent**
  You use this function to perform an ad-hoc agent assignment [Seite 1097], [Seite 1411]

- **Display workflow log**
  This function displays the workflow log [Seite 1420].

- **Display attachments** (refer to attachment management [Seite 1429])

- **Environment**
  - Display objects in workflow (dialog work items only)
  - Display workflow relationships [Seite 1436]
  - Start Workflow [Seite 1450]
Workflow Outbox

- Create link [Seite 1434]
- Change selection period
  - Today only
  - Last 7 days
  - Last 30 days
  - User-defined ...

The functions described above can be called using the relevant context menu.

Activities
You can access the workflow outbox in the overview tree in the Business Workplace [Seite 1368] via Workplace → Outbox.
Workflow Resubmissions

Use
The workflow resubmissions function can be used to resubmit work items for processing at a later point in time or periodically and to display these work items in a list. Technically, the following applies to work items which appear in your workflow resubmission folder: the work item status is set to waiting, your name is entered as the actual agent, and the requested start date is set to the resubmission date.

Integration
As is the case for the workflow inbox and the workflow outbox, the workflow resubmissions is an integral part of the Business Workplace.

Features

Workflow Functions
The screen area in which the worklist is displayed has a toolbar containing the workflow functions and functions for configuring the list. For information about these functions, refer to the documentation on the ABAP List Viewer.

The following functions are available in the workflow resubmissions:

- **Update**
  You use this function to update the workflow resubmissions.

- **Display work item**
  This function goes directly to the work item display.

- **Display workflow log**
  This function displays the workflow log.

- **Display attachments**
  This function is only available if the work item has attachments. Refer to attachment management.

- **End resubmission**
  The work item is put back into the workflow inbox. It has the status reserved.

- **Environment**
  - Display objects in workflow (dialog work items only)
  - Display workflow relationships
  - Start Workflow
  - Create link

The functions described above can be called using the relevant context menu.

Activities
You can access the workflow resubmissions in the overview tree in the Business Workplace via Workplace → Resubmissions.
You are asked for a date for resubmission of the work item. The work item then disappears automatically from the workflow inbox and is not displayed there again until the date specified.
Work Item Preview

Use
The work item preview is in the bottom right corner of the Business Workplace screen and provides a preview of the properties of a work item. All the information on a work item is displayed if you choose the function Display work item in the worklist of the workflow inbox [Seite 1408] or the workflow resubmissions [Seite 1443]. This function goes directly to the work item display [Seite 1411].
A concise view of the workflow log [Seite 1420] is offered in the workflow outbox [Seite 1441] when the started workflows are displayed.

Integration
The work item preview is part of the Business Workplace [Seite 1368].

Features

Workflow functions
The description of the work item is displayed in the work item preview. It is also possible to access objects and attachments of the work item directly from the preview.
The concise view of the workflow log enables you to access processed steps (individual work items), their agents and information objects addressed so far.
For more information about the concise view of the workflow log, please refer to Workflow Relationships [Seite 1436].

Tips & tricks
"Tips & tricks" are displayed when you position the mouse pointer on the title of a "tips & tricks" entry. Whenever the work item preview is updated, a new title is offered.
You can activate/deactivate "tips & tricks" in your personal workflow settings [Seite 1395] (Settings → Workflow settings → Personal settings).

User exit
The work item preview can be adapted to customer-specific requirements using a user exit [Extern].

Activities
The work item preview is activated/updated whenever you select a work item in your worklist.
Workflow Toolbox

Use
SAP Business Workflow's Workflow Toolbox enables the user to access workflow functions even during a workflow-driven application transaction. The following functions are now available even during method execution:

- Display work item text
- Display task text
- Availability of attachments and objects as links
- Create/display/change/delete attachments
- Display possible agents
- Display workflow log
- Forward
- Resubmit
- Send mail

The toolbox is easy to activate from the application side in its dialog transactions. The full functionality of the toolbox is encapsulated in class CL_WAPI_WF_TOOLBOX.

The workflow WS70000651 Demo for WI execution with WF toolbox can be started by any user with transaction SWUI_DEMO, and generates work items in the Business Workplace. The toolbox demonstrates the execution of these work items in its various forms. For further information, refer to Demo and Test Workflows [Seite 1499].

Features

Class CL_WAPI_WF_TOOLBOX
The class CL_WAPI_WF_TOOLBOX encapsulates the full functionality of the toolbox. An instance of this class is created directly before the CALL SCREEN of the application screen. This instance provides all the GUI objects it requires and manages them as well.

Reusing worklist client functionality

Work item preview
The toolbox uses the work item preview [Seite 1445] in the Business Workplace to display work item text, task description and links to attachments and other objects.

Workflow log
The first tab page of the ActiveX version of the workflow log [Seite 1420] is used again to display the complete workflows already processed.
Various Workflow Toolbox modes

Toolbar only (embedded)
If the toolbox is used in its simplest mode C_MODE_INPLACEWITHOUT_INFO, it requires a custom container of approximately 1 x 4 cm on the screen. The work item preview and workflow log are displayed in a separate new screen (modeless). A pushbutton is available to the user for this.

Toolbar and information (embedded)
If the toolbox is used in mode C_MODE_INPLACE_WITH_INFO, it is displayed in two parts on the screen: The toolbar is displayed above the preview initially. The user can switch between preview and log in the lower part.

Toolbar and information (modeless)
If the toolbox is used in mode C_MODE_MODELESS, it is displayed in two parts in a modeless window: The toolbar is displayed above the preview initially. The user can switch between preview and log in the lower part.

Menu button only (embedded)
If the toolbox is used in mode C_MODE_INPLACE_BUTTON_ONLY, it requires a custom container of at least 1 x 1 cm on the screen, depending on whether just an icon or an icon with text is to be displayed. The toolbox opens a modeless window to display work item preview and workflow log when the user chooses the relevant menu option.
Starting Tasks and Workflows

Use
There are various options for starting tasks and workflows. You must be aware of these options when defining a task or workflow.

Integration
A started workflow is represented by a workflow work item [Extern], and a started task by a dialog work item [Extern] or a background work item [Extern].

Features
Tasks and workflows can be started in the following ways.

By an event
The system starts the task or workflow when a triggering event defined for this purpose occurs. The following requirements must be met:

- The event must be defined as a triggering event in the definition.
- The event receiver linkage must be activated.

The linkage may be deactivated by the system if the start was unsuccessful. If this occurs, try to analyze and resolve the error by simulating an event [Seite 1363].

To use data stored with the event in the execution, a binding can be defined from the event container to the workflow container or task container.

A task or workflow can have several triggering events. If one of the triggering events is generated in the system, the relevant execution is started.

Triggering events are only required for a task if the task is to be executed as a single step in response to an event. If you only want to use the task as a step in a workflow, the task does not require any triggering events.

For further information, refer to Creating Triggering Events [Seite 1206].

You can also start a task or workflow defined with a triggering event on a test basis using this event. For further information, refer to Creation of Events for Test Purposes [Seite 1365].

Directly in dialog
You start the task or workflow yourself. For further information, refer to:

- Starting Workflows in Dialog [Seite 1450] for the standard environment at runtime and starting workflows with reference to an application object.

- Starting Workflows (Test Environment) [Seite 1452] for a test environment that allows tasks and workflows to be started.

- Definition of Start Transactions [Seite 1455] for workflows.

A task or workflow started by a triggering event is normally provided with input data from the event container. If you start a task or workflow of this kind in dialog, you
must enter the input parameters manually. Therefore, you will usually only start a task or workflow with a triggering event in dialog for test purposes.

From a message long text
This function is only available with messages for which an appropriate assignment is maintained. A user can start the relevant workflow when the message is triggered. For further information, refer to Starting Workflows or Tasks from Message Long Texts [Seite 1456].

By a workflow
The system starts a task or workflow if the task or workflow is integrated in another workflow as a step. If the task is executed in dialog, possible agents [Extern] must be assigned to it, or it must be classified as a general task.

To execute a task as a step in a workflow, the task does not require triggering events.

For further information, refer to Maintenance of Activities [Seite 1024].
Starting Workflows in Dialog

Use
This function can be used to start the workflows you have created in dialog. "In dialog" means that the workflow is not started by a triggering event but manually. It is irrelevant whether triggering events are defined for the workflow. For further information, refer to Advancing With Immediate Dialog [Seite 1453].

Prerequisites
You can only start tasks in dialog if you are entered as a possible agent [Extern] for them.

Features

Starting workflows
The screen for starting workflows has two parts. The workflows you have defined are listed on the left. The description of the workflow selected on the left is displayed on the right.

You can use the standard toolbar to go to the workflow outbox [Seite 1441] and the Business Workplace [Seite 1368].

You can start a workflow by selecting it and then choosing either\[Image\] in the left screen area or the Start function in the right screen area.

Starting with details
The function Start with details can be used to add information such as attachments, notes or deadlines to the workflow via three tab pages. The attachments and notes are available to all agents of the workflow.

- Tab page Properties
  You can maintain the priority of the workflow, its latest end and its requested start here. If you set the indicator Create flag for started workflow in inbox, you get a work item in your inbox when the workflow is started, which calls the relevant workflow log during execution. The system deletes this flag automatically when the workflow is terminated.

- Tab page Note at start
  The note that you create here is added to the workflow as an attachment.

- Tab page Attachments
  Here you can create, display, import and delete attachments for the workflow.

  You start the workflow with the function Start workflow.

Starting via start transaction
If a start transaction [Seite 1455] is defined for the workflow, the defined transaction is started instead of detail maintenance.
As with detail maintenance, you can change the priority, add attachments and maintain deadline data via .

In contrast to detail maintenance, you cannot create any new attachments. However, it is possible to create ad hoc objects (in contrast to attachments, these are assigned to the container element _Adhoc_Objects).

Make the entries required in the input fields you have created. This assigns values to the import elements of the interface of the workflow to be started. Use the F4 input help if required.
Starting workflows with object references

Starting a workflow with an object reference allows you to extend a process at runtime and initiate additional subsequent activities on an ad hoc and flexible basis. This function is not available everywhere. The function assumes that the system recognizes as a reference object a particular application object (for example document, invoice, item of master data) that you are processing. The process flow is as follows:

- The system establishes a reference object [Extern] from the current call context.
  
  The current call context is, for example, a work item selected in the workflow inbox of the Business Workplace. This work item can then contain the notification of absence xy as a reference object.

- The system establishes the object type for this reference object.

- The system displays all workflows that refer to this object type so you can make your selection. This uses the same function as Starting workflows (see above). If there is no appropriate workflow, you can create a new workflow ad hoc. For further information, refer to Defining Workflows Ad Hoc [Seite 1098].

- The system starts the selected workflow and passes the reference object to the task.

Activities

The function Start workflow can be accessed via Office → Start workflow or via Tools → Business Workflow → Development → Runtime tools → Start workflow.

The function Start workflows with object reference is available within the Business Workplace in the workflow inbox, the workflow outbox and the workflow resubmissions under Environment.
Starting Workflows (Test Environment)

Use
You should only use this environment for starting tasks in a development system.
Here you can start all tasks and workflows, even if you are not one of the possible agents.

Features
In the Task field you can enter the ID of the task to be started. This ID consists of two characters for the task type and an eight-digit number. The task type is identified by:
- T (for customer tasks)/TS (for standard tasks)
- WF (for workflow tasks)/WS (for workflow templates)

If you enter a customer task, there must be a blank between the T and the 8-digit number.

To ensure that the respective current organizational plan is taken into account when the task is started, refresh the organizational environment.

The function Input data can be used to enter initial values required for execution into the task's container. This displays a screen with the mandatory import parameters of the task. To display the rest of the import parameters as well, choose Show elements.

The function Deadline data can be used to enter deadlines for the execution of the task. The relevant missed deadline message recipient is declared in the definition of a single-step task as the default role, or in a workflow definition as part of the basic data.

The selected workflow is started using . After the start you can branch to the Business Workplace via and the workflow log via .

Activities
You can access the test environment for starting workflows via Tools → Business Workplace → Development → Runtime tools → Start workflow (test environment) or from the Workflow Builder via .
Advancing with Immediate Dialog

Use
The runtime system of *SAP Business Workflow* supports advancing through work items *with immediate dialog* as standard.

Features

Advancing with immediate dialog means that when the current agent of a work item in the workflow has executed this work item, the method of the next work item is presented to them directly in dialog provided they are one of the recipients of the next work item. The runtime system behaving in this way means that a user does not have to "detour" via the workflow inbox of the Business Workplace, which would first have to be refreshed, in order to *reserve* or *execute* the relevant work item.

A user decides in a user decision to revise the notification of absence rejected by their superior. Since the user is therefore also the recipient of the next activity "Revise notification of absence", the relevant method is executed for them immediately.

The runtime system behaving in this way also, however, means that a work item is no longer offered to all recipients for processing in the status ready, since the first agent in such a chain of work items automatically becomes the current agent of the next work items.

Starting workflows

If *Advance with immediate dialog* is activated, the method for the first work item appears immediately when a workflow is started in dialog. The initiator of the workflow must, of course, be one of the recipients of this work item.

Since advancing with immediate dialog can only be activated or deactivated for workflows, work items for single-step tasks are always started with immediate dialog.

For further information, refer to *Starting Tasks* [Seite 1448].

Conditions

The following conditions must be met:

- The task underlying the step references a synchronous method [Extern].
- The step does not have a requested start date in the future, but can be started immediately.

Technical details

At runtime, when a step has been completed (= work item has status *completed*), the system checks whether the next work item can be started *immediately* with dialog and whether the actual agent is also a recipient of the next work item.

If this is the case, the work item is reserved for the agent as soon as it is created and the relevant object method is executed.

This procedure continues until *more than one* work item or *no* work items are created that can be executed immediately by the actual agent. More than one can occur when forks are used, for example. A user can also leave the "chain" by canceling execution of the method.
Activities

You activate or deactivate *advance with immediate dialog* in the basic data of a workflow definition. This setting then refers to advancing for all steps in the workflows relating to this definition. The first time the Workflow Builder is called, *advance with immediate dialog* is always activated.

You can also activate or deactivate *advance with immediate dialog* within the definition of an activity in a workflow definition. This setting then refers only to advancing from the preceding step to this step.

If you have deactivated advancing for the workflow in the basic data, you cannot activate it at the step level.
Defining Start Transactions

Use
The workflow system can create an individual transaction to start a particular workflow. A start transaction provides input fields that are used to assign values to the import parameters of the workflow container. You can edit this transaction at a later date using the ABAP Workbench, for example to change the names of input fields or adapt the layout.

If you select a workflow by choosing Office → Start workflow, for which a start transaction is defined, this start transaction is executed.

For further information, refer to Starting Workflows in Dialog [Seite 1450].

Prerequisites
You have defined a workflow and marked the container elements required as import parameters.

Features
The workflow system generates a transaction to start a given workflow. The input fields are derived from the import parameters of the workflow container.

Container elements that reference a structure cannot be assigned values using input fields.

- The workflow system generates the name of the transaction automatically from the ID of the workflow. A successfully-generated transaction can be recognized by the symbol adjacent to the transaction name.

- The workflow system also generates the name of the module pool automatically if you do not specify anything yourself. You determine the screen number. In this manner, you can create various screens for various workflows in the same module pool.

Activities
Choose Tools → Business Workflow → Development → Definition tools → Workflow start transactions to access the workflow start transactions function.

You specify a workflow ID, a screen number and, optionally, a module pool in the field Program. Select .

Alternatively, you can generate a start transaction for a workflow from the Workflow Builder. To do this, select on the tab page Start in the basic data of the workflow in the Workflow Builder. You can also process existing start transactions from here.

You can use the Screen Painter functions to revise the transaction generated. For further information, refer to the Screen Painter [Extern] documentation.

If you want to change the user interface status of the transaction generated, you must define a separate user interface status. You can use the user interface status DYNPRO as a model.
Starting Workflows or Tasks from Message Long Texts

Use

Workflows or tasks can be started from message long texts. If there are appropriate (target) elements in the workflow container or task container, the system passes the following data to the started workflow or started work item:

<table>
<thead>
<tr>
<th>Container element</th>
<th>Data type reference</th>
<th>Explanation</th>
</tr>
</thead>
<tbody>
<tr>
<td>ARBGB</td>
<td>T100-ARBGB</td>
<td>Message ID (application area, for example 'WO')</td>
</tr>
<tr>
<td>MSGNR</td>
<td>T100-MSGNR</td>
<td>Message number (for example '430')</td>
</tr>
<tr>
<td>MSGTY</td>
<td>SYST-MSGTY</td>
<td>Message type ('E', 'A', 'S', etc.)</td>
</tr>
<tr>
<td>MSGV1</td>
<td>SYST-MSGV1</td>
<td>Message variable 1</td>
</tr>
<tr>
<td>MSGV2</td>
<td>SYST-MSGV2</td>
<td>Message variable 2</td>
</tr>
<tr>
<td>MSGV3</td>
<td>SYST-MSGV3</td>
<td>Message variable 3</td>
</tr>
<tr>
<td>MSGV4</td>
<td>SYST-MSGV4</td>
<td>Message variable 4</td>
</tr>
<tr>
<td>TEXT</td>
<td>T100-TEXT</td>
<td>Message text in which the variables (&amp;) are not replaced</td>
</tr>
<tr>
<td>MESSAGE</td>
<td>Without type, that is C(255)</td>
<td>Message text in which the variables are replaced</td>
</tr>
</tbody>
</table>

This data is available in the container of the task started and can be used to control the process flow.

An application document is incomplete. The application communicates this via a message.

The Start workflow function is available to the user in the dialog box with the message long text. They can use it to start the linked workflow that will help solve the problem.

Features

You are supported in the functions described below by a Workflow Wizard. Choose Tools → Business Workflow → Development → Definition Tools → Wizards → Create "Call Workflow From Message".

How do you link a message to a workflow?

To implement this functionality, you enter the assignment between task and message in table T100W. Maintenance of this table is client-specific.

If the error message long text is displayed and a T100W assignment maintained, a function is displayed with which the relevant task can be executed.

A message can be linked to all possible task types (WS, WF, T, TS).

It is important that the container elements described above have been added to the container concerned (workflow container or task container). Otherwise, the information from the message cannot be evaluated in a meaningful way in the task started.

Workflow definition details

When defining a task that is to be started from a message long text, you create the elements specified above in the container. At runtime, the system then fills the container elements with the relevant values from the message.
The object reference to the application object to which the error refers, is not directly available via the message variables as a rule.

If the message variables are to contain the values of object key fields, the object reference must first be created within the workflow from these key fields. To do this, you must provide an appropriate instance-independent method with import parameters as shown in the following example (object method Build):

```
BEGIN_METHOD BUILD CHANGING CONTAINER.
  DATA: KEYFIELD LIKE....
  SWC_GET_ELEMENT CONTAINER 'MSGV1' KEYFIELD.
  SWC_SET_OBJECTKEY KEYFIELD.
END_METHOD.
```
Start Conditions for Workflows

Use
You use this function if you want to define additional conditions for the start of a workflow.

You can only define start conditions for workflows that are started by an event.

Integration
Refer also to the following for workflow start conditions:
- Event [Seite 1321]
- Starting Tasks [Seite 1448]

Features
This function enables the definition of start conditions for a workflow. This means that the start of a workflow no longer depends only on whether a specific event is created, but also on the occurrence of additional conditions.

A workflow is to be started when a notification of absence has been created and the creator of the notification of absence belongs to a specific group of people.

Selection area
The following selection criteria are available for selecting a particular workflow:
- **Start condition**
  Here you can search for the name of a particular start condition defined for a workflow.

- **Business object**
  Here you can search for workflows with start conditions using a particular object type.

- **Event**
  All the events for the chosen object type, which are linked with workflows as triggering events are offered for selection here.

- **Workflow**
  Here you can search directly for workflows with start conditions.

  If a newly defined triggering event [Seite 1206] is not displayed, check that the event linkage was activated when the workflow was defined.

### Definition area

In this area you can define additional start conditions for workflows or view and change existing start conditions. The individual lines of a start condition provide the following information:

- **Name of the start condition**
- **Currency**
- **Triggering event of workflow**
- **Start condition**

To **change** the start condition, the name of the condition or the currency, single-click on the relevant area.

The [condition editor [Seite 1012]] is used to determine conditions.

Select **✓** to **activate** the start condition (or **✗** to **deactivate** it). Select **🗑** to **delete** the start condition.

### Personal settings

You can display and edit personal settings by choosing **Extras → Settings**. In particular, you can set whether the maintenance of start conditions is to be cross-client or client-specific.

In the cross-client view you cannot activate or deactivate start conditions if they are redefined in a client. They are marked with the symbol **⚠️**. Start conditions that are not redefined are only **activated or deactivated for the current client** in the cross-client view.

### Activities

Choose the following for workflow start conditions: **Tools → Business Workflow → Development → Definition Tools → Events → Event Linkages → Workflow Start Conditions**.

To create a new start condition, select **✓**.
You can also create start conditions directly from the Workflow Builder. To do this, go to the tab page Start in the basic data of the workflow. Click on the cell Start condition in the line of the relevant triggering event. This displays the condition editor directly.
Remote Worklist

Use
A user is notified of the existence of work items they are to process in a remote R/3 System by a work item in their Business Workplace. The work items in the other R/3 System form the remote worklist of this user with regard to the current system.

When the user executes the notification work item, the system calls the user's Business Workplace in the remote system without any additional logon. The user can then process the work items in the remote system.

- Once the user has processed their remote worklist fully, the notification work item immediately disappears from the current system.
- If the user has only processed their remote worklist partially, the notification work item remains in the current system with the status in process.

Integration
The current system, in which the user is notified of a remote worklist, is referred to as the client system. The system in which the work items in the remote worklist are to be executed is known as the server system.

Prerequisites
The user must have the same R/3 user name in both the client and the server systems.

Features
Each server system has a **subscriber table** that specifies who wants to be informed in which client system of the presence of work items in the server system. The system executes a report periodically in the server system, which checks the following for each subscriber in the subscriber table:

- Whether there are any work items for them in the server system
- Whether there is already a notification work item for them in the client system

The following possible scenarios result:

<table>
<thead>
<tr>
<th>Scenario</th>
<th>There is already a notification work item in the client system.</th>
<th>There is not yet a notification work item in the client system.</th>
</tr>
</thead>
<tbody>
<tr>
<td>There are work items (and therefore a remote worklist) in the server system.</td>
<td>Since the user has already been notified, no further action is necessary.</td>
<td>The system creates a work item in the client system, which notifies the subscriber of the existence of their remote worklist.</td>
</tr>
<tr>
<td>There are no (more) work items in the server system.</td>
<td>The system deletes the work item from the client system.</td>
<td>No action is required here.</td>
</tr>
</tbody>
</table>

The notification work item in the client system is always created after the associated remote worklist. The maximum time delay is equal to the period with which the report runs in the server system.

When the notification work item is executed by the recipient in the client system, the system calls their current worklist in the server system via RFC. The worklist may not exist anymore on the server system. This situation may arise if there was a remote worklist but it has been processed in the server system and the report has not been run again there in the mean time.

**Activities**

You create a logical RFC destination in both the client system and the server system, each of which refers to the other system. In this way, you tell each system what the other system “is called” and how it is to be called.

The server system contains a subscriber table, which specifies which users in which system are to be notified when a work item is created for them in the server system. Depending on your enterprise organization, this entry is made either by a system administrator or by each user.

You schedule a report periodically in the server system, which determines whether there are work items for each subscriber in the subscriber table and, if necessary, creates notification work items for them in the client system.
Creating a Logical RFC Destination

Prerequisites
The server and client systems must be configured as mutually trusted systems so that a logon via RFC can take place without a disruptive password query. To set this status, carry out the following steps:

1. Choose Tools → Administration → Administration → Network → RFC Destinations.
2. Choose RFC → Trusted systems.
3. Follow the instructions in the online help (Documentation function).

Procedure
Carry out the following system administration steps once in both the server and client systems:

1. Choose Tools → Administration → Administration → Network → RFC Destinations.
2. Choose Edit → Create.
3. Enter any name as a description in the RFC destination field.
4. Enter 3 in the Connection type field (connection to R/3 System).
5. Describe the RFC destination.
   Enter the following text in the first line of the description:
   
   System <target system of RFC destination> Client <client in target system>

   The system uses this text as the work item text. You can enter any text in the other two lines.

6. Enter the message server of the target system in the Target host field.
7. Select Trusted System Yes to register the current system as a trusted system in the target system.
8. Select the Current user indicator in the Logon box.
9. Enter the target system Client in the Logon box.
Specifying Subscribers

Use
All users who want to be informed about their worklists must be entered in the subscriber table. This subscriber table is maintained in the relevant server system.

Prerequisites
Appropriate authorization is required to maintain the table.

Procedure
1. In the server system, choose System → Services → Table maintenance → Extended table maintenance.
2. Enter the view name SWLCREMABO in the Table/view field.
3. Select Maintain.
4. Select New entries.
5. Identify the subscriber(s) with an appropriate organizational object.
   To do this, enter an organizational object type in the Object type field. The entries allowed are US (user), S (position), O (organizational unit), C (job) and A (work center).
   In the field Agent ID, enter the ID of an organizational object appropriate for the object type specified above.
   
   Possible entries include:
   Object type US, Agent ID MÜLLERH
   Object type O, Agent ID 50001234
   If you enter an organizational unit, the subscribers are the employees assigned to this organizational unit or a subordinate organizational unit.
   If you enter a job, the subscribers are the employees whose positions are described by this job.
6. Specify the RFC destination by which the client system is known in the server system (that being here) in the first Destination field.
7. In the Task ID field, enter the task that informs the subscriber in the client system. You can enter the standard task TS60001345 here.
   If you do not like the description text or work item text of this supplied task, you can use a copy of this task, which will also refer to the method ProcessAllItems of the object type REMOTE_WLC.
   This task must exist in the client system.
8. Specify the RFC destination by which the server system is known in the client system (that being in the remote system) in the second Destination field.
9. Save your entries.
Scheduling a Report

Use

The report RSWLJCJOB evaluates the subscriber table in the server system and, if necessary, creates the notification work items in the relevant client systems. If the subscribers in the subscriber table are specified as jobs, positions, work centers or organizational units rather than as users, this information is updated each time the report is run. You schedule this report periodically.

You can also execute the report online for test purposes.

Procedure

1. Choose Tools → CCMS → Jobs → Definition.
2. Schedule a job of job class A under any job name.
3. Create the ABAP program RSWLJCJOB as the first (and only) step.
4. Specify a start date/time and a period value.
   The period length depends on the time delay (which you can take into account). Generally, execution on an hourly basis is sufficient.
Workflow Information System

Use

Standard analyses

The Workflow Information System [Extern] (WIS) lets you make the following standard analyses for completed work items:

<table>
<thead>
<tr>
<th>Analysis View</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Organization view</td>
<td>Workload analysis that emphasizes the work item’s agents and their organizational assignments.</td>
</tr>
<tr>
<td>Process view</td>
<td>Process analysis that emphasizes the superordinate workflow (the “process”).</td>
</tr>
<tr>
<td>Object view</td>
<td>Analysis that emphasizes the processed object.</td>
</tr>
<tr>
<td>Group view</td>
<td>Analysis that emphasizes certain objects from the environment of the processed work item.</td>
</tr>
</tbody>
</table>

The data basis for all analyses is returned by completed work items only. These work items were transferred to the information structure of the Logistics Information System [Extern] (LIS) by a periodically scheduled job. These tables receive the data for analysis in summarized form. These tables allow a high-performance analysis, independently of running operations.

Enhancement

You can customize the WIS by individually structuring and extending the data basis of your information system. In particular, you can include specific information from the environment of your own workflow in separate standard analyses.

A special analysis determines how often applications for leave are approved or not approved, and by whom.

This analysis is realized with the standard analysis Example scenario. In its information structure, this standard analysis only contains data on the work items that represent the single-step task TS30000016 in the workflow template WS20000011.

Integration

The connection to the LIS allows the WIS to profit from its powerful reporting system. Its advantages include a common user interface, multiple statistical analysis functions, graphics connection, and the export function to Microsoft Excel.

You can find the documentation on these functions in Customizing of the Logistics Data Warehouse under the options Data basis and Updating.
Using the Workflow Information System (WIS)

Use
The following notes tell you how to access the WIS easily for the first time, and describe selected functions.

Integration
For more information, refer to the LO - Logistics Information System [Extern] documentation.

Features

Initial list
After you have called the analysis, and restricted it if necessary by specifying selection criteria, an Initial list of this analysis is displayed.
The first column lists all values for a Characteristic that correspond to the chosen selection criteria. This characteristic is set specifically for each analysis.
The following columns contain the key figure values for the selected characteristics. The corresponding units are displayed for all key figures.
You can use the Settings menu to set the display of the characteristics and key figures for specific users.

Drilldown function - Drilldown list
You can use the drilldown function to vary the amount of information on display. You can determine the detailed data that you want to display by choosing certain criteria.
The WIS uses the so-called Characteristic drilldown: Double-click on a characteristic value to show it in more detail with regards to another characteristic of the same analysis. The order in which the characteristics are drilled down follows a default analysis path called the Standard drilldown.
In contrast to the Initial list, the Drilldown list displays the detailed characteristic with characteristic value in the upper half of the screen.
To display the characteristics and the order of the characteristics of the standard drilldown, choose Extras → Display standard drilldown. A dialog box appears showing the structural graphic of the standard drilldown.
If you want to drill down the characteristics in another order, position the cursor on the characteristic value for which you want the detailed information and choose View → Drilldown by. A dialog box appears on which you can choose the characteristic by which you want to drill down.
To return to the Initial list from a Drilldown list, choose Goto → Initial list.

Switching the drilldown
You can use the function Switch drilldown to change the view on the data displayed in the list.
You can use this function in the Initial list to switch to an Initial list based on a different characteristic, but which is structured according to the same selection criteria.

Key figures
A sequence of key figures is displayed as standard for each characteristic value. To choose other key figures to be displayed in the list, choose Edit → Choose key figures.
A dialog box appears with a list of all available key figures. You can now choose the key figures to be included in the list display. You can delete key figures, change their sequence and choose new key figures from the basic information structure for the list display.
For further information, refer to Key Figures Used [Seite 1470].
Key Figures Used

Key Figures Used in the Standard

**Number**
Number of work items for each selected characteristic.

**Wait time**
Cumulative wait time of the work items for the corresponding characteristic, in hours
The wait time is the difference between the start of processing (= first switch to status *in process*) and the creation time of the work item.

**Process time**
Cumulative process time of the work items for the corresponding characteristic, in hours
The process time is the difference between the end of processing (= switch to status *completed*) and the start of processing (= first switch to status *in process*) of the work item.

**Total time**
The cumulative total time is the sum of wait time and process time (in hours).

**Average**
The average total time in hours is the ratio of cumulative total time to the number of work items.

**Dispersion**
Dispersion (mean squared deviation) of the average total time.

**Optional Key Figures**

**Average wait time: Tar_Mittel**
The average wait time in hours is the ratio of cumulative wait time to the number of work items.

**Average process time: Net_Mittel**
The average process time in hours is the ratio of cumulative process time to the number of work items.

**Dispersion: Wait time: Tar_Sigma**

**Dispersion: Process time: Net_Sigma**

**Wait time: Squared total: Tar^2_Sum**
Squared total of the wait times.

**Process time: Squared total: Net^2_Sum**
Squared total of the process times.
Standard Analysis Organization View

Use
The standard analysis Organization view selects work items according to their position in the organizational plan. This emphasizes the agents and their department. This analysis allows you to find out who did what, and how long did they take.

Integration
The information structure S300 forms the data basis for the standard analysis Organization view.

Prerequisites
The information structure S300 has to be filled with data. This is done by transferring the data to the LIS with the program RMCADATA, that has to be scheduled periodically as a job. The update rules for the standard analysis Organization view determine which data from the runtime tables is copied to the information structure S300.

Features
Characteristics
The standard analysis Organization view uses the following characteristics: (The characteristics are given in the order in which they are shown.)
- Organizational unit to which the agent belongs (through the agent's position)
- Agents
- Task identifier
- Date of final processing
- Work item

Key figures
For further information, refer to Key Figures Used [Seite 1470].
Standard Analysis Process View

Use
The standard analysis Process view selects work items according to their position in the process structure. This emphasizes the superordinate workflow and the related multistep task. This analysis allows you to find out what processes took place, for what reason, and how long they took.

Integration
The information structure S301 forms the data basis for the standard analysis Process view.

Prerequisites
The information structure S301 has to be filled with data. This is done by transferring the data to the LIS with the program RMCA DATA, that has to be scheduled periodically as a job. The update rules for the standard analysis Process view determine which data from the runtime tables is copied to the information structure S301.

Features
Characteristics
The standard analysis Process view uses the following characteristics: (The characteristics are given in the order in which they are shown.)

- Process definition (= multistep task whose instances you want to analyze)
- Process instance (= workflows as instances of a multistep task)
- Date
- Work item

Key figures
For further information, refer to Key Figures Used [Seite 1470].
Standard Analysis *Object View*

**Use**
You can use the standard analysis *Object view* to answer queries on a particular object type.

What has been done in the area of notifications of absence and how long did it take?

**Integration**
The information structure S302 forms the data basis for the standard analysis *Object view*. The *Object view* is based on the object that is referenced in the mandatory task container element _WI_Object_ID_. This container element is always filled for dialog work items.

**Prerequisites**
The information structure S302 has to be filled with data. This is done by transferring the data to the LIS with the program RMCA DATA, that has to be scheduled periodically as a job. The update rules for the standard analysis *Object view* determine which data from the runtime tables is copied to the information structure S302.

**Features**

**Characteristics**
The standard analysis *Object view* uses the following characteristics: (The characteristics are given in the order in which they are shown.)

- **Object type** (of the object referenced in _WI_Group_ID_)
- **Object** (= concatenated key fields of the referenced object)
- **Task identifier**
- **Date**
- **Work item**

**Key figures**
For further information, refer to [Key Figures Used Seite 1470](#).
Standard Analysis Group View

Use

You can use the standard analysis Group view to answer queries on a particular object type. This object is not a central object that is processed directly in the work item, but an object that has an ordering role in the process environment.

An analysis must be made in connection with the processing of production orders. This analysis originates in the higher-level sales order.

Integration

The information structure S303 forms the data basis for the standard analysis Group view. The Group view is based on the object that is referenced in the task container element _WI_Group_ID. This container element is not filled as standard. For purposes of grouping and analysis it must be given an object reference explicitly by using binding.

For the example above, this means that:

In the element _WI_Object_ID of the task container, each of the work items for processing a production order references one object of the type Production order.

In addition to this, in each task container, the element _WI_Group_ID has to be filled with the object reference to the sales order relating to the production order.

Prerequisites

The information structure S303 has to be filled with data. This is done by transferring the data to the LIS with the program RMCADADATA, that has to be scheduled periodically as a job. The update rules for the standard analysis Group view determine which data from the runtime tables is copied to the information structure S303.

Features

Characteristics

The standard analysis group view uses the following characteristics: (The characteristics are given in the order in which they are shown.)

- Group type (object type of the object referenced in _WI_Group_ID)
- Group (concatenated key fields of the referenced object)
- Task
- Date
- Work item

Key figures

For further information, refer to Key Figures Used [Seite 1470].
Standard Analysis Example Scenario

Use
This standard analysis shows you an example of a customer-specific extension. The communication structure MCWF_TRANS is extended, and the data retrieval for the extension is placed in the subroutine WIS_CODING_TEMPLATE.

Integration
The information structure S304 forms the data basis for the standard analysis Example scenario. This information structure only contains the data of completed work items of the single-step task TS30000016 in the workflow template WS20000011.

Prerequisites
The information structure S304 has to be filled with data. This is done by transferring the data to the LIS with the program RMCADATA, that has to be scheduled periodically as a job. The update rules for the standard analysis Example scenario determine which data from the runtime tables is copied to the information structure S304.

Features

Characteristics
The standard analysis Example scenario uses the following characteristics: (The characteristics are given in the order in which they are shown.)

- **Processing status** of notification of absence after the approval step. Characteristics are approved or rejected.
- **Applicant** for notification of absence
- **Approver** of the notification of absence
- **Date** of processing
- **Work item**

Key figures

Number
Number of work items for each selected characteristic.

Total time
The cumulative total time is the sum of wait time and process time (in hours). The wait time is the difference between the start of processing (first switch to status in process) and the creation time of the work item. The process time is the difference between the end of processing (switch to status completed) and the start of processing (first switch to status in process) of the work item.
Periodic Data Transfer

Use
The data for the WIS has to be read from the runtime tables of the workflow system, formatted and then copied to the information structures of the LIS. The data is not transferred in a continuous automatic process, instead it is copied by the “data retrieval program” RMCADATA. You must schedule this program periodically.

In order to be able to use the WIS productively, you have to carry out the activities described below.

Prerequisites
Only those work items are copied to the WIS that have the status completed. Another restriction is that work items that are parts of a workflow are only copied when the workflow has the status completed.

Features
The “data retrieval program” RMCADATA reads selected data on the work items that fulfill the above requirements from the runtime tables, and then transfers this data to the LIS in the communication structure MCWF_TRANS. The LIS then “distributes” it to the information structures. The data refers to completed work items of the following types:

- Work item of type W (dialog work item) representing a single-step task with dialog
- Work item of type B (background work item) representing a single-step task without dialog
- Work item of type F (workflow) representing a multistep task.

Only part of the data transfer to the LIS is distributed to the individual information structures. The way in which the data is distributed is controlled by update rules. These rules have to be set for each information structure.

You can extend the communication structure MCWF_TRANS on a customer-specific basis. You have to program a user exit for extended data retrieval. This procedure is described below.

Activities

Creating a variant for the report RMCADATA
Create a variant for the report RMCADATA that assigns the selection criterion evaluation period a variably calculated date.

Scheduling a job
Define a job that has the report RMCADATA as its only step, and schedule it periodically.
Creating a Variant for the Report RMCADATA

Prerequisites

The report RMCADATA copies the data from work items that were completed within a particular Analysis period. The analysis period is set in the selection criteria of the report by setting an end date (single value or interval).

Create a variant to specify values for the selection criteria of a report that are then set when the report is started and allow the report to be executed in the background.

Procedure

Carry out the following steps in the ABAP Editor.

Create a variant of the report RMCADATA.

The variant maintenance screen appears.

Enter a date of your choice in the field Finish date.

Choose Continue to exit variant maintenance.

A screen appears with the attributes of the variant.

Enter a short text in the field Description.

Select the column Selection variable for the field Finish date.

Then choose Selection variables.

A screen appears where you can select the variant variables.

Select the traffic light under the letter D (Dynamic date calculation) so that it turns green.

Enter a dynamic analysis period in the field Variable name.

Depending on how often you schedule the report, in other words how often you want to copy the runtime data into the information system, enter one of the following values for the name of the variable: Use F4 help to make an entry.

Current date

Current date +/- ??? days. Replace ??? with a negative value of your choice.

Save the variant variables.

Save the attributes of the variant.

Result

You have created a variant of the report program RMCADATA that uses dynamic date calculation to specify an analysis period.
Scheduling a Job

Prerequisites
You have created a variant of the report program RMCADATA that uses dynamic date calculation to specify an analysis period.

Procedure
Carry out the following steps in the Computing Center Management System (CCMS).
Define a job with the job class A.

Create a step in which the variant you have defined for the ABAP program RMCADATA is executed.

Schedule the job periodically. Specify the length of the period so that it matches the analysis period set by the variant.

You have defined the variant so that the analysis period is the current date. Schedule the job daily with the start time 23:45:00.

You have defined the variant so that the analysis period lasts from five days ago until the current date. Schedule the job every five days with the start time 23:45:00.
Making Customer-Specific Enhancements to the WIS

Purpose
You want to evaluate the data of a particular task within the WIS. This data:

- Is made up of the contents of the container elements of the task container
- Can be retrieved using expressions that refer to elements of the task container

Prerequisites
Analyze the container definition of the single-step task. Determine which container elements are relevant for the evaluation. Note that you can also access the attributes of any object references in the container elements. Establish the associated data element for each field that is relevant for the evaluation.

Process Flow

Extending the communication structure
Extend the ABAP Dictionary structure (communication structure) MCWF_TRANS by adding the fields that you need for your own analysis.

Filling the communication structure
In a second step, the customer fields that have just been defined have to be filled with data. The function module (user exit) EXIT_SAPLMCWF_001 has to be programmed for this.

LIS Customizing
Make the following settings in Customizing of the Logistics Information System:

- Maintain user-defined field catalogs
- Maintain user-defined information structures
- Updating definition
- Standard analyses

Result
The extension of the communication structure MCWF_TRANS and the user exit EXIT_SAPLMCWF_001 ensure that the periodic background job RMCADATA passes the customer scenario-specific data to the LIS, as well as the SAP data. The data then passes to the information structures S<xxx> according to the update rules, and can be used for standard analyses.
Extending the Communication Structure

Prerequisites
You know which elements in the task container hold the data that is needed for the analysis, and you know the data elements that are referred to by these container elements.

Procedure
Go to the area menu of the WIS and choose Environment → Enhancements → Append structure.

The structure name MCWF_TRANS is set as a default.

Extend the structure with an append structure. This technique allows you to make customer enhancements to SAP data structures, without modifying the SAP structure.

Choose Goto → Append structures.

Give the append structure a name of your choice.

Create new fields in the append structure. Specify those data elements as a reference that you determined earlier for the container elements.

Generate the append structure.

You can find an example of an append structure in the include MCWF_TEMPL of the structure MCWF_TRANS.
Retrieving Data for Customer Fields

Prerequisites
You have extended the communication structure MCWF_TRANS with your own append structure.

Procedure
Go to the area menu of the WIS and choose Environment → Enhancements → Selection program.

   The function module EXIT_SAPLMCWF_001 is set as a default.

   When called, the function module receives the internal table XMCWF_TRANS with the communication structure MCWF_TRANS as a table parameter. The fields defined by SAP are already filled.

Program the source code for this function module.

   Use the form routine WIS_CODING_TEMPLATE of the main program SAPLMCWF as a guide. (Transaction SE80, function group MCWF). The procedure is exactly the same.

   First program an IF or CASE construction, to make sure that you analyze work items for particular tasks only. The task ID is in the field XMCWF_TRANS-WI_RH_TASK.

   Use one of the two function modules SWI_READ_CONTAINER_ELEMENT or SWI_READ_CONTAINER_ATTRIBUTE within the IF or CASE construction so that each of your self-defined fields can read the appropriate data from the runtime tables.

   Both function modules are called with the following parameters:

   WI_ID:

       The work item ID is recognized from the field MCWF_TRANS-WI_ID.

   ELEMENT:

       ID of the container element.

       The function module SWI_READ_CONTAINER_ATTRIBUTE also has the import parameter ATTRIBUTE (attribute of an object that is referenced in ELEMENT).

       Both function modules return the value in the element VALUE.

       Use this to fill the communication structure MCWF_TRANS.

   Activate the user exit EXIT_SAPLMCWF_001.

   Use transaction SMOD/CMOD to activate the project or the SAP enhancement WISEXIT.
LIS Customizing

Decide which of your fields are *Characteristics* and which are *Key figures*, according to the LIS. Create appropriate *Field catalogs*.

The corresponding application for workflow is 20.

Define information structures. You can combine SAP fields and customer fields in almost any way that you want.

The fewer fields, the more summarized the data on the database, but the fewer the drilldown options in an analysis.

Generate the information structures.

Determine the update rules for each of your information structures. The update group is 000100. Stick to the system’s proposals.

Generate and activate the rules.

You can find an entry for each of your information structures by going to the *Area menu* of the WIS and choosing *Standard analyses → User-def. analysis*. Choose an entry to generate and start a standard analysis for each information structure.
Work Item Analysis

Use
You can use work item analysis to carry out statistical evaluations on work items. The work items to be analyzed can be limited according to time, type and task.

Features

Work items per task
Number of work items of the specified type or for the specified tasks generated in the specified monitoring period. The list is sorted according to task. Double-clicking on an entry in the list displays all the work items for the task in another list. From this list you can select to go to the work item display and to go to the workflow log.

Work items with monitored deadlines
Number of work items of the specified type or specified tasks, which are subject to deadline monitoring. The work items are displayed in an overview tree sorted according to the type of deadline monitoring and task. Double-clicking on an entry displays the work item.

Work items by processing duration
Information on the processing duration of work items of the specified type or on the specified tasks whose processing ended in the monitoring period. The list is sorted according to task. Provided there are appropriate work items, the current monitoring period is compared with a prior period of the same length. The variances and differences are shown. For further information, refer to Technical Details on Calculation and Threshold Value and Average Value.

Activities
To call work item analysis, choose Tools → Business Workflow → Development → Reporting → Work item analysis → ...
Workload Analysis

Use
Using the workload analysis, you can establish the workload of individual users, positions, jobs or organizational units.
You can look into the past (“completed since...”) and into the future (“to be processed by...”) in order to establish workload.

Features

Workload for the past
The selection produces a list of work items completed by the members of the organizational entity before the specified date.
A work item must meet the following criteria to be displayed in this list:
5. The work item must represent a single-step task that satisfies the selection criteria specified as a task filter.
6. The work item must be a dialog work item [Seite 1372].
7. The work item must have the status [Seite 1373] completed.
8. The actual agent [Extern] of the work item is a user who is assigned directly or indirectly to the organizational entity specified.
The result of the selection is displayed with the following information in the list Work items completed according to agent and task:
• Organizational unit of the current agent
• Agent
• Number of the single-step task represented by the work item
• Date of final completion
• Work item text
• Process duration
If you double-click on an entry, you go to the relevant work item display [Seite 1411].

Export as Microsoft Excel document
The list displayed can be exported as a Microsoft Excel document. To do this, choose the function Excel.

Formatting for statistical purposes
The information can be formatted for statistical purposes.
To do this, position the cursor on one of the columns Org. unit, Agent, Task or Date, and select the function Statistics.
Depending on the column you selected, the following information is displayed:
• Organizational unit and number of work items completed by an employee from the organizational unit.
• Agent and number of work items completed by this agent.
• Task and number of work items for this task, which were completed.
• Date and number of work items completed on this date.
In addition, the following times are calculated:

- **10% threshold:**
  
  For 10% of the work items, the duration of processing was shorter than the time specified.

- **50% threshold:**
  
  For half of the work items, the duration of processing was shorter than the time specified.

- **90% threshold:**
  
  For 90% of the work items, the duration of processing was shorter than the time specified.

**Workload for the future**

The selection produces a list of work items that must be processed by the members of the organizational entity by the date entered.

A work item must meet the following criteria to be displayed in this list:

7. The work item must represent a single-step task that satisfies the selection criteria specified as a task filter.

8. The work item must be a dialog work item [Seite 1372].

9. The work item must have one of the following statuses [Seite 1373]: waiting, ready, in process, reserved or executed.

10. The actual agent [Extern] of the work item must be a user who is assigned directly or indirectly to the organizational entity specified.

   This only applies to work items with the status waiting, in process, reserved or executed, since there can only be an actual agent in these cases.

11. There must be at least one user among the possible agents [Extern] of the work item, who is assigned directly or indirectly to the organizational entity specified.

   (This only applies to work items with the status waiting or ready since these do not yet have a current agent.)

12. The latest end for processing the work item must be before the date entered.

The result of the selection is displayed in the list Workload for .... The list of work items is grouped according to current agents and single-step tasks. At the end of the list, the work items and tasks for which no actual user exists are displayed under the header Not reserved by an agent.

**When no date is specified**, a user’s workload corresponds to the contents of their worklist in the Business Workplace.

**Operations on the list**

You can go to the definition of a task by double-clicking.

You can display the work items for a task by double-clicking on the plus sign in front of the task.

You can go to the work item display [Seite 1411] by double-clicking on a work item.

**Activities**

To call workload analysis, choose Tools → Business Workflow → Development → Reporting → Workload analysis.
To determine the past workload, choose the option Completed since on the selection screen Workload analysis. For future workload choose To be processed by.
Use the F4 input help for all other specifications.
Task Analysis

Use
Task analysis enables you to display an overview of the defined tasks with their dependencies and relationships.

Task analysis provides information from the environment of the task definition and workflow definition. You do not receive runtime information.

Prerequisites

Authorization
You require appropriate authorization for the functions shown. This is an authorization based on the authorization object S_PROGRAM allowing a program to be executed for authorization group 'SWI_TA'.

Features

Tasks in workflows
You can use this function to create a list of all workflows in which a particular task is used as an activity. The following are displayed:

- Client
- Task
- Version
- Step
- Step name

If you can double-click on an individual step in this list you go to the quick view of the Workflow Builder [Seite 1002].

For information about the functions in the list, refer to the documentation on the ABAP List Viewer [Extern].

The use of a decision task [Extern] in a user decision is not recognized.

Task profile
An activity profile and the organizational authorization for executing and starting tasks are determined starting from an organizational object (work center, job [Extern], position [Extern], organizational unit [Extern] or user). The basis for this analysis is the assignment of possible agents [Extern] for the individual tasks.

You can choose whether you display the list with or without the organizational structure. You can switch between the two views using the function Other representation. You can go maintenance of the tasks displayed by double-clicking.

Activities
To go to task analysis, choose Tools → Business Workflow → Development → Reporting → Task analysis.
Use the F4 input help for all the entries required.
Work Item Selection

Use
This function can be used to select and display work items of all types according to various criteria. The option of finding work items using the selection report is intended primarily as a tool for administrators. In particular, you can use this report for "lost" work items that do not appear in any workflow inbox [Seite 1408] or not in the expected inbox.

If you want to program your own work item selections or evaluations, you must only use the view SWWVPUBLIC for selection. This view contains all the released header data of a work item.

Features
Selection conditions
- **ID**
  Sequence number of a work item, which is assigned automatically by the work item manager. If you know the ID of the work item, for example from an error report, enter it here. If you enter an ID as a selection criterion, the system will ignore all other selection criteria.

- **Work item type**
  For this and the two selection criteria below, refer to work item [Seite 1371].

- **Status of the work item**
- **Priority**
- **Task**
  ID of the task [Seite 1172] represented by the work item.

  The task ID must be appropriate to the work item type selected.

  Work item type W either represents a standard task (ID TSxxxxxxxx) or a customer-defined customer task (ID Txxxxxxxx).

  Work item type F either represents a workflow task (ID WFxxxxxxxx) or a workflow template (ID WSxxxxxxxx).

- **Task group**
- **Creation date, creation time**

  When specifying periods, ensure that the second entry is later than the first. For example, do not use periods such as "from 13:05:00 to 00:00:00".
You select the additional data indicator if the hit list is also to contain the columns number of the workflow definition, version of the workflow definition, workflow administrator and executed by.

**Functions on hit list entries**
The selection report displays the work items found in a hit list. The ABAP List Viewer [Extern] is used to display the hit list.

**Application toolbar functions**
- **Refresh**: This function refreshes the hit list.
- **Display work item**: This function takes you to the work item display [Seite 1411].
- **Display workflow log**: This function takes you to the workflow log [Seite 1420].
- **Display work item container**: This function take you to the work item container [Seite 1511].
- **Display task**: This function displays the underlying task of the work item.

**Functions accessible by double-clicking**
By double-clicking on an entry in the hit list, you can go:
- To the work item display [Seite 1411] if the entry is not a workflow work item (type F).
- To the workflow log [Seite 1420] of a workflow if the entry is a workflow work item (type F).

**Other functions under Edit → Work item →**
- **Restart after error**
  This function restarts errored workflows after the error has been corrected.
  The information written by the workflow system in an error log when the error occurred is now taken into account.
- **Complete manually**
  This function changes the status of the work item to completed.
  In the case of type W work items, this status change together with the current work item container is forwarded to the workflow system for evaluation. If the object method to be executed is a synchronous method with a result, the possible result values are displayed for selection.
- **Execute without check**
  You can use this function to execute the work item without any further checks.
- **Execute batch item**
  This function is for executing background work items [Seite 1385].
- **Complete with event**
  This function enables you to create an event [Seite 1365] that completes the selected work item.
- **Technical data**
  This function displays additional technical data on the selected work item.
- **Activate/deactivate workflow trace**
These functions activate/deactivate the workflow trace [Seite 1496] for an individual work item.

Use this type of call if a workflow has already been started in the background.

**Activities**

To execute work item selection, choose Tools → Business Workflow → Development → Utilities → Work Item Selection.

You start the selection by choosing Program → Execute.
Error Diagnosis

Purpose
Typical problems that may arise when you work with the SAP Business Workflow can be diagnosed and remedied. You have the option of analyzing these problems step-by-step using a diagnostic report.

Process Flow
To call error diagnosis, you choose Tools → Business Workflow → Development → Utilities → Diagnosis.
The diagnostic report always operates on a task with errors or which has been causing you problems. Enter the type of task (T, TS, WF, WS) and its 8-digit task number in the appropriate input fields.
In the lower half of the screen, you can see a list of the tasks last used from which you can choose the relevant task.

Authorization
To carry out the actions described above you require the appropriate authorization. This is an authorization based on the authorization object S_PROGRAM for executing a program for authorization group ‘SWU_DIAG’.

Workflow Customizing
There is a basic problem if the system settings for SAP Business Workflow were not made or were not completed during Customizing. Before you start error analysis for an individual task, you should ensure that Workflow Customizing is not causing any problems.
Select Verify Workflow Customizing to check the system settings and, if necessary, make or change settings.

Problem: Task is not started
A task that should be started by a triggering event or in dialog has (apparently) not been started because no corresponding work item has appeared in the Business Workplaces of the recipients.
Select Problem: Task not started. Diagnostic options are provided in the next screen.
Carry out the steps in the order shown, noting the help texts.
You first need to differentiate between the following two error causes using the function Determine instances for task:
The task really was not started
   In this case, the error was probably caused in event receiver linkage or event creation.
The task was started and an instance (work item for single-step task, workflow for multistep task) of this task was created as well.
   If you can identify a dialog work item (type W) that represents the single-step task started or a step in the workflow, but cannot be seen in a Business Workplace, check the agent assignment (possible and responsible agents [Exter]).
   If you can identify a workflow work item (type F) that represents the multistep task started, but no other work items that represent the individual steps of this workflow, an error has probably occurred that is connected with the internal communication of the workflow system components.
   For further information, refer to Workflow is “hanging”.
The individual steps that need to be carried out for diagnosis can often be reached via other menu paths.
Problem: Workflow is "hanging"

The problem here is that a workflow does not run as expected although you know that it has been started. You have either already processed steps in this workflow or you have at least seen the workflow in the selection report.

Select Problem: Workflow is hanging. Diagnostic options are provided in the next screen.

Carry out the steps in the order shown, noting the help texts.

You first need to decide whether the problem is the result of errors involving the agent assignment (work item exists but is not visible) or whether it is caused by errors in the internal process control of the workflow system.

Test environment

The test environment contains various useful tools designed to help you with problem analysis.

Note the help texts here as well.
Consistency Check

Use
There are several different functions included under Consistency check. These functions check the system settings and any definitions that have been made.

Prerequisites

Authorization
You require appropriate authorization for the functions shown. This is the authorization (authorization object S_PROGRAM) allowing a program to be executed for authorization group 'SWU_DIAG'.

Features

Consistency check for tasks
Diagnosis reports are available as a result of the following checks:
- Consistency check for standard tasks
- Consistency check for customer tasks
- Consistency check for workflow tasks
- Consistency check for workflow templates

Diagnosis reports display the check results for the individual tasks using different colored traffic lights to indicate the severity of the errors diagnosed. A description is provided for each error that occurred.

Displaying the organizational assignment of single-step tasks and work items
Information about the completeness of the definition and about the organizational assignment of work items at runtime can be displayed for a particular task. The display includes:
- Whether possible agents [Extern] were specified in the single-step task definition.
- All work items that exist as the runtime representations of a specific task with their actual agent assignment.

Activities
To perform a consistency check for tasks, choose Tools → Business Workflow → Development → Utilities → Consistency check → ...
Then select one of the checks above and restrict the tasks to be checked, if applicable.
Choose Program → Execute to start the check.
Workflow Trace

Use
All important internal process flow information is logged in the workflow trace.

The information in the workflow trace is much more detailed than that in the workflow log. You must have more in-depth knowledge of the internal process flows of the workflow system to interpret this information.

Features

Activating/deactivating the workflow trace
To activate/deactivate the workflow trace, choose Tools → Business Workflow → Development → Utilities → Workflow Trace → On/Off. All workflows and work items that you start or execute after activating the workflow trace are logged.

Displaying the workflow trace
To display the workflow trace, choose Tools → Business Workflow → Development → Utilities → Workflow Trace → Display.

Deleting the log file
Whenever the workflow trace is activated, a new log file is created. You specify how often these log files should be deleted in SAP Business Workflow Customizing.

Workflow trace for one workflow
You can activate the workflow trace for one specific workflow, which you start yourself.

Process flow
Proceed as follows:
Prepare the system so that your next action starts the workflow.

Example: Start in dialog
Call the transaction for starting workflows and select the workflow to be started.

Do not choose Start workflow yet.

Example: Start with the event created
Use the application transaction to create a document.
Do not choose Save yet.
Now activate the workflow trace in a parallel session.
Continue the transaction up to the point where the event has definitely been created or the workflow started.
Now deactivate the workflow trace again. The workflow started is still logged.
Workflow trace for an individual work item

If you do not want to activate the workflow trace for a workflow until the workflow's runtime, you can start the workflow trace for a work item. The selected work item and all work items that are then associated with it are logged in the workflow trace.

Process flow
To activate/deactivate the workflow trace, position the cursor on one of the entries displayed by the work item selection [Seite 1490] and choose Edit → Work item → Activate workflow trace.
RFC Monitor

Use
You can use this function to display the log file of the transactional RFC.

Prerequisites

Authorization
You require appropriate authorizations to carry out the actions described above. These are an authorization based on the authorization object S_PROGRAM allowing a program to be executed for the authorization group ‘SWU_DIAG’, and the system administrator authorization (authorization object S_ADMI_FCD) for the network administration.

Features
The log entries are displayed with the target system (log. destination) WORKFLOW_LOCAL_<client>.

- Choose this method of access when you want to follow up errors that have occurred in connection with the feedback from work items to the workflow runtime system after their execution. Depending on the general system load, it may take some time until this feedback is delivered.

- It is essential that you refer to the information on maintaining the logical destination WORKFLOW_LOCAL_<client> in Customizing [Extern].

Activities
To display the log file, choose Tools → Business Workflow → Development → Utilities → Workflow RFC monitor.
Demo and Test Workflows

Use
You use these workflows to test the workflow environment and familiarize yourself with certain functions. For the latter, you process the individual steps of the workflow in the Workflow Builder.

Features

Start demo workflows
When you choose this function, you go to the standard environment for starting workflows. The workflows listed demonstrate certain SAP Business Workflow functions, such as the user exit for the work item preview or the Workflow Toolbox. The individual workflows are explained on the right-hand side of the screen.

For more examples, refer to Demo Examples for SAP Business Workflow.

Start test workflows
When you choose this function, you also go to the standard environment for starting workflows. You can use this workflow to test the workflow environment.

The individual workflows are explained on the right-hand side of the screen.

Demo: Embedded inbox
You can use this function to demonstrate the features of an embedded workflow inbox, workflow outbox or workflow resubmissions.

Demo: Fill out form
You can use this function to create a notification of absence. From a technical point of view, you execute the method Create of the object type Formabsenc. You can use this function, for example, if you have implemented a workflow that is started when a notification of absence is created.

Activities
You can access these functions by choosing Tools → Business Workflow → Development → Environment.
Administration Tools

Use
The administration of the runtime system consists of several reports that can be split up into the following categories:

- Workflow runtime
- Event manager
- Workflow definition time
- WebFlow
- SAPforms [Seite 1718]

Prerequisites

Customizing
The following assumes that you have already performed automatic Customizing [Seite 1504]. Remember task-specific Customizing [Seite 1710], if you want to use SAP workflows or SAP tasks.

Refer to the documentation Workflow Scenarios in Applications [Seite 1703].

Features
If you, as workflow system administrator, need an alternative way of accessing SAP Business Workflow functions, you can use the relevant transaction code. For an overview of the most important transaction codes, refer to Important Transaction Codes [Extern].

Workflow runtime

Work item deadline monitoring
Tasks with deadlines have also have deadline monitoring based on a background job. You can change the configuration of the background job, schedule it, delete it or display it.

For further information about the background job, select (Display background job).

Work item error monitoring
If work items have the status ERROR for a long time, they are restarted automatically. This task is performed by a report that is controlled by a background job. You can change the configuration of the background job, schedule it, delete it or display it. You can also start the report manually via Execute work item error monitoring.

For further information about the background job, select (Display background job).

Reorganization

These functions include reports for archiving and deleting work items. Refer to:

- Archive work item [Extern]

During archiving, data that is no longer required in the system is checked using application-specific criteria and put in an archive file. The data is not removed from the database. Once the files which are to be archived have been copied to the archive file completely, they can be removed from the database in a separate program run.

All interactions in conjunction with archiving are performed via archive management (Tools → Administration → Administration → Archiving).
All actions or programs are processed in the background. The background jobs required can be scheduled in archive management.

For more details about archiving, refer to the documentation Archiving Application Data [Extern].

- **Display workflows from archive**
  You can use this report to display a workflow for an application object. After the workflow work item determined via the selection criteria is read from the archive, the system displays the workflow log. The functions of the workflow log are not fully available however.
  
  For further information, select [ ]

- **Delete work item [Seite 1510]**
  The report Delete work item history (RSWHIDE) deletes the work item history (all workflow log entries relevant to work items) without archiving. If you do not select the indicator Delete immediately, the report is only executed on a test basis.

**Work items without agents**
This report displays all work items without agents for a particular selection period. You can go to work item display [Seite 1411] from the list displayed by double-clicking.

**Work items with monitored deadlines**
This report displays all work items with monitored deadlines for a particular selection period. You can go to the work item display from the list displayed by double-clicking.

**Diagnosis of workflows with errors**
You can use this function to investigate and, if necessary, restart workflows with errors [Seite 1507].

**Unlock workflows**
This report can be used to select and unlock work items that are locked against execution.

  Locking tasks against execution is a function in task maintenance [Seite 1192].

**Execute work item without agent check**
This function enables a user to execute work items without checking agents [Seite 1508].

**Workflow restart after error**
This report can be used to restart workflows with errors [Seite 1509].

**Synchronize runtime buffer**
You have the opportunity to initialize the main memory buffer used by the workflow development environment. This rereads the current values from the database so that the current data is used.

  You should always refresh the runtime buffer when you have changed something in the task definition.

After execution of this report, certain workflow functions initially have lower performance. This applies until the main memory contains all the necessary data.
**Continue workflow after system crash**
You can use this report to select and continue workflows that have had the status *started* for longer than a day. This means that workflows that have come to a halt after system errors can be continued.

**Configure workflow RFC destination**
The workflow runtime system executes its [RFC](#) and tRFC (transactional RFC) calls on a logical destination. You can set up or change a logical destination. No logical destination is configured for the workflow system tRFC calls as standard.

**Upper limits workflow runtime**
You can make the following settings here:
- Maximum number of nested subworkflows permitted at runtime
- Maximum number of nodes that can be processed at runtime within a workflow definition
You can also activate/deactivate the display of agent names in the workflow log.

**Event manager**

**Type linkages and instance linkages**
Tables containing the assignment of events to the event receivers interested in the event. For further information, refer to [Maintenance of Linkage Tables](#).

**Event trace**
The [event trace](#) can be used to log events.

**Event queue**
The [event queue](#) can be used to store events temporarily.

**Workflow definition time**

**Workflow Builder**
Here you can maintain:
- The workflow system administrator with "global" responsibility, that is, responsible for all workflow definitions
- The task executed in a user decision.

If you create a step of the type user decision in a workflow definition, the task entered here is transferred into the workflow definition.

Please refer also to the Customizing activity [Maintaining Administration Data for the Workflow Builder](#).

**WebFlow**

**Customizing Web server**
You use this function to specify the Web server that is configured for your SAP System. For further information, refer to [Defining the Web Server](#).

**SAPforms**

**Mail-enabling**
This report can be used to schedule the sending of R3F messages. This is necessary if you want to execute a work item as a form in an external mail system.
The report is started immediately and then scheduled as a background job in accordance with the period entered.

**Diagnosis**
You have the opportunity to call reports with which you can search for errors that have occurred.

**Trace on/off**
The SAPforms trace logs all SAPforms-related actions.
You have the opportunity to activate/deactivate this trace.
Please also refer to [SAPforms Administration](#).[Seite 1727]

**Activities**
You can access all the workflow system administration reports described by choosing *Tools* → *Business Workflow* → *Development* → *Administration.*
Verifying Workflow Customizing

Use
This function can be used to check the current status of Workflow Customizing and start Automatic Customizing if applicable.

Integration
Automatic Customizing does not cover all areas of Workflow Customizing [Extern]. For example, Auto-Customizing does not cover maintenance of the organizational plan [Seite 1708] or the Customizing of standard tasks and workflow templates [Seite 1710].

Prerequisites
Authorization
You require appropriate authorization for the functions shown. This is the authorization (authorization object S_PROGRAM ) for executing a program for authorization group SWU_TRA1.

Features
The activities checked automatically upon calling are displayed. The result of the check is shown with a graphical symbol. The overall result of the check is also assessed.
If a Customizing activity is shown as having errors, execute Auto-Customizing (see below). You can also double-click on one of the entries displayed to go directly to maintenance of the respective Customizing activity. The description of the Customizing activity is also displayed when you double-click.
After Auto-Customizing, execute the functions listed below (which are located in the Edit menu).

Test RFC destination
Choose Edit → Test RFC destination to test the configuration of the RFC destination. The workflow runtime system executes its tRFC (transactional RFC) calls on this destination for technical reasons.

Automatic Customizing
"Auto-Customizing" was developed to simplify SAP Business Workflow Customizing as far as possible. It is possible to perform all the necessary Customizing activities involving the basic technical settings in one operation.
The following standard settings are made, amongst others:
Configuring a client-specific RFC destination
If there is not one already, a user (including password) is created for the logical destination WORKFLOW LOCAL <client>. Specify a user that has been created in the current client with the user type "background" and that has the authorizations SAP_ALL and SAP_NEW.
Scheduling a background job for deadline monitoring
Scheduling a background job for repeated starting of background steps with temporary errors.
Setting an active plan version
The active plan version is set to "01" if no active plan version has been maintained yet.
Classifying single-step tasks as "general tasks"
The generic decision task (standard task for user decision) and other standard tasks that are used for the workflow scenarios supplied are classified as “general tasks”.
Maintaining a workflow administrator
If no workflow administrator has been maintained yet, your user name (current SY-UNAME) is entered as administrator.

**Start verification workflow**

Choose Edit → Start verification workflow to start a workflow that tests some of the basic functions of the runtime system. Follow the instructions on the screen.

**Activities**

You can access these functions by choosing Tools → Business Workflow → Development → Utilities → Customizing.
Customizing for Tasks and Workflows

Use
In this Customizing activity you activate triggering events of tasks and workflows and assign possible agents to them.

Integration
This Customizing activity enables you to set up centrally in the implementation phase all the tasks and workflows that you will use. However, you can also carry out the steps required individually within the relevant definitions. You only have to call the definition in display mode to carry out the activities described. You do not have to call the definition in change mode.

Prerequisites
You must know:
Which of the tasks and workflows belong to scenarios that you want to use.
The application component to which these tasks and workflows are assigned.

Features
The Customizing of tasks and workflows involves the following activities:

Activation of triggering events
If SAP’s usage of the task or workflow includes starting with a triggering event, the linkage is supplied inactive and you have to activate it if required. This is the only way to restrict the workflow functions to the areas in which you actually want to use them.

Assignment of possible agents
All tasks that are represented by dialog work items [Seite 1372] when they are executed must be assigned to their possible agents [Extern], or be classified as general tasks [Extern]. Whether a task is executed as a single step or as a step in a workflow is immaterial.

All workflows that are to be started in dialog also have to be assigned to their possible agents.

Activities
This function for customizing tasks and workflows can be found in SAP Business Workflow Customizing under Perform Task-Specific Customizing [Extern].
You access the Customizing for particular tasks and workflows via the relevant application component.
Diagnosis of Workflows with Errors

Use
This function establishes all workflows with errors and groups them according to error cause. The evaluation results displayed are runtime information. If the tables are already very full, the report will have a longer runtime (a few minutes).

To be precise, the system establishes *top-level work items* with errors. *Top-level work items* are either workflows or work items of all other types that are not part of a workflow.

Features
The [ABAP List Viewer](#) is used to display the work items with errors.

The possible error causes used by the system for grouping purposes are:
- Agents
- Deadlines
- Binding from workflow to work item
- Other

The following additional functions are available on the application toolbar:
- **Workflow log**
  If you select a work item and choose this function, the relevant workflow log is displayed.

- **Error analysis**
  If you select a work item and choose this function, a dialog box appears listing errors, warnings and notes about the selected work item.

- **Change work item**
  If you select a work item and choose this function, you go to the screen Change work item.

- **Restart after error**
  Choose this function if you have processed a work item with errors and you want to restart it.

Activities
You can access the processing of workflows with errors by choosing *Tools → Business Workflow → Development → Administration → Workflow runtime → Diagnosis of workflows with errors.*
Executing Work Items Without Checking Agents

Use
This function enables a user to execute work items without checking agents. The system does not check whether the user is one of the possible agents [Extern] of the work item to be executed.

Features
You use a selection screen to generate a list of work items. Selection is carried out as in work item selection [Seite 1490]. The following functions are available in the list:

- **Execute without check**: The work item is executed without checking agents.
- **Display work item**: This function branches to the work item display [Seite 1411].
  
  As an alternative, you can **double-click** on the work item.

- **Display workflow log**: This function branches to the workflow log [Seite 1420].
- **Display work item container**: This function branches to the work item container [Seite 1511].
- **Display task**: This function displays the underlying task of the work item.
- **Send...**: The work item is sent as an attachment to a mail.

For information on other functions, refer to the ABAP List Viewer [Extern] documentation.

Activities
You can access this function by choosing Tools → Business Workflow → Development → Administration → Workflow runtime → Execute work items without agent check.
Workflow Restart After Error

Use
This report can be used to display a list of workflows with errors for a particular selection period, and then restart them.
The indicator *Restart immediately* can be used to restart a workflow immediately.

Features
You use a selection screen to generate a list of work items. The following functions are available in the list:

- [Restart workflow]: The selected workflow is restarted.
- [Display work item]: This function takes you to the work item display [Seite 1411].
  As an alternative, you can double-click on the work item.
- [Display workflow log]: This function takes you to the workflow log [Seite 1420].
- [Display work item container]: This function takes you to the work item container [Seite 1511].
- [Display task]: This function displays the underlying task of the work item.

For information on other functions, refer to the ABAP List Viewer [Extern] documentation.

Activities
You can access this function by choosing Tools → Business Workflow → Development → Administration → Workflow runtime → Workflow restart after error.
Deleting Work Items

Use
This report (RSWWWIDE) deletes work items from tables without archiving.

⚠️
Therefore, this report should **not** be used in a production system.

In a production system you **must** use [archive management][Extern] to archive and delete work items in order to ensure data consistency.

Features
On the selection screen, you can set the following indicators in **Technical settings**.

- **Delete immediately**
  
  If you set this indicator, the work items selected are **deleted immediately**.

  Otherwise, a list is displayed in which you can delete work items using the **function**. You can also go to the workflow log by double-clicking.

- **Delete log data as well**
  
  If you set this indicator, the log data for the selected work items is deleted from the workflow log.

- **Number of work items per LUW**
  
  Here you can enter the number of work items to be deleted per database transaction. The setting is database-specific. Refer to the [SAPNet - R/3 Frontend note with the number 107410.][1]

Activities
You can access this function by choosing **Tools** → **Business Workflow** → **Development** → **Administration** → **Workflow runtime** → **Reorganization** → **Delete work item**.
Displaying the Work Item Container

Use
The contents of the container for the relevant work item are displayed in an overall view. You can see the current, runtime-specific data on the specific work item. A distinction should be made between the work item types for which the container is displayed:

Dialog work item [Seite 1372]
The contents of the task container of the task represented by the work item are displayed.

Workflow work item [Seite 1380]
The contents of the workflow container of the workflow represented by the work item are displayed.

Features
In addition to the fields defined and filled by the system, the container also contains any fields created during the definition and filled via a binding. A distinction should be made between:

Constants
Constants are displayed with their value at the time.

Object references
Object references are always specified starting with system and client. These are followed by the ID of the object type and the concatenated key of the object.

You can display the attributes of the referenced object using instance data details.
Displaying XML Documents

Use
You use this function to search for and display XML documents. When a workflow that was to be started by an inbound XML document is not started, this function is the only way to display the XML document.

Integration
XML documents are saved as objects of a work item and can be displayed using the work item display.

Activities
Select an appropriate search period to look for the XML document in question. You can display the XML document from the hit list. The display takes place in the Business Document Navigator [Extern].
Defining the Web Server

Use
You must define the Web server to enable the WebFlow to receive XML reply documents.

Prerequisites
The SAP Internet Transaction Server (ITS) must be installed as well to enable your SAP System to react to XML documents and XML reply documents appropriately.

Procedure
4. On the screen Customizing Web Server: Change, enter the address of your Web server and, if applicable, the port number.

If you want to test the WebFlow in SAP System A and the starting of a Business Workflow in SAP System B, you require the system architecture below:

To check the prerequisites for the test, carry out the following steps:
6. Check whether the release of the ITS installed on SAP Systems A and B is at least 4.6C.
7. Check whether the service [Extern] WF_HANDLER is on the ITS.
8. The user used for the service logon must exist in the SAP System. This user is defined in the global service directory for the ITS or in the service WF_HANDLER.
9. The user defined must be authorized to store documents in the Business Document Service (BDS).
10. The user defined in SAP System B must be one of the possible agents [Extern] of the workflow to be started.

**Result**

The URL required in a Web activity is created with the Web server defined here, as soon as the indicator *Wait for feedback* is set.
Administration of Event Queue

Use
The event queue can be used to store events temporarily in a memory. If the event queue is switched on, certain events are not passed immediately to the event receivers, but are stored temporarily in the event queue instead. This means that the system load caused by a large number of events being created can be spread over a longer time period (which can be set by the workflow system administrator). This combats the threat of system overload.

This is achieved by the transactional RFCs for starting the receivers not being called immediately, but after a time delay and in small numbers.

Using the event queue therefore delays the calling of the receivers.

The event queue can also be used to store events that have errors temporarily.

Integration
The event queue is one of the SAP Business Workflow administration tools [Seite 1500].

Prerequisites
The event queue can only be used for events if the relevant event linkages are appropriately classified. The indicator Enable event queue in type linkage table maintenance [Seite 1358] can be used to do this.

Features

Functions on the application toolbar

- Refresh
- Set default values
- Undo change
- Restore
- Transport administration data
  This function can be used to assign the administration data to a transport request.

- Event trace
  All events created correctly are logged in the event trace [Seite 1366] irrespective of whether potential receivers exist.

- Browser
  The event queue browser [Seite 1526] can be used to view the contents of the event queue.

- Delete events
  This report can be used to delete events from the event queue.
Administration of Event Queue

**Tab page Overview**
When you start event queue administration, the tab page *Overview* is activated. This tab page shows all the statuses of the settings that can be made on the other tab pages. It also shows the current content of the event queue and other statistics.

**Other tab pages**
The following tab pages are available for the actual administration:

<table>
<thead>
<tr>
<th>Tab page</th>
<th>Function</th>
</tr>
</thead>
<tbody>
<tr>
<td>Basic data [Seite 1517]</td>
<td>The administrator and the error feedback behavior are maintained here.</td>
</tr>
<tr>
<td>Activation [Seite 1519]</td>
<td>The event queue is activated here.</td>
</tr>
<tr>
<td>Background job [Seite 1521]</td>
<td>The event queue is read at regular intervals by a background job.</td>
</tr>
<tr>
<td>Event delivery [Seite 1523]</td>
<td>The event delivery specifies how the event receivers are to be started.</td>
</tr>
<tr>
<td>Linkages with errors [Seite 1525]</td>
<td>Those event linkages for which there are events stored temporarily because of errors, or which have an error status are displayed here.</td>
</tr>
</tbody>
</table>

**Activities**
You can access administration of the event queue by choosing *Tools → Business Workflow → Development → Administration → Event manager → Event queue.*
Maintenance of Tab Page Basic Data

Use
You maintain the basic settings for the event queue on this tab page. The administrator and the behavior if there are events with errors are defined here.

Integration
This tab page is maintained within administration of the event queue [Seite 1515].

Features

Administrator
An object of Organizational Management (organizational unit, job, position, work center, user) can be defined as an administrator. This organizational object is notified if no other organizational objects are defined in the relevant context (for example a notification if there are event receivers with errors).

Receiver error feedback
Various errors can occur within an event receiver (for example a workflow started by an event). If an error of this type occurs, feedback can be sent to the event manager.

The runtime behavior in the event of errors can be determined not only in administration of the event queue, but also in maintenance of type linkages [Seite 1358].

The entries on the tab page Basic data work as system presettings. They can be adopted or redefined for each type linkage.

Deactivation of linkage
The linkage for the relevant event receiver is deactivated.
If the linkage is deactivated, this receiver (for example a workflow) is not started for all events that follow. The events are not stored temporarily.
This setting is in accordance with the classic runtime behavior.

Mark linkage as having errors
The event linkage is not deactivated, but marked as Errors.
All events for this linkage are stored temporarily and can be delivered manually. The event linkage must be defined manually as No errors after the error cause has been removed for the receiver to be started again automatically. This can be done either on the tab page Linkages with errors [Seite 1525] or in maintenance of event linkages [Seite 1358]. You can also deliver the events from the queue again on this tab page.

Do not change linkage
If there is error feedback, the linkage is not changed.
The event is stored temporarily for redelivery. If more events follow for this linkage, an attempt is made to start the receiver each time. If an error is discovered, the relevant events are stored temporarily.
In contrast to marking the linkage as Errors, an attempt is made to start the receiver for each event. If there is always an error, the system load is greater than with direct temporary storage of events.
Errors may, however, depend on event parameters. Therefore, only those events for which the error has occurred are stored temporarily with this configuration. Receivers that have no errors based on the event parameters are started immediately.
Activities

You can access administration of the event queue by choosing Tools → Business Workflow → Development → Administration → Event manager → Event queue.
Maintenance of Tab Page Activation

Use
On this tab page you activate the event queue and define the linkages for which the event queue is to temporarily store events and receivers.

Integration
This tab page is maintained within administration of the event queue [Seite 1515].

Features

Settings

Switch on event queue
If the event queue is switched on, all the events created are written into the event queue before delivery, provided delivery via the event queue was specified for them. Delivery via the event queue must be defined separately for each event linkage (see below).
If the event queue is switched off, no events are written into the event queue.

Reading from the event queue does not depend on this setting. Therefore, events are delivered from the event queue even if it is switched off. The background job must be active for this.

Delete event after delivery
The event are stored persistently in the event queue.
You can set whether events are to be deleted after delivery. If the delivered events are not deleted, the indicator delivered is set for them. If delivered events are not deleted, you can still view them even after the delivery.

You should delete delivered events in a production system to accelerate the processing of undelivered events.

Event linkages
The event queue can only be used for events if the linkages are appropriately classified.
You must set the indicator Enable event queue in maintenance of type linkages [Seite 1358].

Active configuration
The information in this status frame answers the following questions:
- Is the event queue switched on?
- Is the background job active?
- Are events deleted automatically after delivery?
- How many event linkages are using the event queue?
Activities

You can access administration of the event queue by choosing Tools → Business Workflow → Development → Administration → Event manager → Event queue.
Maintenance of Tab Page Background Job

Use
The event queue is read at regular intervals by a background job. The events found are delivered to the registered event receivers.

Integration
This tab page is maintained within administration of the event queue [Seite 1515].

Features
You can make the following settings:

- **Operation mode of background job**
  The operation mode can be used to control how often the background job is to be started.
  - **Periodic**
    The background job is scheduled periodically with the specified interval.
  - **Depending on load (dynamic)**
    The intervals at which the background job is to be started can be defined as dependent on the content of the event queue. You should, for example, schedule the background job for a shorter interval if there are still events in the event queue after a processing run. You should choose a longer interval if the event queue does not contain any more events.
    Adapting the interval dynamically can relieve the load on the system, since the background job is started less frequently.
    However, the processing of the events may also be delayed because of the longer interval.
    To switch operation mode, you must first remove the current background job from the schedule.

- **Number of events per read access**
  Here you specify the number of events to be read each time the event queue is read.

- **Time interval between two read accesses**
  Here you specify the time interval between two read accesses. The specification has the following effect:
    - **With periodic processing**
      Processing is carried out with the predefined interval. The interval can only be changed if the background job is removed from the schedule. It is considered when the job is rescheduled.
    - **With processing that depends on load**
Maintenance of Tab Page Background Job

The time interval is adhered to if the event queue contains more events after a processing run. The interval can even be changed when the background job is scheduled, since the job reschedules itself after every processing run (depending on the content of the queue).

- **Interval until event queue next checked**
  
  This interval is adhered to if there are no more events after the queue has been read. This setting can only be made with processing that depends on load.

- **Schedule background job**
- **Display background job**
- **Unschedule background job**

**Activities**

You can access administration of the event queue by choosing Tools → Business Workflow → Development → Administration → Event manager → Event queue.
Maintenance of Tab Page Event Delivery

Use
On this tab page you maintain how the event receivers are to be started.
A defined number of events is delivered by the background job in each cycle. These events can be processed sequentially or in parallel (with aRFC).
The receivers can be started synchronously (with RFC) or asynchronously (with tRFC).

Only synchronous starting of receivers is currently supported for parallel processing.

Integration
This tab page is maintained within administration of the event queue [Seite 1515].

Features
Processing of events

Sequential
Events are delivered from the event queue in defined units (for example five events per read access).
Sequential processing means that these events are processed in succession.

Parallel on server group
Events are delivered from the event queue in defined units (for example five events per read access).
Parallel processing means that the events are processed at the same time in separate processes. These parallel processes can run using a defined server group.

A server group combines a number of application servers to form a logical group. If you specify a server group, the event delivery is executed using the application servers in this group.
If you do not specify a server group, the parallel execution is distributed across all application servers. You are recommended to use a server group with the appropriate resources.

Start of receivers

Synchronous
Various receivers (for example workflows) can be started for an event. Synchronous start of receivers means that they are started using a remote function call (RFC). The processing waits until the receiver call is complete.

Asynchronous
Various receivers (for example workflows) can be started for an event. Asynchronous start of receivers means that they are started using a transactional remote function call (tRFC). The processing does not wait until the receiver call is complete, but is continued immediately instead. The receivers are started asynchronously.
Sequential processing of events with synchronous start can be used for test purposes.

The delivery mode recommended in the standard version is sequential processing of events with asynchronous start of receivers.

Parallel processing (with synchronous start) can be used to distribute the system load on a predefined server group. Delivery then only uses the application servers assigned to this server group. If you do not specify a server group, the parallel execution is distributed across all application servers. You are recommended to use a server group with the appropriate resources.

**Activities**

You can access administration of the event queue by choosing *Tools → Business Workflow → Development → Administration → Event manager → Event queue.*
Maintenance of Tab Page Linkages With Errors

Use
If an error occurs within an event receiver, this error can cause the event data to be lost and prevent the receiver from starting. To enable the event to be delivered again after an error, the event data can be entered into the event queue.

Integration
This tab page is maintained within administration of the event queue [Seite 1515].

Features
Those event linkages for which there are events stored temporarily because of errors, or which have an error status are displayed in the overview.
Temporarily stored events can be initiated again by selecting a linkage and then starting the delivery (function Start delivery). You can specify whether you want to deliver one event or all events (indicator Only deliver one event).
The option of delivering a single event is intended to save resources in the event of another error.
Irrespective of that, events can be redelivered immediately or by the event queue background job. The background job [Seite 1521] must be active for this.
The linkage between the event and the receiver can be set to status No errors by selecting the symbol for the linkage that has an error in the column Status of linkage.
Events can be deleted from the list Linkages with errors using Events. This always deletes all events.
The indicator Only deliver one event is not evaluated.
The function Delete events is available in the application toolbar to delete individual events.

Activities
You can access administration of the event queue by choosing Tools → Business Workflow → Development → Administration → Event manager → Event queue.
Event Queue Browser

Use
The event queue browser displays not only the current content but also a history of the content of the event queue. This is dependent on the selection criteria chosen.
Event container, event trace and workflow log can be displayed from the selected list. Events not yet delivered can also be delivered directly.

Integration
Which events are written into the event queue is defined in the administration environment [Seite 1515] of the event queue.

Features

Selection area
In the selection area you can select events for display in the browser.
If you restrict the search range, the symbol appears next to the pushbutton Standard criteria.
You can "fine-tune" the selection via the pushbutton Further criteria. The symbol appears here too if you enter selection criteria.

Selected list
The list is refreshed automatically every time the selection conditions are changed. An event can be displayed from the list by double-clicking. The general event data and the event container are displayed.

Functions on the application toolbar
- Refresh
Event Queue Browser

- **Event**
  The selected event is displayed.

- **Deliver**
  The selected event is delivered, that is, it is provided to the relevant receiver.

- **Event trace**
  All events created correctly are logged in the event trace [Seite 1366] irrespective of whether potential receivers exist.

- **Work Item**
  This function takes you to the work item display [Seite 1411] of the work item operating as receiver of the selected event.

  You can only go to the work item display if the event has been written into the event trace.

**Activities**

You can access the event queue browser by choosing Tools → Business Workflow → Development → Utilities → Events → Event queue browser.
Transport and Client Copy

Process Flow

Transport of cross-client tasks
As cross-client transport objects, standard tasks [Extern] (task type TS) and workflow templates [Extern] (task type WS) are always connected to the transport system and are automatically included in a transport request, if the client settings (table T000) allow changes to cross-client objects.

Transport of client-specific tasks
Customer tasks [Extern] (task type T) and workflow tasks [Extern] (task type WF) are only included automatically in a transport request in a particular client if Automatic recording of changes is set for this client in table T000.
In other clients, you can include customer tasks and workflow tasks in a transport request manually. To do this, choose Task → Transport or Workflow task → Transport in the relevant definition environment.

Transport of settings for tasks
You maintain the following settings for tasks on a client-specific basis:
- Assignments between tasks and their possible agents
- Event receiver linkages and their activation
- Workflow extension
The system automatically includes these settings in a transport request if Automatic recording of changes is set for the relevant client in table T000.

Client copy
With a client copy [Extern], the assignments between tasks and their possible agents and the event receiver linkages are copied.

The event receiver linkages are always deactivated in the target client after copying.
If you want to copy the activation indicator of each individual event receiver linkage as well, you must specify this explicitly as a parameter option for copying tables of class A.

Event linkages with installation or upgrade
With a new installation or an upgrade, the event receiver linkages are copied into all target clients. But the activation entries are only copied into client 000. Any existing entries are overwritten in this operation.
Authorization Management

Use
The authorizations are maintained in authorization management. You create roles there, which contain the individual authorizations. There are roles, authorization profiles and authorization objects specifically for SAP Business Workflow. These can be used for creating your own roles. For further information, refer to BC - Users and Roles.

Activities
You should use roles for allocating authorizations. The following roles are available:

Roles for Business Workflow

<table>
<thead>
<tr>
<th>Role</th>
<th>Technical name</th>
<th>Explanation</th>
</tr>
</thead>
<tbody>
<tr>
<td>Non-critical, basic authorizations for all users</td>
<td>SAP_BC_ENDUSER</td>
<td>Basic authorizations for all workflow participants</td>
</tr>
<tr>
<td>User for communication, workflow, appointments, etc.</td>
<td>SAP_BC_SRV_USER [Extern]</td>
<td>Authorizations for a workflow agent [Seite 805] and basic authorizations for all workflow participants</td>
</tr>
<tr>
<td>Business Workflow implementation team</td>
<td>SAP_BC_BMT_WFM_PROCESS [Extern]</td>
<td>Authorizations for a process consultant [Seite 96]</td>
</tr>
</tbody>
</table>
Administration of Organizational Management

Use
There are settings for controlling the interaction between Organizational Management functions and SAP Business Workflow functions.

Features
You need only learn more about these settings if you are planning further administration or consulting activities.
They are:

Table settings

Workflow-relevant data for Organizational Management
For further information, refer to Workflow-Relevant Data for Organizational Management [Seite 1533].

Permitted relationships
Table T777E contains all the relationships permitted between organizational objects.
In connection with the substitution rules, relationship 210 (substituted with profile by or substitutes with profile) is used to determine which object types (user, person, position) may be specified as a substitute [Seite 1398] for a position.
This table is supplied with content and can be processed in Personnel Management Customizing (Personnel Management → Organizational Management → Basic settings → Maintain relationships) or using transaction OOVK.

SAP organizational object type assignment
Table T7791 contains information about which SAP organizational object types should be linked with positions, which with jobs and which with organizational units.
This table is supplied with content and can be processed in SAP Business Workflow Customizing (Basis Components → Business Management → SAP Business Workflow → Basic Settings (Organizational Management) → Maintain assignments for SAP organizational object types) or using transaction 0091. For further information, refer to Agent as Attribute of SAP Organizational Object [Seite 1302].

Prefix numbers for standard object types
Table T78NR stores the 3-digit prefix numbers used for the first 3 digits of numbers of standard tasks, multistep tasks and roles, on a system-specific and client-specific basis. The remaining 5 digits of the 8-digit numbers are allocated by internal number assignment.
These values are set in SAP Business Workflow Customizing or in automatic Customizing.

⚠️ Number assignment for customer tasks is not carried out as described above, but with a separate number range object.

The table is supplied with content and can be processed in SAP Business Workflow Customizing (Basis Components → Business Management → SAP Business Workflow → Basic Settings (Organizational Management) → Maintain prefix numbers) or using transaction OOW4.

Permitted object types for role resolution
As a check table, table T7790 contains all the organizational object types [Exter] that can be established as the result of role resolution.
The table is maintained via transaction SM30.

**Reports for PA administration and evaluation**

**Report RHRELAT0 (permitted relationships for object types)**
All the permitted relationships for one organizational object are displayed.
The field *Object type* must contain a permitted abbreviation for an object type.
The field *Relationship type* can remain blank, in which case all relationships in both possible directions are displayed.

**Report RHSHOWOR (display organizational assignment)**
The report displays the organizational environment of a user. The following information is displayed:
- Organizational unit to which the user is assigned.
- Position the user occupies.
- Job that describes the position.
- Tasks that are assigned directly to the organizational unit, the position, the job or the user.

**Report RHSTRU00 (display/maintain structure)**
The report goes into the structure of the organizational plan from the object entered and displays all the objects linked via the evaluation path with the object concerned. The position of the object displayed in the hierarchy is illustrated using indents.
An organizational object is specified with *plan version, object type* and *object ID* (user name for object type US) on the selection screen for this report.
In addition, an evaluation path is entered in the relevant field. You can establish evaluation paths for the objects used using the F4 help.
It is highly recommended that you set the *Recursion check* indicator to avoid endless loops.

**Number range objects**

**Number range object HRSOBJECT (standard objects)**
Number range [Extern] for standard objects.
There is an eight-digit number for:
- Standard tasks (TS)
- Standard roles (AC)
- Workflow templates (WS)
  It is made up of two parts:
  iii. Three-digit prefix number set per client and system in table T77NR.
  iv. Five-digit sequence number assigned internally according to the settings in number range object HRSOBJECT.

**Number range object HRWFOBJECT (PD workflow objects)**
Number range for PD workflow objects.
The eight-digit number for workflow tasks (WF) is made up of two parts:
3. Three-digit prefix number set per client and system in table T77NR.
4. Five-digit sequence number assigned internally according to the settings in number range object HRWFOBJECT.
Number ranges are maintained from the ABAP Workbench via *Development → Other tools → Number ranges*.

**Number range object RP_PLAN**

Number assignment for
- customer tasks (T)
- organizational units, positions, jobs and work centers

is carried out according to the settings in the number range object RP_PLAN. As standard, the interval from 50000000 to 99999999 is set for subgroup $$$ (valid for all plan versions and all object types) with internal number assignment.

You can change this setting in Personnel Management Customizing (*Personnel Management → Organizational Management → Basic Settings → Number Range Maintenance*) or by using transaction OONR.

**Evaluation paths**

For further information, refer to [Evaluation Paths][1].
Workflow-Relevant Data for Organizational Management

Use
You can maintain workflow-relevant data for Organizational Management via table T77S0.

Features

Change active plan version
Table T77S0 contains the value of the active plan version. This value is set in Customizing. In SAP Business Workflow automatic Customizing, the value 01 is entered for the active plan version if the field had no value. This setting can be processed via transaction OOAP.

Work item display for substitutes
Table T77S0 contains an indicator specifying whether the substitute should automatically see the additional work items when a substitution is activated. This indicator should be set. This setting can be processed via transaction SM30. The indicator must be set for AUTOF.

Default value for task classification
Table T77S0 contains an entry for the default value for task classification. The default value is NO_CLASS. This setting can be processed via transaction SM30. The entry must be made for DEFCL.

Buffering in the Organizational Management environment
Table T77S0 contains information about the buffering of tasks. You can refresh the buffer manually with the report RHWINDEXRESET. The buffer is refreshed automatically once a day.

The buffer must be refreshed manually for changes to a workflow definition to take effect immediately. Refreshing does, however, have a negative influence on performance.

Activities
You can access automatic Customizing by choosing Tools → Business Workflow → Development → Utilities → Customizing
Refer also to: Automatic Customizing [Extern]
Evaluation Paths

Definition
An evaluation path describes a chain of relationships that exists between individual organizational objects in the organizational plan.

Use
Evaluation paths are used in connection with the definition of roles [Extern] and views.

The evaluation path O-S-P describes the relationship chain Organizational unit > Position > Employee.

Evaluation paths are used to select other objects from one particular organizational object. The system evaluates the organizational plan along the evaluation path.

Starting from an organizational unit, evaluation path O-S-P is used to establish all persons who belong to this organizational unit or subordinate organizational units via their positions.

Integration
Evaluation paths are defined in the standard system. The evaluation paths available include both elementary evaluation paths with just one relationship (for example A003 belongs to, B007 is described by), and complex evaluation paths (for example O-S-P internal persons per organizational unit, as described above).

You can use the evaluation paths available or define your own.

Evaluation paths can be tested and analyzed using the report RHSTRU00 [Seite 1530].

See also:
Views [Seite 1402]
Maintaining Evaluation Paths

5. To maintain evaluation paths, go to the relevant activity "Maintain evaluation paths" in Organizational Management Customizing. During operation use transaction OOAW.

Here you will find the maintenance view Change evaluation paths: Overview.

6. Assign a name to the evaluation path (maximum of 8 alphanumeric characters).

7. Go to the dependent view Evaluation paths (individual maintenance).

8. Describe the relationship chain to be used for the evaluation.

To establish the permitted relationships for an object type, use report RHRELATO.

When using an evaluation path in a view, you should consider the following:

- Define the evaluation path in such a manner that the relationship chain always starts from a user (object type US in Organizational Management) and ends at an organizational unit, a position or a user.

- When defining the evaluation path, use the Skip indicator (see below) in order not to overload the result of the evaluation.

When using an evaluation path in a role, you should consider the following:

- Define the evaluation path in such a manner that the relationship chain starts from the object type of Organizational Management, which you can pass in the role parameter container.

- You can select the Skip field to determine that this part of the evaluation path should be taken into account but not output.

6. You must always enter * in the field Priority.

Before creating new evaluation paths, check the evaluation paths available as standard.

Starting from an organizational unit, evaluation path O-S-P establishes all persons who belong to this organizational unit or subordinate organizational units via their positions. Evaluation path O-S-P is defined as follows:

<table>
<thead>
<tr>
<th>No.</th>
<th>Object type</th>
<th>A/B</th>
<th>Relationship</th>
<th>Relationship name</th>
<th>Priority</th>
<th>Type</th>
<th>Skip</th>
</tr>
</thead>
<tbody>
<tr>
<td>10</td>
<td>O</td>
<td>B</td>
<td>003</td>
<td>incorporates</td>
<td>*</td>
<td>S</td>
<td></td>
</tr>
<tr>
<td>20</td>
<td>S</td>
<td>A</td>
<td>008</td>
<td>holder</td>
<td>*</td>
<td>P</td>
<td></td>
</tr>
<tr>
<td>30</td>
<td>O</td>
<td>B</td>
<td>002</td>
<td>is line supervisor of</td>
<td>*</td>
<td>O</td>
<td></td>
</tr>
</tbody>
</table>

The relationship defined in the last line ensures that user assignments in subordinate organizational units are included in the evaluation path as well. Since the Skip indicator is not selected for any of the lines, evaluations will always display all the objects established.

Starting from a person or a user, evaluation path WF_ORGUN first establishes the relevant position and then the organizational unit to which this position belongs. Evaluation path WF_ORGUN is defined as follows:

<table>
<thead>
<tr>
<th>No.</th>
<th>Object type</th>
<th>A/B</th>
<th>Relationship</th>
<th>Relationship name</th>
<th>Priority</th>
<th>Type</th>
<th>Skip</th>
</tr>
</thead>
<tbody>
<tr>
<td>10</td>
<td>US</td>
<td>A</td>
<td>208</td>
<td>is identical to</td>
<td>*</td>
<td>P</td>
<td>x</td>
</tr>
</tbody>
</table>
This evaluation path works both when the positions are occupied by users and when the positions are occupied by persons who are then assigned to users.
This evaluation path can be used with a user or a person as the starting point.
Since the *Skip* indicator is selected for the first two lines, evaluations only display the organizational units found most recently.
SAP Business Workflow via the Internet

Use
Workflow functions can be called via a Web browser using an Internet Transaction Server (ITS). This enables employees who seldom or never work with a SAP GUI for Windows to access workflow functions.

Integration
SAP provides not only the option of generating your own Web transactions, but also predefined Web scenarios known as Easy Web Transactions (EWTs).

Prerequisites
You need an ITS to work with Web integration. You also need the SAP@WebStudio.

Features
Integration of Web forms
*SAP Business Workflow* Web integration supports starting workflows and executing work items.

An employee creates a notification of absence in a Web browser using a Web form. Either this process starts a workflow, or the workflow has already been started by the form being called.

The employee's superior then receives a work item in their Web workplace (a replication of their Business Workplace [Seite 1368] in the R/3 System), in which they can approve or reject the request.

The employee is informed of the result via their Web workplace, for example with a mail.

In both situations a Web transaction is generated from a task. These transactions must then be processed with the SAP@WebStudio. Finally, the service and HTML templates generated must be published on the relevant ITS. This is also done using the SAP@WebStudio.

The object type *Form* with method *HtmlProcess* is available for executing work items.

Easy Web Transactions
The following Easy Web Transactions are supplied in the SAP Business Workflow context:

*Business Workplace via the Internet (Web Workplace)* [Seite 1542]
*Workflow Status Reports via the Internet* [Seite 1546]
*Creating a Customer Master Record via the Internet* [Seite 2315]
Creating a Customer Master Record on the Internet (BC-BMT-WFM SAPforms)

Use
This Easy Web Transaction demonstrates how the workflow functionality is integrated in the Web using the SAPforms interface. This is illustrated by creating a customer master record. After a workflow has been started with a Web form (create customer master record), a work item is generated (check creation) which, in turn, is processed with a Web form. Unlike other Easy Web Transactions, two independent services are required here and must be developed separately. The first service starts a workflow, while the second executes a work item. Using both services in one workflow is simply one possibility here and is not mandatory.

You can use the Easy Web Transaction for training purposes or as a template to create your own Easy Web Transaction. This Easy Web Transaction only represents a simplified version of the Create customer master record functionality. First of all, the Web form that starts the workflow does not contain all of the options you can use to create a customer. Secondly, the flow logic of the workflow is extremely basic.

Type of Internet Application Component
Intranet application.

Prerequisites
To allow the workflow WS20000102 to be started on your system and the task TS20000256 to be executed, you should declare both as a general task. In this way, you can ensure that everyone can start the workflow. Since the (default) agent responsible for the task TS20000256 is the workflow system variable _WF_INITIATOR, the person who entered the data is automatically the person who checks it. This allows you to work through this scenario with only one user logged on to the system.

Standard Settings and Preassigned Data
When you work with the Web form for the workflow WS20000102, the following fields can be filled:

<table>
<thead>
<tr>
<th>Address Data</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Form of address</td>
<td>Selection field</td>
</tr>
<tr>
<td>First name</td>
<td>Optional</td>
</tr>
<tr>
<td>Last name</td>
<td>Required field</td>
</tr>
<tr>
<td>Street</td>
<td>Optional</td>
</tr>
<tr>
<td>Postal code</td>
<td>Required field</td>
</tr>
<tr>
<td>City</td>
<td>Required field</td>
</tr>
<tr>
<td>Country</td>
<td>Selection field</td>
</tr>
<tr>
<td>Language</td>
<td>Selection field</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Communication Data</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>1. Telephone number</td>
<td>Optional</td>
</tr>
<tr>
<td>2. Telephone number</td>
<td>Optional</td>
</tr>
<tr>
<td>Fax number</td>
<td>Optional</td>
</tr>
<tr>
<td>E-mail (administrator)</td>
<td>Optional</td>
</tr>
</tbody>
</table>
Organizational Data

<table>
<thead>
<tr>
<th>Field</th>
<th>Requirement</th>
</tr>
</thead>
<tbody>
<tr>
<td>Sales organization</td>
<td>Must be preassigned</td>
</tr>
<tr>
<td>Distribution channel</td>
<td>Must be preassigned</td>
</tr>
<tr>
<td>Division</td>
<td>Must be preassigned</td>
</tr>
<tr>
<td>Reference customer</td>
<td>Must be preassigned</td>
</tr>
</tbody>
</table>

You must define a preassigned value for the sales organization, distribution channel, division, and reference customer fields. In order to do so, you must enter your company-specific data in the HTML template SWU5WS20000102H_150 in the SAP@WebStudio. The existing data in the template only serves as a placeholder.

To define default organizational data, open the HTML template SWU5WS20000102H_150 (service name TS_WS20000102H) in the SAP@WebStudio. The template coding (with comments) is displayed in the right-hand SAP@WebStudio window. The section of the source text in which you can change the organizational data starts after the comment CUSTOMIZING SECTION - BEGIN.

If your sales organization has the value “1234”, for example, enter this value in the template by changing the line after the first PLEASE CHANGE THE VALUE... as follows:

Before: ... "SALESORGANIZATION150150" value = "0001"...
After: ... "SALESORGANIZATION150150" value = "1234"...

You can change the distribution channel, division, and reference customer values in the same way by modifying the next three sections that start with PLEASE CHANGE THE VALUE....

Features

Options

Firstly, the flow logic of the workflow can be extended. In this way, a rejected customer master record could be returned to its creator for resubmission. In this case, the existing workflow must be copied. Secondly, the Web form that starts the workflow can be extended to include the input options that are also available for creating a new customer master record but were not included in this Easy Web Transaction. For more information on controlling workflows with Web forms, please refer to Web form integration [Seite 2284].

Service Name

The service names of this Internet Application Component are TS_WS20000102H and TS_TS20000256H. All of the associated files are stored under these service names in the SAP@WebStudio.

R/3 Development Objects

The following development objects are relevant to this Easy Web Transaction:

<table>
<thead>
<tr>
<th>Development class</th>
<th>Task</th>
</tr>
</thead>
<tbody>
<tr>
<td>Workflow</td>
<td>Task</td>
</tr>
<tr>
<td>VSSF</td>
<td>WS20000102</td>
</tr>
</tbody>
</table>
## Creating a Customer Master Record on the Internet (BC-BMT-WFM SAPforms)

<table>
<thead>
<tr>
<th></th>
<th>TS_WS20000102H</th>
<th>TS_TS20000256H</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Web transaction</strong></td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>Module pool</strong></td>
<td>SWU5WS20000102H</td>
<td>SWU5TS20000256H</td>
</tr>
<tr>
<td><strong>BAPIs</strong></td>
<td>BAPI_CUSTOMER_CREATEFROMDATA</td>
<td></td>
</tr>
</tbody>
</table>
Process Flow (BC-BMT-WFM Creating a Customer Master Record)

This Easy Web Transaction demonstrates how the workflow functionality is integrated in the Web. This is illustrated by creating a customer master record.

Process Flow

The following process is executed when you use the Create a customer master record (BC-BMT-WFM) Easy Web Transaction:

The New Customer form is displayed when you log on to the ITS (Internet Transaction Server) on the corresponding R/3 System (service name TS_WS20000102H).

Complete all of the required fields in the form and choose Submit data. Choose Reset to delete the entries.

The next screen indicates the work item ID under which the task was started. Acknowledge that the task has started.

The employee responsible for checking the customer master record receives a work item in their Web workplace. With this Easy Web Transaction, this person is you (_WF_INITIATOR). Execute the work item.

The form you completed previously is then displayed for you to accept or reject. You can also cancel this procedure if necessary. If you do so, you can execute the work item again later.

The next screen informs you that the work item has been executed. Acknowledge the screen.

Result

If you accept the data, the workflow definition calls a BAPI in the background to create the master record. The master record is created if the data is consistent. Otherwise the BAPI will generate an error message and the master record will not be created.

In each case, a message appears in your Web inbox informing you of the result of the action.
Business Workplace via the Internet

Use
This Easy Web Transaction gives you access to your Business Workplace. The following folders are available:

<table>
<thead>
<tr>
<th>Folder</th>
<th>Function</th>
</tr>
</thead>
<tbody>
<tr>
<td>Inbox</td>
<td>Here you will find all documents sent to you, work items assigned to you and resubmissions.</td>
</tr>
<tr>
<td>Outbox</td>
<td>Here you will find an overview of and additional information on documents you have sent, workflows you have started and work items you have executed or forwarded.</td>
</tr>
<tr>
<td>Private folders</td>
<td>This is a folder structure for managing your own documents, messages, distribution lists and work processes.</td>
</tr>
<tr>
<td>Shared folders</td>
<td>This is a folder structure for publication and management of information on an enterprise-wide or group-specific basis.</td>
</tr>
<tr>
<td>Resubmissions</td>
<td>This provides temporary storage for the documents and work items to be resubmitted into your inbox at some point in the future.</td>
</tr>
<tr>
<td>Trash</td>
<td>This provides temporary storage for the folders and documents you have deleted. This folder gives you the opportunity to undo the deletion.</td>
</tr>
</tbody>
</table>

The Business Workplace provides access to the R/3 mail and folder system as well as SAP Business Workflow end user functions. This enables you to create, process and send documents as well as display and process work items.

The mail and folder functions enable you to display documents and folders, and create, send and process documents or short messages.

Unlike in the R/3 System, although you can display and process documents of every class including their attachments, you can only create text documents. The full subfolder structure of a folder is not displayed, but individual levels through which you can then navigate.

The workflow functions enable you to display and execute work items. After logging on to the R/3 System via the Internet, you can display your worklist. This list of work items is identical to the list that the employee would see if they logged on to the R/3 System directly.

Prerequisites

Authorization/security
An employee who is to have access to their Business Workplace via the Internet needs the authorizations of a Workplace end user. This authorization is in the role SAP_BC_SRV_GBT_USER.

Features

Service name
The service name of this Easy Web Transaction is BWSP. The relevant files are located under this service name in the SAP@Web Studio.
Business Workplace

Use
You can do the following:
Process work items.
Receive and send messages.
Store documents.

Activities
Creating, Processing and Sending Documents [Seite 1545]
Displaying and Processing Work Items [Seite 1544]
Displaying and Processing Work Items

Select the folder *Workflow*.

Select a work item from the worklist.

The functions that can be executed on the work item are shown. These depend on the status of the work item and the properties of the underlying task.

**Execute:** If the execution is aborted, the work item has the status `reserved`.

  - **Reserve:** The work item can no longer be seen by any other recipients.

**Replace:** The work item can now be seen by all recipients again.

**Attachments:** All the functions referring to the attachments of the work item can be accessed via the function Attachments: Create, display, change, delete attachment.

**Forward:** Another agent, who until now has not been a recipient of this work item, now sees the work item in their Business Workplace.

**Reject execution:** This function must be incorporated explicitly and modeled with subsequent steps in the definition of this step.

**Set to 'Done':** This function must be incorporated explicitly in the definition of the single-step task.

If a function is not possible because of the current constellation, the pushbutton is not displayed. In particular, some functions are only possible once the work item has been reserved.
Creating, Processing and Sending Documents

Select a folder.

The following processing options are available:

To create a short message, a document or a folder, choose Create short message, document or Create folder.

To display a short message, a document or a folder, select it from the list by single-clicking.

Which functions you can select depends on the folder in which you are located.

In the screen area in which the content of the document is displayed, you have the following processing options:

**Short message:** Specify a recipient and choose Find so that the recipient type is entered automatically. Select Express if applicable and enter a title and the text. Choose Send. The message is sent to the recipient and stored in your outbox.

**Document:** Enter a title and a text and choose Save. You can process the document with the following functions:

**Create attachment:** If you want to create a text document as an attachment, specify the title of the document. If you want to create a PC document as an attachment, specify its file path and name or choose Browse in order to select the PC document from the folder hierarchy of your PC.

- **Send:** Specify a recipient and choose Find so that the recipient type is entered automatically. Select Express if applicable.

**Resubmit:** Specify a resubmission date and choose Save.

**Move** or **Create link:** Your private folders are displayed. Select the required folder. After moving the document, it is only available in the selected folder. When creating a link, you get the same document in the original folder and in the selected folder, so that changes to this document will be visible in both folders.

**Delete:** The document is moved to the trash. You can retrieve it or delete it permanently there.

**Recipient list and transmission information:** With short messages and documents that have been sent, you can display the recipient list by selecting Recipients in the document display. This also provides information as to which of the individual recipients have read the document and when.

Back [Seite 1543]
Workflow Status Reports via the Internet (BC-BMT-WFM)

Use
Employees in your enterprise whose work centers are not on the premises or who are often away on business trips do not always have direct access to R/3. As a result, these employees are often isolated from the information processes within the enterprise, which affect them. SAP now offers your employees the opportunity to perform selected parts of workflow reporting via the Internet:

Individual item of information: Display of all work items that are linked to a particular object.

Business processes: Display of all work items that are linked to a particular object type.

Decisions: Display of completed work items for a particular task. (In this case, work items for the user decision in the WWW scenario purchase requisition release.)

For demonstration purposes, the Easy Web Transaction Workflow status reports is configured so that the evaluations described above refer to the object type Purchase requisition (BUS2009) and its objects. But the Easy Web Transaction can easily be configured for other object types as well. This documentation contains the relevant information.

An employee can access the workflow information they require from any Internet connection. It is no longer necessary to install the R/3 frontend component and set up a connection to the R/3 System. This helps your external employees who are often off the premises. The worldwide availability of the Internet also means access to the R/3 System at all times, without having to set up a time-limited connection to the “home” R/3 System via the telephone network.

There are also advantages for your internal employees who do not use any R/3 functionality other than querying status reports. The user interface of an Internet browser is often the familiar user interface to these users. They now no longer have to leave this to use workflow functions.

Type of Easy Web Transaction
Intranet

Prerequisites
Basic knowledge of SAP Business Workflow. For further information, refer to the documentation SAP Business Workflow [Seite 1711].

Authorizations/security
The reports that are executed via the Internet are each assigned to an authorization group:

Authorization group SWI_WLA: Decisions: Purchase requisition releases/rejections

Authorization group SWI_OA: Individual item of information: Status of a purchase requisition, business processes: Purchase requisition approval procedure

To be able to start a report a user must have the authorization to execute reports from the relevant authorization group.

This authorization is an authorization based on the authorization object S_PROGRAM (ABAP: Program flow checks), for which the following entries are made in the following fields:

SUBMIT in the field User action ABAP program.

The relevant authorization group in the field Authorization group ABAP program.

The following authorizations based on the authorization object S_PROGRAM (ABAP: Program flow checks) are preconfigured accordingly:
Workflow Status Reports via the Internet (BC-BMT-WFM)

Authorizations based on authorization object S_PROGRAM

<table>
<thead>
<tr>
<th>Authorization</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>S_WF_INF_OA</td>
<td>Select work items for a business object</td>
</tr>
<tr>
<td>S_WF_INF_WLA</td>
<td>Execute workload analysis</td>
</tr>
</tbody>
</table>

“SUBMIT authorization” for reports from the authorization group SWI_OA
“SUBMIT authorization” for reports from the authorization group SWI_WLA

Include the above authorizations into the authorization profile of the employee who is to execute the SAP Business Workflow selection reports via the Internet.

Standard presets and predefined data

The following reports are executed via the Internet in the R/3 System for the purchase requisition example:

**Individual item of information:** Display of work items that are linked to a particular purchase requisition.

- Report RMW3OINS (individual item of information: status of a purchase requisition)

**Business processes:** Display of work items that are linked to the object type purchase requisition (BUS2009).

- Report RMW3OTYP (business processes: purchase requisition approval procedure)

**Decision:** Display of completed work items.

- Report RMW3DONE (decisions: purchase requisition releases/rejections)

Features

Adaptations

In the supplied reports RMW3OINS (individual item of information: status of a purchase requisition), RMW3OTYP (business processes: purchase requisition approval procedure) and RMW3DONE (decisions: purchase requisition releases/rejections) the reference for the object type BUS2009 and the decision task is predefined in the program coding. These reports should be seen as templates for the reports that you must program to extend and adapt the Easy Web Transaction to meet your requirements.

Copy the relevant report and make the following changes:

**Copy of report RMW3OINS (individual item of information):**

- In the data declaration section, replace the object type BUS2009 with the object type of your choice (line 6) and declare the key fields of this object type as PARAMETERS (lines 3 and 4).
- If there is more than one key field, concatenate the key fields into the variable OBJKEY (line 9).

**Copy of report RMW3OTYP (business processes):**

- In the data declaration section, replace the object type BUS2009 with the object type of your choice (line 5).

**Copy of report RMW3DONE (decisions):**

- In the SUBMIT call of the encapsulated report, replace the task TS40007902 with a task of your choice (line 14).

Note that many of the texts that appear on the interface are text elements belonging to the report RMW3OINS.
All reports have a modular structure and each calls a report that executes the actual database selection. Do not change these reports!

**Settings in the SAP Web Repository**
Create your own HTML templates in the SAP Web Repository (transaction SMW0) in your R/3 System. Use the HTML templates `WORKFLOW_STATUS_F*` as models. These can be selected via development class ME.

**Service name**
The service name of this Easy Web Transaction is SWRP. All the relevant files are located under this service name in the SAP@WebStudio.

**Adapting the templates**
You must also adapt the templates that are stored on your Web server in the directory `itsmnt\templates\swrp` (= service name).
Execution of Selection Reports (BC-BMT-WFM Status Reports)

Purpose

The Easy Web Transaction Workflow status reports enables you to execute certain selection reports.

The process flow described here uses the object type Purchase requisition (BUS2009) and can be used as a model for your own development.

Process Flow

Individual item of information: Status of one purchase requisition

All top level work items that relate to the purchase requisition item to be entered are displayed with type, status, start date and work item text.

Top level work items are work items that have no superordinate work items. The following are top-level work items:

Workflow work items (type F) that are not a component of a superordinate workflow (as a subworkflow). The individual work items of a workflow are not displayed since the superordinate workflow is displayed.

Dialog work items (type W) that are not a component of a superordinate workflow but were started as single steps.

Background work items (type B) that are not a component of a superordinate workflow.

Selection parameters for the individual item of information, relating to purchase requisitions are:

Purchase requisition number

Item number

You can go to the workflow log of a displayed workflow by double-clicking on the work item text. In the workflow log, the individual work items of this workflow with creation date, creation time, type, status and description are displayed.

Technical details

A purchase requisition relates to a work item if it is referenced in any element in the work item container or workflow container.

This selection report is one example of querying the processing status of a particular object via the Internet. The report can be used as a model for your own development.

Business processes: Overview of purchase requisitions

All top level work items that relate to the object type purchase requisition (BUS2009) and have been created since the date entered are displayed with type, status, start date and work item text.

The only selection parameter for business processes, relating to purchase requisitions, is the date. All selected workflows were created after this date.

The selection parameter is optional.

The number and item of the purchase requisition are immaterial for this selection.
You can go to the *workflow log* of a displayed workflow by double-clicking on the work item text. In the *workflow log*, the individual work items of this workflow with *creation date, creation time, type, status* and *description* are displayed.

**Technical details**

A purchase requisition relates to a work item if it is referenced in any element in the work item container or workflow container.

This selection report is one example for obtaining an overview of an object type via the Internet. The report can be used as a model for your own development.

**Decisions: Completed work items**

All dialog work items with status *completed*, which represent the decision task *TS40007902 (decision about purchase requisition via WWW)* and have been created since the date entered, are displayed with their current agent and the processing duration. This decision task is used in the user decision in the *WWW scenario purchase requisition release (workflow WS40000001)*.

The only selection parameter for *decisions*, relating to purchase requisitions, is the *date*. All selected work items were created after this date.

The selection parameter is optional.
Web Form Integration

Use
Web forms allow users to communicate with the R/3 System via the Internet using their Web browser. Using the Web forms you can
Start workflows (Starting workflows with Web forms [Seite 2285])
Process work items (Processing Work Items as a Web Form [Seite 2297])

Prerequisites
To use this technology, you must
Configure an ITS (Internet Transaction Server [Extern])
Install the SAP@Web Studio

The SAP@Web Studio is part of the ITS installation and runs on NT4.0 or higher

Features
Unlike electronic forms, Web forms are not integrated using the SAPforms interface.
Instead, an ITS service is developed from a Web transaction generated previously from a task in the R/3 System. This ITS service consists of HTML templates (among other things) that can then be revised using suitable tools.
Starting Workflows with Web Forms

Use
A Web form is completed by an employee or customer on the Internet and sent to the R/3 System. The task (only workflow templates or standard tasks are supported) associated with this form is then started in the R/3 System. The corresponding **import elements** of the container for this task are then supplied with input data from the form.

Features

When the Web form is processed, a Web-enabled R/3 transaction (“Web transaction”) is called from a Web page. The transaction is actually called via the Internet Transaction Server [Extern] (ITS). The ITS communicates with the R/3 System directly via the SAPGUI interface to support Web transactions.
Designing a Web Form to Start a Task

**Purpose**
You want to design a Web form to start a workflow or task in the R/3 System.

**Prerequisites**
A *namespace template* and a *development class* must be defined in Customizing. To do so, carry out the activity *Name range maintenance for task-related transactions* in Customizing for SAP Business Workflow.

**Process Flow**
You define the workflow or task that is to be started by a Web form. If elements in the task container are to be filled with values from the form, they must be identified as *import elements*

- Defined with a data type reference to an *ABAP dictionary field*.
- Multiple-line elements or elements that reference an object or structure are not supported at present.

You then generate a *Web transaction* in the system that allows users to fill the import elements of the task container and start the task.

- A Web transaction is an R/3 transaction that must be specially designed so that it can be called from an HTML page. The name of this transaction is generated and consists of the namespace template that you defined in Customizing and the task ID.

Following this, you develop an *ITS service* from this transaction in the SAP@Web Studio.

An ITS service essentially comprises an ITS service description, HTML templates (one for every screen in the R/3 transaction) and language-support files. The Studio is a development tool linked to the R/3 System that runs on your PC. The Studio assumes that the ABAP transaction already exists.
Defining a Task to be Started with a Web Form

Prerequisites
You are familiar with the procedure for defining tasks.
You want to define the task (single or multistep task) that is to be started by the Web form.

The part of the task definition that is relevant to forms essentially entails extending the container definition to include the elements that are to appear as fields on your form.

Procedure
Defining a Task
To create the task, choose Tools → Business Workflow → Development → Definition tools → Tasks/Task groups → Create.

Enter the type of task you want to create in the Task type field.

Only workflow templates (task type WS) and standard tasks (task type TS) can be used with Web forms.

Then choose Task → Create.

Define the basic data of your task.

Extending the Container Definition
Now extend the container of your task to include the elements that are to be filled via the form.
To launch the container editor, choose Goto → Container.

The container already contains the standard system elements. To add elements to the container, choose Edit → Create.

You can only transfer single values (not object references) from the form. If you want to add container elements for this purpose, therefore, you must always create them as single-line elements with a data type reference to a table field in the dictionary.

Create a suitable container element for all of the fields you want to fill in the form. Define these elements as import elements.

When generating the HTML template in the SAP@WebStudio the container element name is also used as the name of the input field. It is therefore recommended that you choose “meaningful” names (although the HTML templates can of course be processed at any time).

Note that when later changes are made to the name of container elements, the web transaction must be regenerated.
Further Procedure
Complete the task definition:
Declare the possible agents [Extern] that are allowed to start the task by filling out and submitting the form.

Save your task definition in the development class that you entered in the Customizing activity for maintaining the namespace.

If the task is a multistep task, maintain the associated workflow definition.
Generating a Web Transaction

Prerequisites
You have defined and saved the task that is to be started or executed with the Web form.

Procedure
To generate a Web transaction from the task:
Choose Goto → Web transaction in the task definition.

You cannot generate a Web transaction from a task,
if you are in Display mode in the task definition
if you have created a new task but have not saved it yet.

The following dialog box is then displayed: Web Transaction for Task...: Change.
Choose Generate.

The system then creates a transaction including a module pool and screens.

Save your module pool in the development class that you entered in the Customizing activity for
maintaining the namespace. You have also already saved the task in this development class.

Specify a change request.

The system activities (generating screens and module pool) are indicated by texts in the
status line.

Confirm the messages.

The dialog box appears again. The name of the new transaction now appears on the screen.
Click Continue to exit this dialog box.

Result
A Web transaction with a module pool and four screens has now been created in the system. The various
screens in this transaction are used to execute a single-step task and start a task.

If you change the task container after you have generated the Web transaction, you
must delete the transaction in the Web Transaction for Task...: Change screen and
regenerate it for the changes to take effect.

If you call the new transaction partly in the R/3 System, you will start a task.

Enter the name of the transaction in the OK code field in R/3.

Do not enter any data on the first screen. Choose Continue.

A screen is then displayed in which you can select the import parameters for the task
container.

Choose a few entries here as a test.
Choose Transfer.
The task is now started and a work item is generated. The number of this work item is displayed on the next screen.
Background Info on the Generated Web Transactions

Namespace
Specify a namespace template in the Name range maintenance for task-related form transactions activity in Customizing for SAP Business Workflow. The namespace template must be no more than 2 characters long.
The system then uses this information to create
the transaction code prefix Y<namespace template>
the module pool prefix SAPMY<namespace template>
and reserves the corresponding namespaces.

Transaction and Module Pool Names
The Web transaction used to start a task or process a work item is given the name Y<name space template><Task ID>H.
The name of the module pool for this transaction is SAPMY<namespace template><task ID>H.

<table>
<thead>
<tr>
<th>Namespace template</th>
<th>AB</th>
<th>Defined once in Customizing</th>
</tr>
</thead>
<tbody>
<tr>
<td>Task identifier</td>
<td>WS98700001</td>
<td>Assigned automatically by the system when the task is saved</td>
</tr>
<tr>
<td>Transaction</td>
<td>YABWS98700001H</td>
<td>Created with the Generate task function</td>
</tr>
<tr>
<td>Module pool</td>
<td>SAPMYABWS98700001H</td>
<td></td>
</tr>
<tr>
<td>Screens</td>
<td>50, 100, 150, 200</td>
<td></td>
</tr>
</tbody>
</table>

Module Pool Screens
The system creates four screens for the module pool. Matching HTML templates are then generated for these screens in the subsequent step Developing an ITS Service from a Transaction [Seite 2306].

Screen 050
This screen is filled with technical information and used internally.

Screen 100
This screen is only used if the transaction for executing a single-step task is executed. It contains the import elements of the container for the single-step task as display fields and the export elements of the task container as input fields.

Screen 150
This screen is only used if the transaction for starting a task is executed. It contains the import elements of the task container as input fields.

Screen 200
This screen contains final messages. If the procedure was successful, the number of the work item that was started is displayed here.
Developing an ITS Service from a Transaction

Prerequisites
You have created a Web transaction from a task, as described in the previous steps.

You know the task and the name of the transaction with the associated module pool. If you do not know the transaction and module pool, you can find out their names from the Repository Browser in the ABAP Workbench.

You have installed the SAP@WebStudio.

Procedure
The SAP@Web Studio contains separate documentation on using the Studio. For this reason, only information relevant to this scenario will be provided here.

Launch the SAP@Web Studio.

Create a project to store your ITS components.

Define a service for this project. Use the Service Wizard for this purpose.

- Enter the name of the transaction as the Service Name.
- Enter the name of the system in which the task is defined and other logon information.
- Select Use Diag-Channel on the dialog box and enter the name of the transaction.

Every ITS service based on a service description, in which, for example, the R/3 System and the transaction to be used are determined. Every service description is in its own service file.

For each ITS there is also a global service description that contains the standard settings for all ITS services. These settings can be modified and/or overwritten by the individual service descriptions.

Each service description is in its own service file (ASCII file ending in srvc) in the file system of the ITS computer. The HTMLBusiness-templates that are needed for the Web transaction (HTML files with specific placeholders for R/3 data) are also located there.

The SAP@Web Studio allows the user to create and maintain service descriptions and HTMLBusiness-templates.

Create the HTML templates for the Web transaction. Use the Template Wizard for this purpose.

- Enter the name of the module pool for the transaction in the Program field.
- Enter 50;100;150;200 in the Screen number field.
- Enter the name of the service created in Step 3 (name of the transaction) in the Service field.
- Enter the 99 as the theme.
One Theme is a logical name for the optical appearance of a web transaction. The name consists of two characters (letters or numbers). In principle, as many Themes as you like can be used for a web transaction. The Theme should always be set to 99 for Web forms.

The Studio creates an HTML template file on your PC for every screen in the transaction. The name of this file appears in the object list for the service. To display the HTML source code, double-click the name of the template. Revise the HTML templates. For more information, see Extending HTML templates [Seite 2310].

Define an ITS site for the system, if you have not already done so. Transfer your ITS files to the ITS site (publish function).

**Result**

All of the files required by this service have been copied to your ITS server and, therefore, can be called when necessary.
Background Info on the Generated HTML Templates

An HTML template is created for each of the four screens. The SAP@Web Studio allows you to edit these templates so that you can design your own Web forms to match your corporate design. The names of these HTML templates comprise the name of the module pool and the screen numbers:

\[ \text{SAPMY<namespace template><task identifier>H}_{<\text{screen number}}.\text{HTML} \]

The HTML templates are used as follows:

\[ \text{SAPMY<namespace template><task identifier>H}_{50}.\text{HTML} \]

When you call the service from the Internet, as described in the section Starting a Task with a Web Form [Seite 2296], the fields in this template are filled but the template itself is not displayed.

Since this HTML template is never displayed, it is not edited.

\[ \text{SAPMY<namespace template><task identifier>H}_{100}.\text{HTML} \]

This HTML template is the template for the form that processes a work item. As standard, the HTML template contains:

- display fields created by the import elements (that are not also export elements) of the task
- input fields created by the export elements of the task
- a button for completing processing and transferring the modified data to the task in the R/3 System.

\[ \text{SAPMY<namespace template><task identifier>H}_{150}.\text{HTML} \]

This HTML template is the template for the form that starts a workflow template or a standard task. As standard, the HTML template contains:

- input fields created by the import elements of the task
- a button for completing processing and transferring the data to the task in the R/3 System.

\[ \text{SAPMY<namespace template><task identifier>H}_{200}.\text{HTML} \]

This HTML template is the template for the form that acknowledges that the form has been successfully processed. As standard, the HTML template contains:

- three lines for message texts
- a button for acknowledging processing.

\[ \begin{array}{|c|c|}
\hline
\text{Service} & \text{YABWS987000001H.SRVC} \\
\hline
\text{HTML templates} & \text{SAPMYABWS98700001H}_50.\text{HTML} \\
& \text{SAPMYABWS98700001H}_100.\text{HTML} \\
& \text{SAPMYABWS98700001H}_150.\text{HTML} \\
& \text{SAPMYABWS98700001H}_200.\text{HTML} \\
\hline
\end{array} \]

Service YABWS987000001H.SRVC

Must be created with the transaction name

\[ \begin{array}{|c|}
\hline
\text{HTML templates} & \text{SAPMYABWS98700001H}_50.\text{HTML} \\
& \text{SAPMYABWS98700001H}_100.\text{HTML} \\
& \text{SAPMYABWS98700001H}_150.\text{HTML} \\
& \text{SAPMYABWS98700001H}_200.\text{HTML} \\
\hline
\end{array} \]
Starting a Task with a Web Form

Prerequisites
You have developed an ITS service in the SAP@Web Studio with the four HTML templates for starting tasks or processing work items, as described in the steps above. You have also edited the templates (if necessary) to design your own individual Web form.

Procedure
To start the task with the form:
Open your Internet browser.
Call the ITS service. The URL must be structured as follows:

\[ 
\text{http://<ITS-Server>/scripts/wgate/<webtransaction>/?WEB\_FLAG=X&~OkCode=CONT} 
\]

\(<\text{ITS Server}>\) is the name of your Internet Transaction Server.
\(<\text{webtransaction}>\) is replaced with the name of your web transaction.
\(\text{WEB\_FLAG=X}\) is the transaction that is executed using a Web form.
\(\text{~OkCode=CONT}\) - the initial screen (screen 50) is skipped
The form then appears with the fields for entering or displaying data.

Edit the form.
Transfer the form to the R/3 System using the function provided (usually by clicking a pushbutton).
A message is then displayed informing you of the work item number with which the task was started.
Processing Work Items as a Web Form

Use

Employees can also display their Business Workplace, with all the work items they have received, on the Internet (SAP Business Workplace Internet application component). While all of the work items displayed can also be executed in the R/3 System, only work items for tasks that fulfill certain technical conditions can be started over the Internet.

Work items for Web form tasks [Seite 2298] can always be executed on the Internet.

Features

If the work item representing the Web form task is listed in the Web-Workplace, the corresponding Web form can be processed. When the form is transferred to the R/3 System, the work item container of the relevant work item is updated. The work item is then terminated.

If the same work item would be listed in the Business Workplace in the R/3 System, it would be processed with the Web transaction that forms the basis of the Web form.
Web form tasks

Definition

Single-step task that executes the `HTMLProcess` method of the `FORM object type` [Seite 74]. The `Executable on the Internet` indicator must be selected in the properties of this task.

Use

When a Web form task is executed as a step in a workflow, a work item is displayed as standard in both the `Business Workplace` in the R/3 System and in the `Web-Workplace` of the recipients for this step. This work item can be executed from the Web-Workplace.

In this case, the work item container is processed with a Web form in the Internet browser.
Designing a Web Form to Process a Work Item

Purpose
You want to design a Web form to process a work item.

Prerequisites
A namespace template and a development class must be defined in Customizing. To do so, carry out the activity Name range maintenance for task-related transactions in Customizing for SAP Business Workflow.

Process Flow
You define a standard task that is to be executed as a Web form. This task must reference the HTMLProcess method of the FORM object type. The elements in the task container that are to appear and be processed on the form must be identified as export elements

- Defined with a data type reference to a DDIC field.
- Multiple-line elements or elements that reference an object or structure are not supported at present.
- The elements in the task container that are only to be displayed on the form must be identified as import elements

- Defined with a data type reference to a DDIC field.
- Multiple-line elements or elements that reference an object or structure are not supported at present.

You use this standard task in an activity in a workflow definition. Here, you must define the binding between the elements in the task container and the corresponding elements in the workflow container.

You can then generate a Web transaction from the task in order to execute the task, that is, that processes the export elements of the task container or displays the import elements.

A Web transaction is an R/3 transaction that must be specially designed so that it can be called from an HTML page. The name of this transaction is generated and consists of the namespace template that you defined in Customizing and the task ID.

Following this, you develop an ITS service from this transaction in the SAP@WebStudio.

An ITS service essentially comprises an ITS service description, HTML templates (one for every screen in the R/3 transaction) and language-support files.
The Studio is a development tool linked to the R/3 System that runs on your PC. The Studio assumes that the ABAP transaction already exists and that you want to create (or change) the ITS service components.
Defining a Task to be Executed as a Web Form

Prerequisites
You have defined a workflow and now want to add a single-step task that can be processed with a Web form.

The part of the task definition that is relevant to forms essentially entails extending the container definition to include the elements that are to be processed as fields on your form.

Using the FORM.HTML Process object method.

Procedure
Defining the Form Task
To create the tasks, choose Tools → Business Workflow → Development → Definition tools → Tasks/Task groups → Create.

Enter the type TS for standard task in the Task type field.

Then choose Task → Create.

Define the basic data of your task.

Enter FORM as the object type and HTMLProcess as the object method.

Select the Executable on the Internet field in the task properties.

Extending the Container Definition
You can add other container elements to a container as form-specific extensions. The contents of these elements can then be displayed and edited in the fields on the form. All of the data that is to be displayed to the form user must exist as container elements in the task container.

To launch the container editor, choose Goto → Container.

The container already contains the standard system elements. To add elements to the container, choose Edit → Create.

You can only transfer single values (not object references) from the form. If you want to add container elements for this purpose, therefore, you must always create them with a data type reference to a table field in the dictionary.

For each field

that is to be displayed on the form, create a corresponding container element and identify it as an import element

that is to be entered on the form, create a corresponding container element and identify it as an export element

that is to be displayed and overwritten on the form, create a corresponding container element and identify it as an import and export element
When generating the HTML template in the SAP@WebStudio the container element *name* is also used as the name of the display/input field.

It is therefore recommended that you choose "meaningful" *names* (although the HTML templates can of course be processed at any time).

Note that when later changes are made to the name of container elements, the web *transaction* must be regenerated.

**Further Procedure**

Complete the task definition:

Declare the *possible agents [Extern]* that are allowed to start the task by filling out and submitting the form.

Save your task definition in the development class that you entered in the Customizing activity for *maintaining the namespace*. 
Generating a Web Transaction

Prerequisites
You have defined and saved the task that is to be started or executed with the Web form.

Procedure
To generate a Web transaction from the task:
Choose Goto → Web transaction in the task definition.

You cannot generate a Web transaction from a task,
if you are in Display mode in the task definition
if you have created a new task but have not saved it yet.

The following dialog box is then displayed: Web Transaction for Task...: Change.
Choose Generate.

The system then creates a transaction including a module pool and screens.

Save your module pool in the development class that you entered in the Customizing activity for
maintaining the namespace. You have also already saved the task in this development class.

Specify a change request.

The system activities (generating screens and module pool) are indicated by texts in the
status line.

Confirm the messages.

The dialog box appears again. The name of the new transaction now appears on the screen.
Click Continue to exit this dialog box.

Result
A Web transaction with a module pool and four screens has now been created in the system. The various
screens in this transaction are used to execute a single-step task and start a task.

If you change the task container after you have generated the Web transaction, you
must delete the transaction in the Web Transaction for Task...: Change screen and
regenerate it for the changes to take effect.

If you call the new transaction partly in the R/3 System, you will start a task.

Enter the name of the transaction in the OK code field in R/3.
Do not enter any data on the first screen. Choose Continue.

A screen is then displayed in which you can select the import parameters for the task
container.
Choose a few entries here as a test.
Choose Transfer.
The task is now started and a work item is generated. The number of this work item is displayed on the next screen.
Background Info on the Generated Web Transactions

Namespace
Specify a namespace template in the Name range maintenance for task-related form transactions activity in Customizing for SAP Business Workflow. The namespace template must be no more than 2 characters long.
The system then uses this information to create
the transaction code prefix Y<namespace template>
the module pool prefix SAPMY<namespace template>
and reserves the corresponding namespaces.

Transaction and Module Pool Names
The Web transaction used to start a task or process a work item is given the name Y<namespace template><Task ID>.
The name of the module pool for this transaction is SAPMY<namespace template><task ID>.

<table>
<thead>
<tr>
<th>Namespace template</th>
<th>AB</th>
<th>Defined once in Customizing</th>
</tr>
</thead>
<tbody>
<tr>
<td>Task identifier</td>
<td>WS98700001</td>
<td>Assigned automatically by the system when the task is saved</td>
</tr>
<tr>
<td>Transaction</td>
<td>YABWS98700001H</td>
<td>Created with the Generate task function</td>
</tr>
<tr>
<td>Module pool</td>
<td>SAPMYABWS98700001H</td>
<td></td>
</tr>
<tr>
<td>Screens</td>
<td>50, 100, 150, 200</td>
<td></td>
</tr>
</tbody>
</table>

Module Pool Screens
The system creates four screens for the module pool. Matching HTML templates are then generated for these screens in the subsequent step Developing an ITS Service from a Transaction [Seite 2306].

Screen 050
This screen is filled with technical information and used internally.

Screen 100
This screen is only used if the transaction for executing a single-step task is executed. It contains the import elements of the container for the single-step task as display fields and the export elements of the task container as input fields.

Screen 150
This screen is only used if the transaction for starting a task is executed. It contains the import elements of the task container as input fields.

Screen 200
This screen contains final messages. If the procedure was successful, the number of the work item that was started is displayed here.
Developing an ITS Service from a Transaction

Prerequisites
You have created a Web transaction from a task, as described in the previous steps.
You know the task and the name of the transaction with the associated module pool. If you do not know the transaction and module pool, you can find out their names from the Repository Browser in the ABAP Workbench.
You have installed the SAP@WebStudio.

Procedure
The SAP@Web Studio contains separate documentation on using the Studio. For this reason, only information relevant to this scenario will be provided here.
Launch the SAP@Web Studio.
Create a project to store your ITS components.
Define a service for this project. Use the Service Wizard for this purpose.
   Enter the name of the transaction as the Service Name.
   Enter the name of the system in which the task is defined and other logon information.
   Select Use Diag-Channel on the dialog box and enter the name of the transaction.

Every ITS service based on a service description, in which, for example, the R/3 System and the transaction to be used are determined. Every service description is in its own service file.

For each ITS there is also a global service description that contains the standard settings for all ITS services. These settings can be modified and/or overwritten by the individual service descriptions.

Each service description is in its own service file (ASCII file ending in srvc) in the file system of the ITS computer. The HTML templates that are needed for the Web transaction (HTML files with specific placeholders for R/3 data) are also located there.

The SAP@Web Studio allows the user to create and maintain service descriptions and HTML templates.
Create the HTML templates for the Web transaction. Use the Template Wizard for this purpose.
   Enter the name of the module pool for the transaction in the Program field.
   Enter 50;100;150;200 in the Screen number field.
   Enter the name of the service created in Step 3 (name of the transaction) in the Service field.
   Enter the 99 as the theme.
Developing an ITS Service from a Transaction

One Theme is a logical name for the optical appearance of a web transaction. The name consists of two characters (letters or numbers). In principle, as many Themes as you like can be used for a web transaction. The Theme should always be set to 99 for Web forms.

The Studio creates an HTML template file on your PC for every screen in the transaction. The name of this file appears in the object list for the service. To display the HTML source code, double-click the name of the template.

Revise the HTML templates. For more information, see Extending HTML templates [Seite 2310].

Define an ITS site for the system, if you have not already done so.

Transfer your ITS files to the ITS site (publish function).

Result

All of the files required by this service have been copied to your ITS server and, therefore, can be called when necessary.
Background Info on the Generated HTML Templates

An HTML template is created for each of the four screens. The SAP@Web Studio allows you to edit these templates so that you can design your own Web forms to match your corporate design. The names of these HTML templates comprise the name of the module pool and the screen numbers:

\[
\text{SAPMY namespace template\<task identifier\>H\_<screen number>.HTML}
\]

The HTML templates are used as follows:

\text{SAPMY namespace template\<task identifier\>H_50.HTML}

When you call the service from the Internet, as described in the section Starting a Task with a Web Form [Seite 2296], the fields in this template are filled but the template itself is not displayed.

Since this HTML template is never displayed, it is not edited.

\text{SAPMY namespace template\<task identifier\>H_100.HTML}

This HTML template is the template for the form that processes a work item. As standard, the HTML template contains:

- display fields created by the import elements (that are not also export elements) of the task
- input fields created by the export elements of the task
- a button for completing processing and transferring the modified data to the task in the R/3 System.

\text{SAPMY namespace template\<task identifier\>H_150.HTML}

This HTML template is the template for the form that starts a workflow template or a standard task. As standard, the HTML template contains:

- input fields created by the import elements of the task
- a button for completing processing and transferring the data to the task in the R/3 System.

\text{SAPMY namespace template\<task identifier\>H_200.HTML}

This HTML template is the template for the form that acknowledges that the form has been successfully processed. As standard, the HTML template contains:

- three lines for message texts
- a button for acknowledging processing.

<table>
<thead>
<tr>
<th>Service</th>
<th>YABWS98700001H.SRVC</th>
<th>Must be created with the transaction name</th>
</tr>
</thead>
<tbody>
<tr>
<td>HTML templates</td>
<td>SAPMYABWS98700001H_50.HTML</td>
<td></td>
</tr>
<tr>
<td></td>
<td>SAPMYABWS98700001H_100.HTML</td>
<td></td>
</tr>
<tr>
<td></td>
<td>SAPMYABWS98700001H_150.HTML</td>
<td></td>
</tr>
<tr>
<td></td>
<td>SAPMYABWS98700001H_200.HTML</td>
<td></td>
</tr>
</tbody>
</table>
Extending HTML Templates

**HTML Business**

The [Internet Transaction Server](Ext) (ITS) requires an HTML Business-template for each R/3 transaction screen. An HTML Business-template is an HTML page in which HTML Business-commands are embedded. The ITS generates a page of pure HTML for each template.

**HTML Business**

Is *not* a programming language

Is an extension of HTML for inserting data from the R/3 System into HTML pages

Supports the transfer of data from web forms to the R/3 System

Does not require client extension

**Properties**

HTML Business-commands are bracketed with ` (backtick) when mixed in with "normal" HTML.

There is no difference between upper and lower case in an HTML Business-expression.

There must be at least one character between two consecutive HTML Business-commands enclosed by a ` (backtick).

The SAP@Web Studio shows the HTML Business-orders in blue.

All names that occur in an HTML Business-expression are interpreted as names of an R/3 screen field.

If an HTML Business-expression consists of only one field name, this expression is replaced at runtime with the value of the R/3 screen field *with the same name*.

If a field with the addition `.LABEL` is used in an HTML Business-expression, this expression is replaced in the R/3 screen with the field indicator of the same name.

As well as `.LABEL`, the following additions also exist, which can be used with every screen field that is *ready for input*.

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>MAXSIZE</td>
<td>Maximum entry length of a screen field</td>
</tr>
<tr>
<td>VISSIZE</td>
<td>Maximum number of characters visible in a screen field</td>
</tr>
</tbody>
</table>

**Programming Function Keys**

If you want to design your form in such a way that inputs are not made directly in a field but with buttons, you must edit the HTML template as shown in the following example:

The standard task for approving an application contains the *approval status* element in its container. This element is marked as an import and export element and is defined with a data type reference to a *Char1* field (for example `SYST-INPUT`).

The HTML template for screen 100, therefore, contains the *Approval status* input field and the *Transfer* button as standard.

The functionality you want to provide on the form will allow users to set the approval status to *approved* or *rejected* using one of two buttons and therefore end processing.

The *Approval status* input field and the *Transfer* pushbutton are to be removed.

The lines that generate this input field and button, therefore, must be deleted or commented out in the HTML template source code.

```
approval status010100.label`
<INPUT TYPE="text" name="approval status010100" VALUE="approval status010100" maxlength="1" size="1" />
<INPUT TYPE="submit" name="OkCode(STRT)" value="START.label">
```
The following lines are then added to the coding:

```html
<form ACTION="`wgateURL()`" METHOD="post">
  <INPUT TYPE="submit" name="~OKCode(STRT),approval status010100=A" value="approve">
  <INPUT TYPE="submit" name="~OKCode(STRT),approval status010100=R" value="approve">
</form>
```

This code creates the two buttons and labels them Accept and Reject. The value of the approval status container element is then set to A or R, depending on the button selected, and returned to the R/3 System. If the button text is to depend on the logon language, you must use language resources.

**Integration of Possible Entries**

By inserting a command it is now also possible in the web to use an input help for particular field types.

```html
<input type="text" size="10" maxlength="10"
  name= "`LASTDAY1_TEST015004.name`1"
  value= "`LASTDAY1_TEST015004.value`">`assert(LASTDAY1_TEST015004.name)`
  <INPUT TYPE="SUBMIT" NAME="~searchhelp(`LASTDAY1_TEST015004.name`)" VALUE="?">
```

**Simplification of Process Flow for Work Item Execution**

You can accelerate and simplify the process flow of execution by modifying the HTML template for the execution of a work item in the web described below. The Web Workplace is displayed directly, for example, if you cancel an activity.

```html
Template 50:
Instead of the old <form..., after <body>:
<form action="`wgateURL()`">

Template 100:
After </title>, before </ head>:
<script language="JavaScript">
  function logoff()
  {
    hiddenConfirm = new Image();
    hiddenConfirm.src = "`wgateURL(~OKCode="/NEX")`";
  }
  function return_bwsp()
  {
    parent.location.href="~ret_url`BW02_1400-CHOICE=BAIN&~okcode=WORK&~confirm_url=`~confirm_url`&~dec_state=1`";
  }
</script>
At the start, directly after <body ...>: Replace <form method="post" action="`wgateURL()`">
with:
<form NAME="theform" METHOD=POST
  ACTION="`wgateURL(~OKCode="WORK")`&`~dec_state=0&~ret_url=`~ret_url`&`confirm_url= wgateURL()`">
  Before the cancel button: Instead of <input type=submit name="~okcode="/NEX
  value="`CANCEL.label`">
</form>
```
<FORM onsubmit="logoff()" METHOD=POST ACTION="`~ret_url`&BW02_1400-CHOICE=BAIN&OKCode=WORK" target="_parent">
<input type=submit name="" value="`CANCEL.label`">

Template 200:
Before</head>:
<script language="JavaScript">
function logoff() {
  hiddenConfirm = new Image();
  hiddenConfirm.src = "`wgateURL(~OKCode="/NEX")`";
}

function return_bwsp() {
  parent.location.href="`~ret_url`&BW02_1400-CHOICE=BAIN&okcode=WORK&confirm_url=`~confirm_url`&~dec_state=1";
}
</script>
In the <form></form>-blocks delete the following line:
<input type=submit name="okcode="/NEX" value="`SUBMIT.label`">
After</head>:
`if (~dec_state == "0")`
<body onload="return_bwsp()" BGCOLOR="#ffffff" LINK=BLUE VLINK=BLUE>
`elseif (~dec_state == "1")`
<body onload="logoff()" BGCOLOR="#ffffff" LINK=BLUE VLINK=BLUE>
Before</html>:
`end`
Adding tasks to a workflow definition

Prerequisites
You have defined a Web form task [Seite 2298], generated a Web transaction for it, and developed an ITS service in the SAP@WebStudio with the four HTML templates, as described in the steps above. You have also edited the templates (if necessary) to design your own individual Web form.

Procedure
The workflow definition for executing your form application is processed as follows:

Add a step that represents an activity in which your Web form task is referenced.

Define the binding between the workflow container and the import elements in the task container (Remember: These are the elements that are displayed on the form.)

Define the binding between the export elements in the task container and the workflow container. (Remember: These are the elements that can be entered in the form.)

Delete all of the binding definitions that may have been created automatically and that refer to the element _WI_Object_ID.

If data was changed in the form application and copied to the workflow container, you must define the following steps (condition, multiple condition) to evaluate this information.
Processing a Work Item on the Internet

Prerequisites
The workflow system has created a work item that represents the Web form task [Seite 2298]. This work item is displayed in the Business Workplace of its recipients.

Procedure
To process this work item as a Web form on the Internet:
Open your Internet browser.
Call the ITS service BWSP (Web-Workplace).

You can then process the work item from your Web-Workplace.

Your Business Workplace on the Internet is only a “mirror” of your Business Workplace in the R/3 System. In other words, you could also execute the work item in the R/3 System.

In this case, it is not processed with a Web form but with the Web transaction that was generated from the task.
BC - SAP Business Workflow - Tutorials

These tutorials introduce you to SAP Business Workflow, its fundamental principles, its operation and some of its functions.

Introduction

If you have little or no experience with SAP Business Workflow, work through the following two tutorials first:

Tutorial: Workflow Modeling [Seite 1583]
Tutorial: Maintaining the Organizational Plan [Seite 1639]

We also recommend that you work through these tutorials as preparation for taking part in an SAP training course.

Events

If you are particularly interested in options for event creation, refer to the following tutorials:

Tutorial: Event Creation Upon Status Changes and Creation of "Mails" [Seite 1647]

Object types

Object type definitions are covered in a separate tutorial:

Tutorial: Workflow Programming [Seite 1667]

Programming knowledge is a prerequisite for this tutorial. It is therefore only suitable for advanced users.

Input conventions

At several points in the tutorials, you are requested to enter names, descriptions, or texts. These inputs are shown in bold letters. You can replace entries in pointed parentheses (<example>) with your own texts.

You can choose your own IDs and descriptions. All of the abbreviations and IDs you use in this tutorial should start with the same character string, for example, your initials. This will help you to find your own definitions. The points at which an input is required are indicated by the character string ini.

Options and outputs form the R/3 Systems are indicated in a different typeface.

Prerequisites

To work through this tutorial successfully,

You should already be familiar with the basic terms in SAP Business Workflow.

You do not have to have any practical experience with SAP Business Workflow.

You must be able to carry out all of the individual steps in the system directly.

Client-independent objects

You will be shown how to create client-independent [Extern] objects. You will only be able to work through this tutorial on the system if you are authorized to create these objects.

Customizing

All of the Customizing settings must be defined for SAP Business Workflow. To check these settings, choose Tools → Business Workflow → Development → Tools → Customizing. If the system detects that some of the Customizing settings have not been defined, carry out the automatic Customizing procedure.
For further information, please refer to Maintaining the Standard Settings for SAP Business Workflow [Extern].

You do not need to check the settings if you are already using SAP Business Workflow. You can use the automatic Customizing function more than once, since existing settings are not overwritten.
Tutorial: Workflow Modeling

Purpose
This tutorial uses an example in a series of easy-to-follow units to explain the most important tools in SAP Business Workflow. The example used here is based on a scenario for approving a notification of absence. At the end of this tutorial, you will have defined and executed a workflow that automatically submits a notification of absence (leave request) to your superior for approval, and informs the requester of the result of the approval process. You will become familiar with the following areas of SAP Business Workflow throughout the course of this tutorial:

- Definition tools
- Business Workplace
- Reporting and analysis tools
- Using the Workflow Builder

The tutorial is not intended to provide a full description of all functions and concepts. This information is available in the documentation on SAP Business Workflow [Extern]. This tutorial does not deal with the definition of object types. If you want further information on this subject, please work through the tutorial on Workflow Programming [Seite 1667].

Process Flow
Work through the individual units in this tutorial in the specified order. Important units are followed by tests that you can use to test what you have learned to date. Please make sure to complete these tests.

Result
Example - the notification of absence
The scenario in this example begins with the completion of a leave request by an employee (requester or creator of the notification of absence). The completed form is then forwarded automatically to the head of department (employee’s superior).

- If the head of department approves the request, the employee receives a notification and the workflow is terminated.

- If the head of department rejects the request, the employee can decide to revise the request (possibly in accordance with the head of department’s wishes) or withdraw it. If the employee decides to revise it, the request form is resubmitted to the head of department after the revision is made.
Business Process and Workflow: Example

The diagram above shows that additional steps could follow the approval, such as updating the leave account, or notifying the secretary. These steps, however, do not arise in this example.

All of the units at a glance

The diagram below shows all of the units in this tutorial. Similar units are listed in the same column.
Unit 1: Organizational Structure

Use
In order for the workflow system to establish the relationship between the requester and their superior, you must create an organizational plan in the system. For this tutorial, of course, this organizational plan does not have to be complete and valid across the enterprise. To keep the test as simple as possible initially, define an organizational plan that only contains one administrator and one head of department. Assign both items to yourself. As a result, all work items will appear in your own Business Workplace. Later you will modify the organizational plan such that you will have to work through the scenario with two users.

Procedure
The organizational plan required for this tutorial consists, when complete, of one organizational unit (= “department”), which contains two positions: a head of department and an administrator. Each position is described by one job and each position is assigned one user as holder. The head of department position is also designated as chief position of the organizational unit.

Of course, a “real” organizational plan is created by arranging several organizational units with their positions in a hierarchy. Usually several positions are described by one job.

For further information, refer to the documentation Organizational Plan.

The procedure in this unit is divided into four parts:
4. You create an organizational unit.
5. Enter necessary jobs as required.
6. You create a position for the head of department in the new organizational unit.
   You define a position in three steps:
   i. You create a position that is assigned your organizational unit.
   ii. You assign a holder to the position.
   iii. You assign a job to the position.
4. You create a position for the administrator in the organizational unit.

Creating an organizational unit
10. Confirm the validity period proposed in the dialog box Creating a Root Organizational Unit. This takes you to the Create Organization and Staffing (Workflow) screen. This user interface is divided into four screen areas:
11. On the Basic Data tab in the details area, enter an abbreviation and a name in the Organizational unit input fields.

   Abbreviation:  <ini_sales>
   Name:         <OrgUnit: Sales (ini)>

12. Choose \[button\].
   You can now create the position for a head of department and one administrator.

**Create jobs**

When enhancing an organizational unit, the necessary jobs are usually already available. For this tutorial however, you create the necessary jobs for head of department and administrator yourself.

6. Choose \[button\] \[Action\] \[Create jobs\].
   You go to the dialog box Create jobs. The lower area contains a list of existing jobs and the upper area contains an input table in which you can create new jobs by entering abbreviations and names.

7. In the input table, enter an abbreviation and a name for each of the new jobs.
   Job - head of department:
   Object abbreviation: <ini_dhead_C>
   Name: <job: head of department (ini)>
   Job administrator:
   Object abbreviation: <ini_admi_C>
   Name: <job: administrator (ini)>
8. Choose 🖍.

**Creating Position for Head of Department**

11. Change to the overview area in the staff assignments [Extern] of the organizational unit, in order to assign positions, jobs and holders. Choose the arrow 🖍 on the right next to the 🖍 and then the staff assignments (list).

12. Choose 🖍.

A new position is then created in the staff assignments and is displayed in a new line in the table. The position is vacant and no job is assigned to it.

13. Open the details view for the new position in the details area by double-clicking on the entry in the table.

14. On the Basic Data tab, enter a code and a description in the Position input fields. Overwrite the previous contents.

   Abbreviation: <ini_dhead_S>
   Description: <position: head of department (ini)>

**Assigning a holder to the position**

You now assign R/3 users to the positions. The staff assignments for your organizational unit are displayed and you see the vacant position in the table.

15. Select User in the search area and enter the search criteria in order to find your user names.

   All of the user names that match your search criterion are listed in the selection screen.

16. Select your user name in the selection area and drag it to the Person/User column of the position in the overview area.

   Confirm the message that the relationship period of the validity has been changed.

17. Set the Head of own organizational unit indicator in the details area.

**Assigning a job to the position**

Assign the job of the head of department you created earlier to the position.

18. Select Job in the search area and enter the search criteria in order to find the job of the head of department.

   All jobs that match your search criterion are listed in the selection screen.

19. Select job: head of department (ini) in the selection area and drag it to the Job column of the position in the overview area.

20. Choose 🖍.

   The job is assigned the position. Check this by switching to the staff assignments of the organizational unit. Select the organizational unit in the overview area, choose the arrow 🖍 on the right next to the 🖍 and then the staff assignments (list). The newly created job is displayed in the job column.

**Create position for administrator and assign holder and job**

You are now in the staff assignments of the organizational unit.

9. Choose 🖍.
A new position is then created in the staff assignments and is displayed in a new line in the table. The position is vacant and no job is assigned to it.

10. On the Basic Data tab in the details area, enter an abbreviation and a name in the Position input fields. Overwrite the previous contents.

   ID: <ini_admi_S>

   Description: <position: administrator (ini)>

11. Choose.

12. Select User in the search area and enter the search criteria in order to find your user names.

13. Select your user name in the selection area and drag it to the Person/User column of the position in the overview area.

   Confirm the message that the relationship period of the validity has been changed.

14. Select Job in the search area and enter the search criteria in order to find the job of the administrator.

15. Select job: administrator (ini) in the selection area and drag it to the Job column of the position in the overview area.


**Result**

Display your entire staffing schedule again and make sure that all the information listed is correct. You can display a detailed view of jobs, users, and positions. Choose the relevant cell in the table by double-clicking it.

In the details view of a position or job, all of the assigned tasks are displayed on the Tasks tab.

You have now completed the first unit (defining the organizational plan). You can now start on the next unit. To exit processing of the organizational structure, choose **Back**.
Unit 2: Creating a Workflow

Use
To define the flow of the approval process in the system, you first create a multistep task workflow template.
For further information, refer to Definition of Multistep Tasks [Seite 1194] in the SAP Business Workflow documentation.

Procedure: Creating a workflow template
   This takes you to the Task: Maintain screen.
5. In the Task type field, choose the Workflow template entry.
6. Choose.
   The Workflow Template: Add screen is then displayed.

Entering the basic data for the workflow template
13. Enter an abbreviation (of your choice) and a name (of your choice) for the workflow template to be created.

   Abbreviation: <ini_ws>

   Name: <Workflow: notification of absence (ini)>
15. In the Create Object Directory Entry dialog box, choose Local object.
   The system has now allocated an 8-digit number for your workflow template. This number is made up of the 3-digit prefix number (Customizing setting) and a 5-digit number taken from a number range. This number together with the letters WS forms the workflow ID. Make a note of the ID so that you can later call the workflow more quickly.

Entering a workflow description
16. Go to the Description tab.
17. Choose the text type Task description and then.
   The SAPscript text editor is then launched.
18. Enter a text that describes the task. You can use the following proposal:

   <You can use this workflow to create a notification of absence. This will then be sent to your superior for review. You will be notified of the result of this review. If your request is rejected, you can revise it and resubmit it for review or withdraw it completely.>
19. Choose to return to the workflow template and then save the workflow template.
**Determining agents**

To enable the workflow template to be started in dialog, the workflow template must be assigned to its possible agents.

20. Select **Additional data → Agent assignment → Maintain**

   The workflow template: agent assignment screen is displayed.

21. Position the cursor on the name of your workflow template.

22. Choose **Properties…**

   The dialog box Task: *<Workflow: Notification of absence (ini)>* is displayed.

23. Set the **General task** flag.

24. Choose ![Copy](https://example.com) and then ![](https://example.com).

   Designating your workflow template as a general task [Extern] means that every user in the system is authorized to start this workflow template in dialog.

**Result**

You have now created a workflow template as a "framework" for a workflow definition.

To create a workflow definition, you need tasks that are used as steps in the workflow definition.

**Additional information: Tasks**

A task can either be a single-step task [Extern] or a multistep task [Extern]. In our example of the notification of absence, making the application and checking the application are single-step tasks. The entire procedure involved in processing a leave request consists of these (and other) single steps, and is therefore a multistep task.

In this tutorial, a single step task is generally described as a task. The difference between the task types is only explicitly explained if there is a risk of confusion over multistep tasks. The term workflow is used as the umbrella term for the workflow template (and thus the multistep task) and the workflow definition. More precise differentiation only takes place if there is a risk of confusion.

On a technical level, the main difference between single-step tasks and multistep tasks is that:

- Every single-step task refers to an object method
- Every multistep task refers to a workflow definition

A workflow definition is made up of a sequence of different steps (for example, all steps required to process a leave request).

**Standard tasks** and workflow templates are client-independent. Customer and workflow tasks are client-dependent. Customer and workflow tasks are no longer supported and are therefore no longer used in this tutorial.

An Activity is a possible step type in a workflow definition. An activity refers to a task. It references a task, which can be a single-step or multistep task (subworkflow).

For further information, refer to Tasks and Task Groups [Seite 1172] in the SAP Business Workflow documentation.
Unit 4: Defining and Inserting Task "Create Notification of Absence"

Use

The notification of absence is created in the first step of the workflow. You will also need a task that executes this function. You can create the task separately from the workflow definition or directly from the Workflow Builder. This unit describes how you create the task from the Workflow Builder. For further information on creating an individual task, refer to Definition of Single-Step Tasks [Seite 1175] in the SAP Business Workflow documentation.

Prerequisites

This task is a single-step task and it references one object method. The object type used (FORMABSENC (notification of absence)) and the required method (Create) are already defined and implemented in the Business Object Repository.

You are in the Workflow Builder and the initial workflow definition is displayed.

Procedure

Creating an Activity as a Step in the Workflow Definition

Add a first step of the type Activity to the workflow definition.

3. Choose under step type. The cursor changes shape.
4. Position the cursor on the Undefined step and click.
   By doing so, you add an activity to your workflow definition. The step definition is shown in the right-hand part of the screen. You are on the tab page Control.

Defining a Task "Create Notification of Absence"

You now create a new task, with which a notification of absence can be created.
6. Choose the arrow on the right next to the and then Create task.
   The Basic data tab in the Standard task: Create screen is then displayed.

Entering the basic data

7. Enter an abbreviation (of your choice) and a name (of your choice) for the standard task to be created.
   Abbreviation: <ini_creat>
   Name: <Create notification of absence (ini)>

Work item text

The work item text appears as an information text in the Business Workplace as soon as there is a work item for the step with this task.
8. Enter Create notification of absence in the Work item text field.

Object type and method

9. Enter an object type and a method:
   Object type: FORMABSENC
Method: **CREATE**

You can also use the F4 input help to select the method. Additional indicators associated with this method are set automatically.

You can view the definition of the object type entered by double-clicking on it. Information about its components is provided.

10. Save your task as a local object.

The system has now allocated an 8-digit number for your task. This number is made up of the 3-digit prefix number (Customizing setting) and a 5-digit number taken from a number range. The identification of this task consists of **TS** and the eight digit number.

**Determining agents**

Selecting the **possible agents [Extern]** for this task determines who is to be authorized to make leave requests. For this task, this should be all employees in your enterprise. Define the task as a **general task [Extern]**.

7. Select **Additional data → Agent assignment → Maintain**

This takes you to the **Standard task: Maintain Agent Assignment** screen.

8. Position the cursor on the name of your task.

9. Choose **Properties…**

The dialog box **Task: <Notification of absence (ini)>** is displayed.

10. Set the **General task** indicator.

11. Choose ![Copy](copy.png) and then ![Paste](paste.png).

12. Exit the task definition with ![Close](close.png).

**Interim result**

You have now defined your first task in full. This task is used in the activity that you created in the Workflow Builder.

**Completing Activity**

The **Define Container Elements and Binding** dialog box is then displayed in which the system proposes a change to the workflow container and a binding definition.

- **Upper part of the dialog box: Proposal for new container elements in the workflow container.**

  The system proposes that you create a local container element in the workflow container, which can store the reference to the **notification of absence** created in this step. The element has the technical name **AbsenceForm**.

- **Lower part of dialog box: Proposed binding definition**

  The system proposes that you define a binding from the task container to the workflow container. This binding transports the reference to the generated object (i.e. the created notification of absence) **from the task** (from the **_WI_Object_ID** element in the task container) **to the workflow** (to the **AbsenceForm** element in the workflow container).
Further information on automatic binding definition proposals is available under Binding Definitions from the Workflow Container [Seite 1223] and Binding Definitions from the Task Container [Seite 1225] in the documentation on SAP Business Workflow.

9. Confirm the proposal with ✅.

The new AbsenceForm container element is displayed in the workflow container. Local container elements are marked 🟢. The remaining elements in the workflow container are workflow system elements. For further information, please refer to Workflow System Elements in the Workflow Container [Extern].

10. Enter a text to describe the step.

You can, of course, leave the description of the task that is proposed by the system.

(The current step in your workflow definition appears here.)

11. Go to the Outcomes tab.

All of the outcomes for this step are displayed here.

12. Enter a text for the description.

The description (proposal: <Application filled out>) in the Task executed synchronously line.

Each step has at least one subsequent event (in this case: Task executed synchronously). The description is the text with which the outcome is labeled in the workflow definition.

The Form does not exist and Form could not be generated outcomes are the possible exceptions that could occur in the method execution. You “run the risk” of not intercepting these exceptions and not modeling any exception handling in your workflow definition. If one of the exceptions actually occurs at runtime, the workflow will assume the error status.

Checking the binding definition (optionahttp://vuassociates.com/~pvu/l)

5. Go to the Control tab.

6. Choose Binding (present).

The WF Builder: Binding for Step ‘<Create notification of absence>’ dialog box is then displayed.

- In the upper part of the screen, the binding defined from the workflow container to the task container is displayed. The system executes this binding before providing the task for execution.

This binding direction is not relevant for this step since no information is to be transported from the workflow to the task.

- In the lower part of the screen, the binding defined from the task container to the workflow container is displayed. This system executes this binding after the user has processed the task.

Here, the binding is _WI_ObjectId. &ABSENCEFORM& has been automatically defined here by the system. This binding ensures that the reference to the notification of absence created is not only known in the single-step task, but also in the workflow.
Unit 4: Defining and Inserting Task “Create Notification of Absence”

You can view the contents of both containers by choosing Workflow container or Task container.

(The element of the task container _WI_Object_ID is called Notif. of absence.)

5. Choose 🔄.
You then return to the step definition.

Entering the agents responsible
On the tab page Control, choose the entry Workflow initiator (expression) in the area agent as the agent responsible [Extern].

The system enters the expression &_WF_INITIATOR& in the Expression field and in the input field beside the checkbox.

You start your executable workflow later manually. At this point in time, the system fills the workflow container element _WF_Initiator automatically with your user name.

The above assignment informs the workflow system that the first work item for creating the notification of absence should be addressed as the “starter” as the workflow.

You return to the screen Workflow definition: Create step: Activity

Concluding step definition

15. Choose ✓ to check the step definition.
16. Choose ✓ to exit the step definition.
17. Choose ✓ to view the entire workflow definition in the workflow area on the screen.

You will notice that your workflow definition now contains a new step (the activity just created). The undefined step is also still there.

18. Choose ✓.

The workflow definition is checked, saved, and, provided it does not contain any errors, activated. The system displays a message if problems are encountered during the test.

19. Exit the Workflow Builder.

Result
You have now completed this unit and can start the next unit.
Unit 3: Starting the Workflow Builder

Use
A workflow definition is created with the help of the Workflow Builder.

Prerequisites
You have now created a workflow template (Unit 2: Creating a Workflow).

Procedure
Calling the Workflow Builder

   The Workflow Builder is then launched with the workflow definition that was processed last. If the Workflow Builder has not yet been called, then an initial workflow definition is displayed.

   The Workflow Builder screen is divided into the following areas.

   ![Diagram of Workflow Builder areas]

   The workflow definition is displayed in the Workflow area on the screen. If the workflow is shown in display mode, the step type area is not displayed. For further information, refer to Workflow Builder [Seite 1002] in the SAP Business Workflow documentation.

4. Choose and enter the abbreviation of your workflow template (<eight-digit number>).
You can use the *F4 input help*. Enter the start of the abbreviation for your workflow (for example: `<ini*>`) and choose 🔄 All of the workflows that match are displayed. Select your workflow and choose 🔄.

**Result**

The workflow definition is displayed and if necessary can be processed. If the Workflow Builder is called for the first processing of a newly created initial workflow definition, the following graphic is displayed:

**Initial Presentation of a Workflow Definition**

![Initial Presentation of a Workflow Definition](image)

If the graphic displayed differs from that above, check the graphic settings in the Workflow Builder. Choose *Extras → Options*, and click the *Graphic* tab. Choose the *Without event-driven process chains* view.

- The start of the workflow definition is indicated by 🔄*Start Workflow*. If the new workflow definition has been called from the definition of a multistep task for which *triggering events* [Extern] are defined, these are represented by the symbol 🔄 and their description.
- The end of the workflow definition is indicated by 🔄*Workflow terminated*.
- The area in which the new workflow definition can be added is indicated by an undefined step with an outcome. Steps are displayed in the form of symbols. The description of an outcome is illustrated in the standard view with an arrow.

This unit is now complete. In the following unit you create the first step in your workflow and define a task for this purpose at the same time.

**Creating an Activity as a Step in the Workflow Definition**

Add a first step of the type *Activity* to the workflow definition.

4. Choose 🔄 under step type. The cursor changes shape.
5. Position the cursor on the *Undefined* step and click.

By doing so, you add an *activity* to your workflow definition. The step definition is shown in the right-hand part of the screen. You are on the tab page *Control*. 
Unit 3: Starting the Workflow Builder

Entering basic data for control

6. Enter the abbreviation (TS<8-digit number>) of the standard task Create Notification of Absence in the Task field and choose ☑.

   If you no longer know the full abbreviation, you can use different methods to search for it:
   - Use the F4 input help to search via the object type and method of the single-step task.
     This brings you to the Search and Find: Tasks dialog box.
     On the tab page Obj. type and method, enter the object type FORMABSENCE and the method CREATE.
     Then double-click your single-step task in the list. You should recognize your single-step task from your initials in the abbreviation. Choose ☑.
   - Enter a character string in the abbreviation or description and choose ☑.
     A list of tasks is then displayed in which you can double-click the desired task.

   Every activity, as a step in a workflow definition, refers to a task.
   At this point, the unique reference to this task is entered. The identification is made up of an abbreviation (T, TS, WF, or WS) followed by an 8-digit number.
   If you select a workflow task (WF) or workflow template (WS) as your task, your activity is a subworkflow [Extern]. If you select a customer task (T) or standard task (TS), your activity is a single step.

   The Define Container Elements and Binding dialog box is then displayed in which the system proposes a change to the workflow container and a binding definition.
   - Upper part of the dialog box: Proposal for new container elements in the workflow container.
     The system proposes that you create a local container element in the workflow container, which can store the reference to the notification of absence created in this step. The element has the technical name AbsenceForm.
   - Lower part of dialog box: Proposed binding definition
     The system proposes that you define a binding from the task container to the workflow container. This binding transports the reference to the generated object (i.e. the created notification of absence) from the task (from the _WI_Object_ID element in the task container) to the workflow (to the AbsenceForm element in the workflow container).

   Further information on automatic binding definition proposals is available under Binding Definitions from the Workflow Container [Seite 1223] and Binding Definitions from the Task Container [Seite 1225] in the documentation on SAP Business Workflow.

13. Confirm the proposal with ☑.

   The new AbsenceForm container element is displayed in the object area of the workflow container folder. Local container elements are marked ☑. The remaining elements in the
workflow container are workflow system variables. For further information, please refer to Workflow System Variables in the Workflow Container [Extern].

14. Enter a text to describe the step.

You can, of course, leave the description of the single-step task that is proposed by the system.

(The current step in your workflow definition appears here.)

15. Go to the Events tab.

All of the outcomes for this step are displayed here.

16. Enter a text for the description.

The description (proposal: <Application filled out>) in the Task executed synchronously line.

Each step has at least one subsequent event (in this case: Task executed synchronously). The description is the text with which the outcome is labeled in the workflow definition.

The Form does not exist and Form could not be generated outcomes are the possible exceptions that could occur in the method execution. You “run the risk” of not intercepting these exceptions and not modeling any exception handling in your workflow definition. If one of the exceptions actually occurs at runtime, the workflow will assume the error status.

Checking the binding definition (optional)

7. Go to the Control tab.

8. Choose Binding (present).

The WF Builder: Binding for Step ‘<Create notification of absence>‘ dialog box is then displayed.

- In the upper part of the screen, the binding defined from the workflow container to the task container is displayed. The system executes this binding before providing the task for execution.

  This binding direction is not relevant for this step since no information is to be transported from the workflow to the task.

- In the lower part of the screen, the binding defined from the task container to the workflow container is displayed. This system executes this binding after the user has processed the task.

Here, the binding is _WI_ObjectId. &ABSENCEFORM& has been automatically defined here by the system. This binding ensures that the reference to the notification of absence created is not only known in the single-step task, but also in the workflow.

You can view the contents of both containers by choosing Workflow container or Task container.

(The element of the task container _WI_Object_ID is called Notif. of absence.)

6. Choose ✔.

You then return to the step definition.
Entering the agents responsible

Choose *Workflow initiator (expression)* under *Agents*.

The system enters the expression &_WF_INITIATOR& in the *Expression* field and in the input field beside the checkbox.

By doing so, you choose the *agent responsible [Extern]*.

You start your executable workflow later manually. At this point in time, the system fills the workflow container element _WF_Initiator automatically with your user name.

The above assignment informs the workflow system that the first work item for creating the notification of absence should be addressed as the "starter" as the workflow.

You return to the screen *Workflow definition: Create step: Activity*

Concluding step definition

20. Choose to check the step definition.

21. Choose to exit the step definition.

22. Choose to view the entire workflow definition in the workflow area on the screen.

You will notice that your workflow definition now contains a new step (the activity just created). The undefined step is also still there.

23. Choose .

The workflow definition is checked, saved, and, provided it does not contain any errors, activated. The system displays a message if problems are encountered during the test.

24. Exit the Workflow Builder.

Result

You have now completed this unit and can start the next unit.
Unit 5: First Test

Use
You are now going to start a workflow in dialog for the first time, which will run according to your workflow definition. This workflow definition only contains a step for creating a notification of absence so far.
For further information on starting workflows, refer to Starting Workflows (Test Environment) [Seite 1452] in the SAP Business Workflow documentation.

Prerequisites
The creation of a notification of absence, which you carry out within your workflow, is reported on a system-wide basis by an event. Other workflows entered as event receivers for this event may therefore be started. The notification of absence created by you (and published by the event) is then also processed with these workflows.
This event is not used in connection with this tutorial.
To ensure that only you work exclusively with the notification of absence, you can deactivate any existing linkages between the event and its receivers. Proceed as follows:

10. Enter FORMABSENC in the Object type field.
11. Enter created in the Event field.
12. Choose Enter.
   The system simulates the event FORMABSENC.Created and establishes which workflows would be started.
13. If you find one or more entries in the list under the branch Tasks without syntax errors to be started, position the cursor on an entry and choose Event linkage.
   The dialog box Event linkage: Triggering events is displayed.
14. Position the cursor on the entry with the red background for the event and choose.
   The linkage between workflow and event is deactivated.
15. Choose.
16. Repeat steps 5 to 7 or repeat the whole simulation as applicable.

Procedure
Starting the workflow and filling out the form
13. If you are still in the Workflow Builder, choose.
Unit 5: First Test

The Start Workflow (Test Environment) screen is then displayed. The number of your workflow template is entered.

If you are no longer in the Workflow Builder, choose Tools → Business Workflow → Development → Runtime Tools → Start Workflow (Test Environment). Enter the identification of your workflow or use the F4 input help.


To handle work processes more quickly, SAP Business Workflow supports Advancing with Immediate Dialog [Extern]. When a workflow is started in dialog, this means that the first work item of the workflow is made available for processing immediately provided that the user who starts the workflow is also one of the recipients [Extern] of this first work item. This is the case here because you entered the container element WF_Initiator as the agent in the step definition.

The Create notification of absence screen, therefore, is displayed immediately where you can see the notification of absence form. It is the method Create that is executed with the first work item.

15. Fill out the form with entries of your choice.


The Start Workflow (Test Environment) screen is then displayed again.

17. Choose and exit the Workflow Builder if necessary.

Starting a workflow / Business Workplace

Before you extend your workflow definition, carry out the following test. It will familiarize you with the Business Workplace.


You go to the screen Start task.

11. Choose your workflow in the table on the left.

12. Choose Start on the right-hand side of the screen.

The notification of absence form is displayed.

13. Do not make any entries and do not save. Instead, choose .

You have now started the workflow but canceled the processing of its first step. But the processing of the work item is not yet completed. You have only broken the processing chain of advancing with immediate dialog.


16. Open the Inbox node and choose the Workflow folder.

Your workflow inbox now contains (at least) one work item for processing. This is the work item for creating a notification of absence, the processing of which you canceled previously.

17. Select the work item, if necessary, and choose .

You return to the Create notification of absence screen where the notification of absence form is displayed.
18. Fill out the form and choose ✅.

   The work item disappears from your workflow inbox.

**Result**

You have now completed this unit and can start the next unit.
Reporting and analysis

In this first test, you will take a look at the work item analysis function.


   The Work Items Per Task screen is then displayed.

9. Select the monitoring period Today.

10. Select the work item type (Sub-)Workflow only.

11. Choose ✂

   The system then determines all of the workflows that were started today. These are then listed on the Work Items Per Task from <Date> to <Date> screen. The number of work items in each workflow is also shown here.

   ⚠

   Make sure that you only ever analyze the work items for your workflow and your absence notification.

12. Display the list of associated work items by double-clicking the workflow ID.

   Both the workflows you just started for this workflow identification are displayed with status Completed.

13. Display the workflow log by double-clicking a work item ID.

   The Workflow Log screen is then displayed.

   For information on the workflow log, see Workflow Log [Seite 1420] in the SAP Business Workflow documentation.

14. Exit the work item analysis function.
Unit 6: Creating a Task "Check Notification of Absence"

Use
You define the task Check notification of absence. This task is incorporated into your workflow definition as the second step.
As this is not the first task you have defined, many steps will doubtless be familiar to you. (Refer to Unit 4: Defining and Inserting Task "Create Notification of Absence [Seite 1593].")

Prerequisites
The object type used (here: FORMABSENC (notification of absence)) and the required method (here: Approve) are already defined and implemented in the Business Object Repository.

Procedure
   The screen Task: Maintain is displayed.
5. In the Task type field, choose the Standard task entry.
6. Choose.
   The Basic data tab in the Standard task: Create screen is then displayed.

Entering the basic data
8. Enter an abbreviation (of your choice) and a name (of your choice) for the task to be created.
   Abbreviation: <ini_check>
   Name: <Check notification of absence (ini)>

Object type and method
9. Enter an object type and a method:
   Object type: FORMABSENC
   Method: APPROVE
   You can also use the F4 input help to select the method.

Work item text
10. In the Work item text field, enter the text Check notification of absence from.
11. Choose. Save your standard task as a local object.
12. You can include variables in your work item text to be filled with values from the task container at runtime..

   The name of the creator of the notification of absence is to be included into the work item text. As soon as the work item appears in the superior's Business Workplace, the name of the creator appears in the work item information text.

   Position the cursor in the work item text after the word of.
13. Choose \( \text{ } \).

The *Please choose an expression* dialog box is then displayed.

14. Choose the expression *Notif. of absence* by double-clicking the *Container node.* → *Issuer object ref. → Name.*

\[ \text{ } \]

If \_WI_Object_ID\_ is displayed instead of *Notif. of absence,* choose \( \text{ } \). The system then displays the description of the expressions instead of their technical names.

The variable \& WI_Object_ID.Creator.Name\& has been added to the work item text automatically. The variable name displayed here is the technical name of the expression.

You can integrate all of the elements in the task container in your work item text. This enables you to include information into the work item text, which is not available until runtime.

**Entering the description text**

7. Go to the *Description* tab.

8. Choose *Task description* in the *Test type* field.

The current task description is then displayed. This text is intended to inform future recipients of a work item in which this task is referenced and help them in their work.

The text will often be similar to the work item text or may supplement it.

9. Choose \( \text{ } \).

10. Enter the following text:

\[ \text{Please check notification of absence no. } \& \_WI_OBJECT_ID\_\text{.NUMBER}\text{ }} \& \_WI_OBJECT_ID\_\text{.CREATEDATE}\text{ of employee } \& \_WI_OBJECT_ID\_\text{.CREATOR.NAME}\text{.} \]

**Decide whether to approve the request.**

If you want to insert an expression from the task container as a text variable, choose *Include → Expression.* Then choose the relevant expression in the *Please choose an expression* dialog box by double-clicking the *Container node.*

11. Choose \( \text{ } \).

12. Choose \( \text{ } \).

**Determining possible agents**

9. Select *Additional data → Agent assignment → Maintain*

This takes you to the *Standard task: Maintain Agent Assignment* screen.

10. Position the cursor on the name of your task.

11. Choose \( \text{ } \).

You go to the dialog box *Choose agent type.*

12. Double-click *Job.*

The dialog box *Choose Job* is displayed.
13. Enter either part of or the full abbreviation of the job that you created for the head of department in Unit 1: Organizational Plan [Seite 1587].

You return to the screen Standard task: Maintain agent assignment.


You have linked the task to the job of a head of department. In the organizational plan, you use this job to describe a position. The holder of the position is also displayed.

15. Choose 🔄.


**Result**

The task for checking the notification of absence is defined in full. You can now exit the screen for defining a task.

In the next unit, you will incorporate this task into the workflow definition.

Display your organizational plan again.


f. Choose your organizational unit in the search area. Display it by double-clicking the entry in the table.

g. In the overview screen, choose the arrow 🔄 on the right next to the 🔄 and then Staff assignments (list).

h. Display a detailed view of the position of the head of department by double-clicking the entry in the list.

Here, you can see that this position is assigned to the job and, on the Tasks tab, that it is also assigned to the task Check notification of absence.
Unit 7: Including "Check Notification of Absence" in the Workflow Definition

Use

The task with which an employee can check a notification of absence is ready to be included into your workflow definition as the next step.

Prerequisites

You have defined a workflow with a step (Unit 4: Task Defining and Inserting "Create Notification of Absence" [Seite 1597]) and created the task Check Notification of Absence (Unit 6: Create Task "Notification of Absence" [Seite 1606]).

Procedure

Calling the Workflow Builder


   The Workflow Builder is then launched in display mode with the workflow that was processed last.

5. Work through the following steps to display your workflow for this tutorial in change mode.

   iv. If the workflow you created for this tutorial is already displayed, choose 📊. Go to step 3.

   v. If a different workflow is displayed, check whether the workflow you created is listed in the My available tasks area. Double-click it and choose 📊. Go to step 3.

   vi. Choose 📊. Enter the complete abbreviation in the Task field on the Open other workflow definition dialog box and press RETURN. You can now also enter the workflow abbreviation. Enter either the full abbreviation or just the first characters (for example: <1*>). Press F4. The search result is then displayed. Choose your workflow by double-clicking it. When you enter the abbreviation in the Task field on the Open other workflow definition dialog box, all of the available versions are displayed. Since you have not created any versions, select version 0000 and choose 📊. The workflow is then loaded in display mode. To switch to change mode, choose 📊.

   6. Choose 📊 to display the entire workflow in the workflow area on the screen.

Creating a container element in the workflow container

You need an additional container element in the workflow container to store the name of the user who will execute this step. This user name is to be used later in the notification text that is sent to the requester.

4. Choose the entry <Double-click to create> by double-clicking in the Workflow Container.

   The dialog box for entering a container element is displayed.

5. Make the following entries:

   Element: Approver
Unit 7: Including "Check Notification of Absence" in the Workflow Definition

Name: Approver
Description: Approver of the absence request
Reference table: WFSYST
Reference field: AGENT

6. Choose .

You have now added a local container element to the workflow container, which is ready to take an agent name, based on its data type reference. However, you have not yet determined a value for this container element.

Creating an Activity as a Step in the Workflow Definition

20. Choose under step type. The cursor changes shape.

21. Position the cursor on the Undefined step that follows the outcome Request completed and click.

By doing so, you add an activity to your workflow definition. The step definition is shown in the right-hand part of the screen. You are on the tab page Control.

Entering basic data for control

22. In the Task field, enter the abbreviation (TS<8-digit number>) of the task Check Notification of Absence that you defined in the previous unit. Choose .

If you no longer know the full abbreviation, you can use different methods to search for it:

- Use the F4 input help to search via the object type and method of the task.
  This brings you to the Search and Find: Tasks dialog box.
  On the tab page Obj. type and method, enter the object type FORMABSENCE and the method CREATE.
  Then double-click your task in the list. You should recognize your task from your initials in the abbreviation. Choose .

- Enter a character string in the abbreviation or description and choose .
  A list of tasks is then displayed in which you can double-click the desired task.

Every activity, as a step in a workflow definition, refers to a task.

At this point, the unique reference to this task is entered. The identification is made up of an abbreviation (T, TS, WF, or WS) followed by an 8-digit number.

If you select a workflow (WS or WS) as your task, your activity is a subworkflow [Extern]. If you select a task (TS or T), your activity is a single step.

You go to a dialog box in which the system proposes binding from the workflow container to the task container.

Here, the object reference to the notification of absence is transported from the workflow container (container element AbsenceForm) to the task container (to the container element _WI_Object_ID).
Unit 7: Including "Check Notification of Absence" in the Workflow Definition

23. Choose to confirm the proposal.

**Extending the binding definition**

Now define another binding via which the current agent [Extern] of the work item (the superior) is stored in a container element of the workflow container.

24. Choose Binding (present).

The Binding for Step 'Check notification of absence' dialog box is then displayed. If the first column is entitled Element, choose to display the element description.

The system has created the following binding definition from the workflow container to the task container:

Notif. of absence  &AbsenceForm&

This binding ensures that the correct notification of absence is checked in the task.

25. Choose.

All of the container elements in the task container are now displayed on the left-hand side. You can define a binding from these container elements to the workflow container.

26. Position the cursor in the lower half of the screen, in the empty input field beside the container element Actual agent and call the F4 input help.

27. Double-click Approver.

In addition to the bindings already existing, the following binding is now defined:

Actual Agent  &Approver&

28. Choose.

**Determining outcomes**

The method APPROVE of the object type FORMABSENC, to which you refer in the standard task definition, is defined with a result [Extern]. The three possible values of this result are offered on the tab page Outcomes as outcomes of this step:

- Approved
- rejected
- New

The indicates that these outcomes have already been transferred to the workflow definition. The outcome New is not really appropriate at this point. At this stage of processing, the notification of absence cannot be “new” any more. It should therefore not be incorporated in the definition.

3. Click on the symbol in front of the outcome New.

The symbol changes to .

The outcome Processing obsolete is processed if the relevant work item is set to obsolete via a process control step. This functionality is not used in this tutorial.

The outcome Form not available results from the exception [Extern] defined for the method APPROVE. You could include this outcome into the workflow definition and would then have to model appropriate subsequent steps. However, you do not model
Unit 7: Including "Check Notification of Absence" in the Workflow Definition

anything for the purposes of this tutorial and accept that your workflow will have an error at runtime if this exception is raised.

4. Choose ✔.

Two branches are now inserted in your workflow definition for the two marked outcomes Approved and Rejected.

Inputs for the agent

If you are working through this tutorial alone, do not specify anything for responsibility at this point. If it is processed when the workflow is executed, therefore, this step will be “offered” for processing to all employees declared as possible agents [Extern] of the single-step task Check notification of absence. This means all user whose positions are described with the job head of department.

In your organizational plan, there is only one users who is a possible agent -> you.

- You have maintained the organizational plan with jobs, positions, and user assignments in Unit 1: Organizational Plan [Seite 1587].

- You have defined the single-step task “Check notification of absence” and have assigned the possible agents in Unit 6: Definition of the Standard Task “Check Notification of Absence” [Seite 1606].

Concluding step definition

Choose ✔.

The workflow definition is checked, saved, and, provided it does not contain any errors, activated. The system displays a message if problems are encountered during the test.

Result

You can subject your workflow definition to a second test.
Unit 8: Second Test

Use
You start your workflow in dialog and two steps are executed. The notification of absence is created in the first step and then checked in the second.

Procedure
Starting the workflow and filling out the form
18. If you are still in the Workflow Builder, choose 🔄.

The Start Workflow (Test Environment) screen is then displayed. The number of your workflow is entered.

If you are no longer in the Workflow Builder, choose Tools → Business Workflow → Development → Runtime Tools → Start Workflow (Test Environment). Enter the number of your workflow or use the F4 input help.

19. Choose 🔄.

You go to the screen Create Notification of Absence.

From the requester’s perspective
When the workflow is started, the work item for creating the notification of absence is provided to you directly for processing. This is because of advancing with immediate dialog. You were introduced to this in the first test.

4. Fill out the form and choose 🔄.

From the head of department's perspective
You created the step for checking the notification of absence in your workflow definition without specifying the agents responsible. This means that all of the possible agents of the single-step task are recipients of the work item. Since you occupy the position of both requester and head of department, you are automatically and immediately presented with the work item for approving the application. This is again because of advancing with immediate dialog.

As the superior, you have several options:
• Approve the request
• Reject the request
• Cancel processing of the request.

5. Choose 🔄 to cancel processing.

The Start Workflow (Test Environment) screen is then displayed again.

Exit this dialog and, if necessary, the Workflow Builder.

Reporting and analysis: Workflow outbox
In the first test of your workflow, you were introduced at this point to work item analysis. Now, you will be introduced to the workflow outbox.

Amongst other things, the work items which you started in dialog along with date, time and current status are displayed in the workflow outbox.

11. In the tree displayed, choose Outbox → Started workflows.

12. Choose the arrow on the right next to the and then Today only.

   The work items for all the workflows you started today are then displayed. Note the
   statuses of the work items displayed in the Status column of the Business Workplace.
   Read the F1 help for the Status column.

13. Position the cursor on the work item for the workflow just started. This work item has the
    status in process.

    From the current workflow data, you can see that the step Create notification of absence
    has been completed successfully, when that was, and who processed it. The work item
    has the status completed.

    You can also see that the work item representing the task for checking the request still
    has status ready and can therefore still be seen in the superior's Business Workplace.

**Business Workplace - check notification of absence**

You now once again occupy the position of the superior who canceled processing earlier when the
application was to be checked.


   The work items for you to process are displayed with their work item texts and certain
   other attributes.

   A preview of the selected work item is displayed below the list and contains the
   description of the work item.

15. Select the work item for checking the notification of absence and choose .

    You go to the work item display. There you find amongst other things the description text,
    also with replaced text variables.

16. Choose to return to the Business Workplace.

17. Choose or start execution by double-clicking the entry.

18. Either reject or approve the request. You should start the workflow twice and test each
    variant.

To check whether the workflow was completed correctly, go to your workflow outbox. The procedure is
described above.

**Result**

Your workflow definition has now been tested and you can continue with the next unit.
Unit 9: Adding a User Decision

Use
In the previous units, you created a workflow whose definition contains two activities. These activities are based on two standard tasks that you created. In this unit, you will add a user decision [Extern] to your workflow definition. With the user decision, the requester can decide to revise and resubmit the notification of absence if the head of department rejects it. For further information, refer to Maintaining a User Decision [Seite 1074] in the SAP Business Workflow documentation.

Procedure
You start the Workflow Builder and open your workflow definition in Change mode. Proceed as described in Unit 7: Including "Check Notification of Absence" in the Workflow Definition [Seite 1609].

Creating a user decision as a step in the workflow definition
8. Choose under step type. The cursor changes shape.
9. Position the cursor on the Undefined step that follows the outcome Rejected and click. By doing so, you add a user decision to your workflow definition. The step definition is shown in the right-hand part of the screen. The Decision tab is displayed.

Entering basic data for user decision
10. Enter Your request was rejected by &. Revise? in the Title field. This text is used as the title for the alternative decisions when the workflow is executed. The variable & is a placeholder for a parameter that is filled from the task container at runtime.
11. Choose the F4 input help in the Parameter 1 field. The Expression for first parameter of user decision dialog box is then displayed.
12. Choose the Approver container element by double-clicking it.

Entering alternative decisions
13. Define the alternative decisions. Enter the following texts:

<table>
<thead>
<tr>
<th>Decision texts</th>
<th>Name</th>
</tr>
</thead>
<tbody>
<tr>
<td>Decision: Revise request?</td>
<td>Revise</td>
</tr>
<tr>
<td>Decision: Withdraw request?</td>
<td>Withdraw</td>
</tr>
</tbody>
</table>

The decision texts you enter here appear as decision options on the screen that can be processed by the requester after their leave request has been rejected. The descriptions are the terms used to describe the outcomes in the workflow definition.

Entering the agents responsible

The system enters the expression &_WF_INITIATOR& in the Expression field and in the input field beside the checkbox.
Concluding step definition

12. Choose ✂️ to check the step definition.
13. Choose ✔️ to exit the step definition.
14. Choose 📝 to view the entire workflow definition in the workflow area on the screen.

   You can see that your workflow definition contains a user decision in the rejected branch. This has two outcomes whose branches converge again in the rejected branch. Each new branch contains an undefined step.
15. Choose ✨.

   The workflow definition is checked, saved, and, provided it does not contain any errors, activated. The system displays a message if problems are encountered during the test.

Result

You can now subject your workflow definition to another test.
Unit 10: 3rd Test

Use
You start your workflow in dialog and three steps are executed: Two activities, for creating and checking a notification of absence, and one user decision.

Procedure

Starting the workflow and filling out the form, checking the notification of absence, user decision

20. If you are still in the Workflow Builder, choose .
   The Start Workflow (Test Environment) screen is then displayed. The number of your workflow is entered.
   If you are no longer in the Workflow Builder, choose Tools → SAP Business Workflow → Development → Runtime tools → Start workflow (test environment). Enter the number of your workflow or use the F4 input help.

   You go to the screen Create Notification of Absence.

From the requester’s perspective

22. Fill out the form and choose .
   When you save the form, you (as superior) can immediately approve or reject the application (reason: advancing with immediate dialog).

From the superior’s perspective

23. Do not approve the application.

24. Choose .
   If the request is not approved, the next step in the workflow definition is the user decision. The requester was entered as the agent for the step. The decision is therefore offered to you again as the requester (reason: advancing with immediate dialog).

From the requester’s perspective

You are given three options:

- Revise request?
- Withdraw request?
- Cancel

It is irrelevant which option you choose since you have not defined any follow-up steps. You should start the workflow several times and try out the different options. The complete process flow can be checked each time in the work item analysis or workflow outbox.
Unit 11: Define and Include "Revise Notification of Absence" in the Workflow Definition

Use
You will now add the activity Revise notification of absence to your workflow definition. You will create the necessary task within the workflow definition as in unit 4.

Procedure
You start the Workflow Builder and open your workflow definition in Change mode. Proceed as described in Unit 7: Including "Check Notification of Absence" in the Workflow Definition [Seite 1609].

Creating an activity as a step in the workflow definition
3. Choose under step type. The cursor changes shape.
4. Position the cursor on the Undefined step that follows the outcome Revise after the user decision, and click.

By doing so, you add an activity to your workflow definition. The step definition is shown in the right-hand part of the screen. You are on the tab page Control.

Creating a standard task for use in this step
13. Choose the arrow on the right next to the and then Create task.

The Basic data tab in the Standard task: Create screen is then displayed.

14. Enter the following data.
   
   Abbreviation:  
   Name:  
   Work item text: Revise notification of absence 
   Object type: FORMABSENC
   Method: UPDATE

You can also use the F4 input help to select the method.

15. Save your standard task as a local object.
16. Select Additional data → Agent assignment → Maintain
17. Position the cursor on the name of your standard task.
18. Choose Properties…
19. Set the General task indicator.
20. Choose Copy and then .

The screen for editing the standard task is then displayed.

21. Choose and then .

You then return to the step definition in the Workflow Builder. For this task, a binding must be defined between the workflow and task containers. The system proposes a
Unit 11: Define and Include “Revise Notification of Absence” in the Workflow Definition

binding and displays it for you to check in the Define container elements and binding dialog box. This proposal defines that the object reference to the notification of absence is transported from the workflow container (container element AbsenceForm) to the task container (container element _WI_Object_ID).

22. Confirm the proposal with ✔.

The system has entered the abbreviation for the new standard task in the Task field. The Step description field contains a description of the standard task. You can change this entry if necessary.

23. Switch to the Outcomes tab and enter a description (for example request revised) for the Task executed synchronously outcome.

Entering the agents responsible


The system enters the expression &_WF_INITIALIZER& in the Expression field and in the input field beside the checkbox.

By doing so, you choose the agent responsible [Extern].

Concluding step definition

25. Choose ✔ to check the step definition.

26. Choose ✔ to exit the step definition.

27. Choose ✔ to view the entire workflow definition in the workflow area on the screen.

You will notice that your workflow definition now contains a new step (the activity just created). The undefined step is also still there.

28. Choose ✔.

The workflow definition is checked, saved, and, provided it does not contain any errors, activated. The system displays a message if problems are encountered during the test.

Result

You can now subject your workflow definition to another test. Carry out the test as described in Unit 10: Third Test [Seite 1617].

Test proposal

To test the new step, proceed in the following order:

5. Create notification of absence.

6. Reject application

7. Revise user decision.

8. Revise request (the new step)

Since you are both the requester and approver and because advancing with immediate dialog is activated, all of the steps are presented to you directly.
Unit 12: Integrating the UNTIL Loop for Reapproval

Use
If the requester decides to revise and resubmit the notification of absence to their superior, the step Check notification of absence must be executed again.

There are different ways of solving this problem. The solution outlined here is only one example.

- You create a container element in the workflow container that you use as a flag. This flag contains different values, depending on the status of the notification of absence:
  - Approved
  - Not approved and revised
  - Not approved and not revised

To assign values to the container element, container operations must be executed on the workflow container.

For information on the workflow log, see Definition of the Workflow Container [Seite 1213] in the SAP Business Workflow documentation.

- You use this container element to define the condition in the UNTIL loop [Extern].

  For further information, refer to Maintaining an UNTIL loop [Seite 1087] in the SAP Business Workflow documentation.

Procedure
You start the Workflow Builder and open your workflow definition in Change mode. Proceed as described in Unit 7: Including "Check Notification of Absence" in the Workflow Definition [Seite 1609].

Create container element in the workflow container
4. Choose the entry Double-click to create by double-clicking in the Workflow Container.

The dialog box for entering a container element is displayed.

5. Make the following entries:
   - Element: Flag
   - Name: Flag
   - Description: Flag for approval status
   - Reference table: SYST
   - Reference field: INPUT

6. Choose ✅.

You have now added a local container element to the workflow container, which is ready to store a flag, based on its data type reference. However, you have not yet determined a value for this container element.

Integrating the UNTIL loop as a step
29. Choose 🕰 under step type. The cursor changes shape.
30. Move the cursor to the *Undefined* step before the event workflow end and click.

   By doing so, you add an *UNTIL loop* to your workflow definition. The step definition is shown in the right-hand part of the screen.

**Entering the basic data**

31. Enter *Resubmission required?* in the *Step description* field.

32. Click on the condition area to open the condition editor.

   The bottom half of the screen contains the empty condition, whereas the top half shows the system fields as well as the content of the workflow container. The field in the *Expression 1* column is ready for input and marked 📊.

33. Choose the *Flag* container element as the first expression by double-clicking it.

   The first expression is now entered in the condition in the lower part of the screen. The *Expression 2* field is now ready for input and marked 📊.

34. Choose 📊 as the *operator*.

35. Enter the constant *X* in the field for *Expression 2* and confirm by pressing *Enter*.

36. Choose ✔.

   You have now defined the condition $\text{Flag} = \text{X}$.

   For further information, see *Condition Editor [Seite 1012]*.

   The comparison between the container element *Flag* and the constant *X* returns either the result *true* or the result *false*. Two outcomes are therefore possible.

37. Enter the following texts for the outcomes:

   - *True*: *Flag equal to X - no resubmission*
   - *False*: *Flag not equal to X - resubmission*

   If the result of the evaluated condition is *false*, the loop is processed again. If the notification of absence is to be resubmitted to the superior, the container element *flag* must not have the value $\text{X}$. For this purpose, you will insert *container operations [Extern]* in the workflow definition in the next unit.

38. Choose ✔.

   You have now inserted an isolated UNTIL loop into the workflow definition, which does not yet contain any steps. You must change the workflow definition so that all of the steps that are required to check and resubmit the notification of absence are within the UNTIL loop.

   In order to understand the steps required, you must familiarize yourself with block operations.

**Block operations**

3. Choose 📊 to display the entire workflow definition in the workflow area on the screen.

4. Display the block structure of your workflow definition. Choose *Graphic → Blocks → Show*.

   The workflow definition is block-oriented. Every block represents a basic structural element that is a self-contained and consistent arrangement of steps and outcomes.
If you, for example, create a new step that has one outcome, this pair (step, outcome) represents a block. Operations that are performed on one step (delete, cut,...) always affect the entire block associated with that step.

In the above example, in which the block consists of the pair (step, outcome), using the delete function will also delete the corresponding outcome as well.

You should test the operations listed under Edit → Block on blocks:

- First save your workflow definition.
- Test the various block operations.

You can undo changes at any time by choosing . Another possibility is to exit the Workflow Builder without saving your work and to call it again.

For further information, refer to Block Orientation in the Workflow Definition [Extern].

**Copying steps to the UNTIL loop**

The step Check notification of absence, which follows the outcome Request completed, is the first step in a new block. To implement the resubmission, the workflow must send the notification of absence to this step after it has been revised by the requester. You have used an UNTIL loop for this purpose. The step Check notification of absence must be located within this loop. In order to do so, first cut the block that begins with this step. Then paste the cut block in the UNTIL loop.

7. Select the step Check notification of absence.

8. Choose to cut the block.

9. Select the Undefined outcome in the UNTIL loop and choose .

10. Choose to align the graphic.

11. Choose to view the entire workflow definition in the workflow area on the screen.

You will notice that your workflow definition now contains steps within the UNTIL loop.

12. Choose .

The workflow definition is checked, saved, and, provided it does not contain any errors, activated. The system displays a message if problems are encountered during the test.

**Result**

The integration of the UNTIL loop is now complete. The workflow up to this point, however, delivers an incorrect result, since the Flag container element is not assigned a value during execution. The result of the evaluated condition in the UNTIL loop is always false and the workflow can never be completed. To ensure that the workflow is executed correctly, you will add further container operations to the workflow definition in the next unit.
Unit 13: Inserting Container Operations

Use
You have added an UNTIL loop to your workflow definition. To ensure that this loop condition is executed correctly, you have to add container operations [Extern] to your workflow definition that change the value of the flag container element in accordance with how the workflow is executed.

Prerequisites
You have added an UNTIL loop to your workflow definition.

Procedure
You start the Workflow Builder and open your workflow definition in Change mode. Proceed as described in Unit 7: Including "Check Notification of Absence" in the Workflow Definition [Seite 1609].

Integrating container operations as a step
You can manipulate the individual elements of the workflow container using a container operation type step.
You need to insert a container operation at three different points in your workflow definition.

- After the outcome Request revised
  At this point, you assign the workflow container element Flag the value Z. The comparison in the UNTIL loop then returns the result false. In this case, the workflow continues via the outcome Flag does not have value X and the request is resubmitted to the superior’s Business Workplace.

- After the outcome Withdraw
  At this point, you assign the workflow container element Flag the value X. The comparison in the UNTIL loop returns the result true. The loop is exited.

- After the outcome Approved:
  You assign the container element Flag the value X. The comparison in the UNTIL loop returns the result true. The loop is exited.

Container operation after the outcome revised event

6. Choose ☑ under step type. The cursor changes shape.
7. Position the cursor on the Undefined step that follows the outcome Request revised and click.
   By doing so, you add a container operation to your workflow definition. The step definition is shown in the right-hand part of the screen.
8. Enter the following basic data:
   
   **Step name:** Set flag to Z
   
   **Outcome name:** Flag = Z

9. Specify the following for the operation:
   
   **Result element:** Flag  (select using F4 input help)
   
   **Expression:** Z
The input fields for the operator and second expression must be empty.

When this step is executed, the Flag container element in the workflow container receives the value Z. This value remains as long as it is not changed explicitly.

10. Choose ✔.

**Container operation after the Withdraw outcome**

6. Choose 🡡 under step type. The cursor changes shape.

7. Position the cursor on the Undefined step that follows the outcome Withdraw and click.

   By doing so, you add a container operation to your workflow definition. The step definition is shown in the right-hand part of the screen.

8. Enter the following basic data:

   - **Step name:** Set flag to X
   - **Outcome name:** Flag = X

9. Specify the following for the operation:

   - **Result element:** Flag (select using F4 input help)
   - **Expression:** X

The input fields for the operator and second expression must be empty.

10. Choose ✔.

**Container operation after the Approved outcome**

Since this container operation is identical to the one created above, you can simply copy it.

3. Select the container operation Set flag to X after the Withdraw outcome and choose 🡡.

4. Select the Undefined step that follows the Approved event and choose 🡡.

**Concluding step definition**

Choose ✔.

The workflow definition is checked, saved, and, provided it does not contain any errors, activated. The system displays a message if problems are encountered during the test.

**Result**

You have now extended the workflow definition in such a way that the results of the condition evaluated in the UNTIL loop are meaningful. You can now subject your workflow definition to another test. Carry out the test as described in Unit 10: Third Test [Seite 1617].
Unit 14: Changing Staff Assignments

Use
In the tests you have carried out so far you acted both as the requester and as the superior.
In this unit, you will change the organizational plan in such a way that the requester and superior are two
different R/3 users.

Procedure
You enter a different user as the superior in the staff assignments of your organizational plan. Ask a
colleague whether they would like to take part in this test. Or have yourself a second user created.

8. Choose Tools → SAP Business Workflow → Development → Definition tools →
Organizational Management → Organizational plan → Change.

   This takes you to the Change Organization and Staffing (Workflow) screen.

9. Search for your head of department position in the search area. The positions that were
found are displayed in the selection area. Open yours by double-clicking on the entry in the
change mode.

Deleting position holders
10. Select the row in the overview area in which the assignment to the user is entered.
11. Choose the arrow on the right next to the and then Assignment.

   The position of the department head is no longer occupied.

Reassigning position holders
12. Choose User in the search area and search for the user whom you want to assign to the
position.

   All of the users that match the search criteria are displayed in the selection area.

13. Select the user whom you want to assign to the position and drag him or her to the line
position: head of department (ini) in the overview area. Confirm the adaptations of the
relationship period.

   This completes the changes to the staff assignments. The two positions are held by
different persons.

Result
You have changed the staff assignments by changing the assignment of users to a position. You can check
this by displaying the staff assignments. Proceed as described in Unit 1: Organizational Plan [Seite
1587].

You can also start with Unit 16: Fourth Test [Seite 1629] directly after this unit. The recipients of the
work item for the step Check notification of absence are then determined from the possible agents for the
single-step task. You assigned the job of the head of department as the possible agents of this task in Unit 6: Create task “Check Notification of Absence” [Seite 1606].
Unit 15: Using a Role

Use

In the organizational plan you have created up to now, there are two positions, each of which is described by a job. Each position must be linked to a different user. The job of department manager is linked to the department manager position. The job of department manager, in turn, is linked to the task Check notification of absence. In a more extensive organizational plan containing several positions for department managers, each position can be linked to the job of department manager. These links ensure that each user who is assigned to a position as department manager is one of the possible agents of the task Check notification of absence. If you want to assign all of the department managers as possible agents of a different task, all you have to do is link this task to the job of department manager. You do not need to make any detailed changes.

You can add a further organizational unit to your organizational plan by creating a new position as department manager and linking it to a third user. When you link this new position to the existing job of department manager, the third user is automatically added to the list of possible agents for the task Check notification absence.

If you then create a notification of absence, it is offered to all of the possible agents for processing. This is clearly undesirable, since the notification of absence should only be presented to your own superior. This superior is the head of department who has the chief position in the organizational unit to which you too belong.

To ensure that the notification of absence is not submitted to all of the possible agents, you must define agents responsible in your step definition. All of the agents responsible who are also possible agents for this task then become recipients of the work item for the task. In this example, the agents responsible must be defined dynamically, since the relevant agent depends on the requester (person initiating the workflow). To do so, you use a role.

The role Superior of is shipped as standard with the R/3 System. This role first establishes a user's position and the relevant organizational unit, and then finds the chief position in this organizational unit.

This role only works correctly if you are not your own superior. You should only work through this unit if you have completed Unit 14: Changing Staff Assignments and adapted your organizational plan accordingly.

By using roles to assign responsibilities, you do not need to have any specific knowledge of the organizational plan when you define a workflow definition. When the role is resolved at runtime, all of the necessary information is provided via the binding definition from the workflow container to the role container.

Procedure

You start the Workflow Builder and open your workflow definition in Change mode. Proceed as described in Unit 7: Including "Check Notification of Absence" in the Workflow Definition.

Selecting a role

3. Open the step definition for the activity Check notification of absence by double-clicking the activity icon in the workflow area on the screen.
4. Choose **Superior of workflow initiator** under Agents.

   The system then enters the role ID (0000016B) in the Role selection field and in the input field.

   The binding between the workflow container and the role container is created automatically here. If you choose Role in the checkbox and enter the role in the input field using the F4 input help, you must define the binding manually.

### Concluding step definition

16. Choose to check the step definition.

17. Choose to exit the step definition.

18. Choose .

   The workflow definition is checked, saved, and, provided it does not contain any errors, activated. The system displays a message if problems are encountered during the test.

### Result

You can now subject your workflow definition to another test.
Unit 16: Fourth Test

Use
In the organizational plan that you changed in Unit 14: Adjusting the Staff Assignments [Seite 1625], your colleague has now assumed the tasks of the superior. As the requester, you must start the workflow. The work item for checking the request will then be sent to the Business Workplace of the superior.

Procedure

Procedure as requester

   All of the workflows that you are allowed to start are listed on the left-hand side of the screen. A description of the selected workflow is shown on the right.

11. Select your workflow and choose Start.

   The work item for creating the notification of absence is proposed to you immediately for processing.

12. Fill out the form and choose .

   You have now submitted your request and your superior must decide whether to approve it. Since you are not now the recipient of the next work item, advancing with immediate dialog does not have any effect.

Procedure as superior
20. Choose Office → Workplace to start the Business Workplace.

21. Open the folder Inbox → Workflow.

   The request awaiting approval is displayed as a work item in this folder.

22. Select the work item.

   The task description of the single-step task is displayed in the work item preview. As you can see, the expressions used in the task description have been replaced with the current content.

23. Choose but do not approve the request.

24. Choose .

   You are now in your Business Workplace. The work item you have just executed is no longer displayed in the list.

Procedure as requester
25. Choose Office → Workplace to start the Business Workplace.

7. Open the folder Inbox → Workflow.

   This folder contains a work item awaiting a user decision.

26. Select the work item and choose .

27. Select Revise application in the user decision dialog box.
Since the flag *advancing with dialog* is set in the step definition for the activity *Revise notification of absence*, the request is immediately presented to you for revision. If this flag is not set, you must execute the work item from your Business Workplace.

28. Change the request and choose and .

You are now in your Business Workplace. The work item you have just executed is no longer displayed in the list.

**Procedure as superior**

If you have not closed your Business Workplace, choose to refresh the work item display. You have received the request from the requester again. They can proceed as above to approve the request or reject it again. This workflow does not have a modeled response to a situation in which the requester and superior do not reach agreement. The workflow definition can be easily extended for this situation (for example, automatic notification of a third person if the request is exchanged between the employee and head of department several times).

**Reporting and analysis with the workflow log**

You should start the workflow several times together with your colleague and try out the alternatives. These alternatives can then be analyzed via work item analysis or the workflow outbox.

Use the workflow log in this unit. If the workflow you want to analyze is displayed in the work item analysis or in the workflow outbox, select the entry and choose . The workflow log provides information about all the stages of processing. This includes the following:

- Which steps were executed?
- Who was the agent?
- When was processing carried out?
- What was the result of the processing?

If the workflow log is displayed with ActiveX, you can view the various steps in the workflow in the form of a graphic. To do so, choose . For further information on the workflow log, please refer to *Workflow Log: Standard View [Seite 1420]*, and on the ActiveX view under *Personal Settings [Seite 1395]*.
Unit 17: Including "Send Notification" into the Workflow Definition

Use
You have created a complete workflow with which a notification of absence can be created, revised and approved.
In this unit, you will insert a step in the workflow definition that sends a notification to the requester after the request has been approved.

Procedure
You start the Workflow Builder and open your workflow definition in Change mode. Proceed as described in Unit 7: Including "Check Notification of Absence" in the Workflow Definition [Seite 1609].
A separate step type is available for the sending of mails.

8. Select the outcome Approved of the step Check notification of absence and choose Create in the context menu.

The dialog box Select step is displayed.

9. Choose Send mail.

The step definition is then displayed. The input fields required for sending a mail are found on the tab page Mail.

10. Due to the fact that the mail is to be sent to the requester (the workflow initiator), you do not need to change the standard settings. Inspite of this, choose the entry Workflow initiator in the field Recipient type.

When entering a recipient, you must first select a recipient type. The entries have the following meanings:

- **Workflow initiator**: Choose this entry if only the workflow initiator is to receive a mail. The recipient type is automatically set to Organizational object and the expression \&_WF_INITIATOR\& is entered in the field Recipient. These entries are the standard settings.

- **Organizational object**: You must enter an Organizational object [Extern] in the field Recipient or choose an expression that contains one or several organizational objects. Use the F4 input help when making the entries. When you enter an expression, the system automatically creates a container element in the task container and adds the binding definition.

- **E-Mail address**: In the field Recipient, enter either an e-mail address directly or by using the F4 input help, select a container element from the workflow container that contains one or several Internet addresses at runtime. The corresponding expression is entered. The system automatically creates a container element in the task container and adds the binding definition.

When you change the recipient type for the first time, the system automatically creates the necessary task for this step. You must also enter the abbreviation and the name of this task.

11. Enter the basic data for the task:
Unit 17: Including "Send Notification" into the Workflow Definition

**Abbreviation:** <ini_nottxt>

**Name:** <Notification request accepted (ini)>

In the Create Object Directory Entry dialog box, choose Local object.

12. Enter the following text for the mail subject:

<Your leave request was reserved>

13. Enter the following text that the system is to send to the requester if the request is approved:

<Dear colleague,

Your leave request number &AbsenceForm.Number& from &AbsenceForm.CreateDate& was approved on &AbsenceForm.ApprovDate& by &AbsenceForm.Approver.Name&.>

If you want to include an expression that is to be replaced at runtime, choose 📈. Choose the expression from the container elements of the workflow container. The system automatically creates a container element in the task container and adds the binding definition.


**Activating workflow definition**

19. Choose 📚.

The workflow definition is checked, saved, and, provided it does not contain any errors, activated. The system displays a message if problems are encountered during the test.

**Test**

Test your workflow definition in the usual way.

You receive a mail when your “superior” has approved your "notification of absence". You find the mail in your Business Workplace in the folder *Inbox → Unread documents*. 
Unit 18: Work Item Attachments

Use
The superior is to have the opportunity to add an attachment to their work item for checking the notification of absence. This attachment is a document (any SAPscript or PC document), which is then made available to all subsequent recipients in the workflow. The superior can use this to give a full explanation for a rejection.

An attachment can be added to a workflow before the workflow is actually executed in the Business Workplace.

To create an attachment after the workflow has been executed, you have to change the definition of the associated standard task.

To enable the superior to create an attachment after the notification of absence has been checked, you must set the Confirm end of processing flag for the task Check notification of absence. This property means that the agent has to confirm the end of processing of the work item explicitly. The agent can also add an attachment before this confirmation.

For further information about using documents in workflows, refer to Document Processing [Seite 1237] in the SAP Business Workflow documentation.

Procedure
You start the Workflow Builder and open your workflow definition in Change mode. Proceed as described in Unit 7: Including "Check Notification of Absence" in the Workflow Definition [Seite 1609].

Changing the task "check notification of absence"

6. Open the step definition for the step Check notification of absence by double-clicking it.

7. Double-click the task ID to open the task definition.

You go to the screen Standard task: Display.

8. Choose to switch to change mode.

9. Set the Confirm end of processing flag under Execution.

10. Choose and then .

Concluding step definition

4. Choose to check the step definition.

5. Choose to exit the step definition.

6. Choose .

The workflow definition is checked, saved, and, provided it does not contain any errors, activated. The system displays a message if problems are encountered during the test.

Result
Execution of the work item for the step Check notification of absence must be confirmed expressly. This enables you to create an attachment after the request has been approved or rejected.

You can also create attachments while a work item is being executed. An example of this is provided in the demo workflows. Choose Tools → Business Workflow →
Development → Environment → Start demo workflows. Start the demo workflow Demo for WI execution with WF toolbox. Further information is provided on the right-hand side of the screen.
Unit 19: Fifth Test

Use
After you have executed the step Check notification of absence, you must confirm the end of processing.

Procedure

Procedure as requester
   All of the workflows that you are allowed to start are listed on the left-hand side of the screen. A description of the selected workflow is shown on the right.

14. Select your workflow and choose Start.
   The work item for creating the notification of absence is proposed to you immediately for processing.

15. Fill out the form and choose .
   You have now submitted your request and your colleague must decide whether to approve it. Since you are not now the agent of the next activity, advancing with immediate dialog does not have any effect.

Procedure as superior
29. Choose Office → Workplace to start the Business Workplace.
30. Open the folder Inbox → Workflow.
   The request awaiting approval is displayed as a work item in this folder.

31. Select the work item.
   The task description of the task is displayed in the work item preview. As you can see, the expressions used in the task description have been replaced with the current content.

32. Choose but do not approve the request.

33. Choose .
   The dialog box for confirming the end of processing appears.

34. Choose the arrow on the right next to the and then Create attachment.
   The Create Document: Header dialog box is then displayed.

35. ChooseRAW as the Type and enter a title:
   Title: Reason for rejecting the request

36. Choose.
   The SAPscript editor is then displayed.

37. Enter your reason and choose . Choose .
   You then return to the dialog box for User confirmation of end of processing. The attachment is displayed under Objects and attachments.
38. Choose Complete work item.

You are now in your Business Workplace and the work item you have just processed has been removed from the worklist.

Procedure as requester

8. Choose Office → Workplace to start the Business Workplace.
9. Open the folder Inbox → Workflow.

This folder contains a work item awaiting a user decision. The symbol in the Attachment column indicates that the work item has an attachment.

10. Select the work item.

All of the attachments for this work item are displayed in the work item preview.

11. Choose the attachment in the work item preview.

The attachment is then opened and displayed in a new screen.

12. Execute the work item.

Further Procedure

Continue processing the workflow and analyze it using the workflow log.
Unit 20: Monitoring Missed Deadlines

Use

When defining steps that require dialog with the user, you can instruct the workflow system to monitor certain deadlines. The workflow system can monitor the following types of deadline:

- Requested start
- Latest start
- Requested end
- Latest end

This unit explains how missed deadlines are usually handled: The standard system response to an exceeded deadline is to notify a recipient of the message for missed deadlines [Extern] by sending a work item for missed deadlines [Extern] to his or her Business Workplace.

For further information, please refer to Maintaining the Deadline Tabs [Seite 1081] in the SAP Business Workflow documentation.

In this unit, the workflow is to be extended so that the requester should be notified if the superior has not made the decision about the leave request after 10 minutes.

Procedure

You start the Workflow Builder and open your workflow definition in Change mode. Proceed as described in Unit 7: Including "Check Notification of Absence" in the Workflow Definition [Seite 1609].

Determining the recipient of the message for missed deadlines

12. Open the step definition for the step Check notification of absence by double-clicking it.
13. Go to the Latest end tab.
14. Choose the entry Work item creation as the reference time.

   The active deadline is indicated by the symbol in the tab index of the relevant tab.
15. Enter the value 10 in the field + and choose Minutes as the time unit.
16. On the Display text tab, choose the entry Workflow initiator (expression) as the recipient.

   As a result, the entry \_WF_Initiator is automatically copied to the adjacent field and the entry in the checkbox changes to Expression.

   If a deadline is exceeded, the text visible in the display is automatically copied to the work item for missed deadlines. You will now change this text in the task definition.
17. Choose the highlighted text in the display to open the task definition.
18. Choose and go to the Description tab.
19. Choose the text type Text for latest end and then .

   The SAPscript text editor is then launched.
20. Enter the following text:

   Your notification of absence no. \_WI_OBJECT_ID.NUMBER\_ from \_WI_OBJECT_ID.CREATEDATE\ has not been approved yet.

   You should remind your superior.
If you want to insert an expression from the task container as a text variable, choose Include → Expression. Then choose the relevant expression in the Please choose an expression dialog box by double-clicking the Container node.


22. Choose 📥 and then 📥.

**Concluding step definition**

20. Choose 📥 to check the step definition.

21. Choose 📧 to exit the step definition.

22. Choose 📥.

The workflow definition is checked, saved, and, provided it does not contain any errors, activated. The system displays a message if problems are encountered during the test.

**Test**


All of the workflows that you are allowed to start are listed on the left-hand side of the screen. A description of the selected workflow is shown on the right.

17. Select your workflow and choose Start.

The work item for creating the notification of absence is proposed to you immediately for processing.

18. Fill out the form and choose 📧.

You have submitted your request and should now allow the deadline to expire. Once the latest end passes, a work item for missed deadlines appears in your Business Workplace notifying you of the missed deadline. This work item is a deadline monitoring work item, for which there is no executable method. You can go from the display of this work item to the work item display of the late work item.

**Notes on extensions**

The recipient of a message for missed deadlines is notified with a work item. The notification text is stored in the task definition of the monitored task. If you want to customize the notification, you can define your own Text for latest end on the Description tab in the task definition for the single-step task Check notification of absence. This text can contain text variables that are replaced at execution time.
Tutorial: Maintaining the Organizational Plan

Purpose
This tutorial shows you how to create the framework for an organizational plan quickly and effectively. An organizational plan describes how an employee is assigned within the organization. You can store the employees who are responsible for carrying out individual business activities in your organizational plan.
The organizational plan is maintained for specific clients in the Organizational Management component. An organizational plan created for HR purposes can also be used in SAP Business Workflow as long as the workflow functionality and the HR application are in the same client. Creating your organizational plan also creates the prerequisites for the Workflow System to determine the appropriate agents of a work item at runtime.

Process Flow
The individual units of this tutorial contain step-by-step instructions on how to create the basic framework of your organizational plan in order to reflect the structure and HR environment of your enterprise. The following items, each of which represents a separate unit, provide you with an overview of the tutorial procedure.

4. You create a root organizational unit. Once you have done so, you create the subordinate organizational units.

5. You create jobs, provided that they do not already exist in your job index. You also create positions, which represent the specific instances of jobs in your enterprise. You assign R/3 users as position holders. You can also indicate that a position is a chief position for an organizational unit.

6. You assign tasks to the objects you have created. These objects can be jobs, positions, or organizational units. You can use tasks to describe jobs and positions.

Examples are included to help you work through the individual units. These are intended to illustrate the step-by-step procedure for creating your organizational plan. For further information, see Organizational Management.
Unit 1: Creating an Organizational Structure

An organizational plan consists of the organizational units that exist in the company. The organizational units are related one to the other in a hierarchical reporting structure. However, they can also be created independently of each other.

To create a new organizational plan, you create a root organizational unit. This is the highest unit within the organizational structure, such as the Executive Board. You then set up the organizational structure, starting at the root organizational unit and working downwards.

Procedure

Creating a Root Organizational Unit


14. Confirm the validity period proposed in the dialog box Creating a Root Organizational Unit.

This takes you to the Create Organization and Staffing (Workflow) screen. This user interface is divided into four screen areas:

15. On the Basic Data tab in the details area, enter an abbreviation and a name in the Organizational unit input fields for your root organizational unit.

   Abbreviation: <ini_org>
   Name: <Organizational plan(initi)>


The root organizational unit you have just created is then selected in the overview area.
Creating further organizational units

Starting from the root organizational unit, you create subordinate organizational units. In this unit, you will create two further organizational units.

8. Choose \(\text{New Unit} \). A new organizational unit is now created below the root organizational unit.

9. Open the details view of the new organizational unit by double-clicking the corresponding line in the display area.

10. In the details area, enter an abbreviation and a description for the Organizational unit in the input fields.

   Abbreviation:  \(<\text{ini\_org\_sa}>\)

   Name:  \(<\text{Sales (ini)}>\)

When determining the validity period of objects and relationships, the system chooses the validity period of the superordinate object as standard.

If you want to change the validity period for the organizational units to be created and their relationships, change the values in the corresponding input fields in the details area.

11. Select your root organizational unit.

12. Choose \(\text{New Unit} \). A new organizational unit is now created below the root organizational unit.

13. In the details area, enter an abbreviation and a description for the Organizational unit in the input fields.

   Abbreviation:  \(<\text{ini\_org\_rd}>\)

   Name:  \(<\text{Research and development (ini)}>\)

14. Choose \(\text{New Unit} \). You can also include existing organizational units in your organizational plan. To do so, drag them from the selection area to your organizational plan.

Reassigning organizational units

If changes are made to the organizational structure of your company, you can reassign the organizational units involved.

If you reassign organizational units, please note the following:

- If you change the assignment of an organizational unit, you also change the associated relationship records.
- If the validity period of the new relationship overlaps with the validity period of the original relationship, the system delimits the validity period of the original relationship accordingly.
Root objects cannot be reassigned. In your example, the root object has the abbreviation ini Org.

3. Select the organizational unit that you want to reassign in the selection area.
4. Drag it to the organizational unit under which it is to be assigned.

**Delimiting organizational units**

You delimit organizational units if you want to change their validity period, that is, bring forward the validity end date.

⚠️

When you delimit objects (in this case, organizational units), all of the associated links are automatically delimited at the same time.

4. Double-click the organizational unit to open the details view.
5. Switch to display with periods. Choose 🗓️.
6. Change the entry in the Valid to field.

You can only change a period if you also change a different entry of the organizational unit at the same time. For example, change the description.
Unit 2: Creating Staff Assignments

The staff assignments for each organizational unit are maintained. The organizational unit is assigned positions in these staff assignments. A position is derived from a descriptive job and assigned to one or more users in your company.

All of the positions must be linked to jobs. The positions inherit the attributes and properties of the job.

In this unit, you will create the staff assignments for each of your organizational units. To do so, you will:

5. Create positions
6. Classify a position as a chief position.
7. Assign jobs to positions
8. Assign users to positions

Procedure

Choose Tools → Business Workflow → Development → Definition Tools → Organizational Management → Organizational Plan → Change. The Change Organization and Staffing (Workflow) screen is then displayed.

Create jobs

You must assign every position a job that contains general functions and tasks and which passes on the position. Jobs are normally defined centrally for an organizational plan in the job index. When positions are created, the corresponding job must only be assigned. For this tutorial, you create the positions yourself.

2. Choose Edit → Create job.

The dialog box Create jobs is displayed. The lower area contains a list of existing jobs and the upper area contains an input table in which you can create new jobs by entering abbreviations and names.

9. In the input table, enter an abbreviation and a name for each of the new jobs.

   Job - head of department:
   
   Abbreviation: <ini_hd_C>
   
   Name: <Head of department job (ini)>

   Job administrator:
   
   Abbreviation: <ini_ad_C>
   
   Name: <Administrator job (ini)>

   All new jobs receive a validity period from the current date to the 31.12.9999 as standard.

10. Choose ✅.
Opening staff assignments for an organizational unit

3. If the root organizational unit (ini_org) that you created in unit 1 is not open for processing, choose organizational unit in the search area, enter a search term and select your organizational unit from the organizational units found by double-clicking in the selection area.

4. In order to open the staff assignments for an organizational unit, select the organizational unit in the overview area and select the arrow on the right next to the Staff assignments (list).

   The staff assignments for the organizational unit selected are now displayed in the overview screen.

Create positions

You create positions for your organizational unit in the associated staff assignments.

9. Open the staff assignments for the organizational unit Research and development (ini).

   In this organizational unit, assign a position for an administrator and a position for a head of department.

10. Choose .

   The new position is now displayed in the staff assignments. You can edit all relevant data for the position in the details area.

   The details view for positions is available in the key date mode and the periods mode. You can toggle between the two modes with the and buttons.

11. In the key date mode, enter the abbreviation and name of the new position in the Position fields on the Basic data tab:

   Abbreviation: <ini_ad_S_rd>

   Name: <Administrator position: (ini) R&D>

   You can change the validity period of the position in periods mode.

12. Repeat steps 2 and 3. Use the following data:

   Abbreviation: <ini_hd_S_rd>

   Name: <Head of department job (ini) R&D>

13. Set the Head of own organizational unit indicator. Confirm the information using the adaptations of the relationship period.

   Chief positions [Extern] are marked in the staff assignments.

   You must define chief positions if you want to use roles [Extern] to determine recipients in workflows that determine a user’s superior.

14. Choose and then the arrow on the right next to and Organizational Structure.
The organizational unit **Research and Development (ini)** is then displayed.

15. Choose to switch to editing of your root organizational unit.
16. Repeat steps 1 - 7 for the organizational unit **sales (ini)**. Use the following abbreviations and names for the positions:

   **Abbreviation:** <ini_ad_S_sa>
   **Name:** <Administrator position: (ini) Sales>

   **Abbreviation:** <ini_hd_S_sa>
   **Name:** <Head of department job (ini) Sales>

   Set the **Head of own organizational unit** indicator for the head of department job sales.

### Assign jobs to positions

You assign a job to every position.

10. Display your created position in the selection area. Choose the entry **Position** in the search area and enter the abbreviation you used (ini) as the search term.

   All positions created in this unit are displayed in the selection area.

11. Select the **Administrator position (ini) R&D** for editing by double-clicking.

   The position and the organizational objects assigned to you (job, organizational unit, user) are displayed in the overview area.

12. Display your created job in the selection area. Choose the entry **job** in the search area and enter the abbreviation you used (ini) as the search term.

   All jobs created in this unit are displayed in the selection area.

13. Select the job **Administrator job (ini)** in the selection area.
14. Drag the job to the position in the overview area.
15. Choose .

   The job **administrator job (ini)** is assigned the position **administrator position (ini) R&D**.

16. Repeat steps 1 to 6 for the **head of department job (ini) R&D** and assign it the job **head of department job (ini)**.
17. Repeat steps 1 to 6 for the **head of department job (ini) Sales** and assign it the job **head of department job (ini)**.
18. Repeat steps 1 to 6 for the **administrator position (ini) sales** and assign it the job **administrator job (ini)**.

### Assigning users to positions

When you assign users to positions, you define the **holders [Extern]** of these positions.

This assignment is necessary for workflow applications if you use roles or objects in the Organizational Management to determine recipients.
If you use Personnel Administration (PA), the system recognizes employees who are assigned directly to positions. To ensure in this tutorial that the R/3 user can be established as the agent starting from the employee, employees must have a relationship to R/3 users. If you do not use Personnel Administration, users are assigned directly to positions. At workflow runtime, they are established directly as the agents of particular single-step tasks. You can assign the user to the position in the staff assignments or in the single processing of a position. In this tutorial you assign the users using the single processing of a position.

7. Display your created position in the selection area. Choose the entry Position in the search area and enter the abbreviation you used (ini) as the search term.

   All positions created in this unit are displayed in the selection area.

8. Select the position in the selection area for which you want to assign a user by double-clicking.

   The position and the organizational objects assigned to you (job, organizational unit, user) are displayed in the overview area.

9. Display the user assigned to in the selection area. Choose the entry user in the search area and enter an appropriate search term.

   All of the users that match your search criteria are displayed.

10. Select the user whom you want to assign in the selection area.

11. Drag the user to the position in the overview area.

   Users can occupy a position either in full or in part. This depends on the working hours assigned to the position, and on the working capacity of the person or user.

   The staffing percentage refers to the working capacity of a person or user assigned to a position.

12. Choose .
Tutorial: Event Creation During Status Changes

Purpose
This tutorial covers the following topics:

- **Flexible event creation during status changes**
  You can generate an event for a status change in order to respond to a certain business circumstance.

- **Send pre-formulated documents as mail**
  You will learn how to send a text as a message (for example, with a note) to a receiver as a workflow step.

- **Attributes with object reference and multi-level expressions**
  The topics are linked to an integrated scenario whose creation you can follow step by step in individual units.
  The tutorial is designed for customers and consultants who want to extend the scenarios provided by SAP. The explanations and working methods are consciously covered in great depth; the fact that existing knowledge is repeated has been taken into account in this case.

  It is of central importance to the scenario (and this tutorial) that the changes you make are logged with a system or user status in an application component. You can only transfer what you learn in this tutorial to other scenarios if you make this setting.

Process Flow
In **Unit 1: Testing the Existing Application Functionality [Seite 1649]**, you will release a production order in the R/3 System.

**Identifying and modifying object types**
The object type that is important with respect to the scenario must be identified. This is the object type **BUS2005 (Production order)**.

  This object type is supplied as standard in the **Business Object Repository**.

You should use this object type in the manner for which it was intended. For this scenario, however, you have to extend it to include the released event [Extern].

In **Unit 2: Identifying and Extending Object Types [Seite 1651]**, you will extend the definition of the object type.

**Creating an event**
Setting the status **REL** for the status object type **ORH** should be publicized with the event released and therefore made known across the entire system. You must make sure that the event is generated.

The creation of this event is a basic requirement for executing the workflow.
In **Unit 4: Flexible Event Creation [Seite 1657]**, you will link the creation of events to the setting of a specific status.
Agent determination
The recipients of the mail are all employees in an organizational unit. This organizational unit is only created to group together the recipients or agents. It thus takes on the characteristics of a distribution list. In Unit 5: Finding Agents - Organizational Plan [Seite 1659], you will create the organizational unit and assign users to it.

Defining a workflow
In Unit 6: Defining a Workflow [Seite 1661] you will create a workflow for which the event released is defined as the triggering event. In order to do so, you define a binding from the event container to the workflow container that provides the reference to the released production order.

Send text as mail
In Unit 7: Editing Workflow Definitions [Seite 1663] you will define a task with which an e-mail is automatically sent. To define this task, you will use a wizard.
Unit 1: Testing the Existing Application Functionality

Use
This scenario is based on the existing application functionality of the PP component (Production Planning and Control).

Production orders are an essential part of the Production Planning System. They are used to control and monitor production within a plant, as well as being a controlling instrument within cost accounting.

The internal activities are handled via production orders. A production order determines which material should be produced when, where, and how. It also determines which resources must be used and how the order costs are to be offset.

When you release a production order for production, the system status REL (released) is set for the status object type ORH (PP,PM: Order header) (=activated).

Procedure
The following steps can be executed as described above within the standard functionality of the application component PP in a test or demonstration system (for example, IDES [Extern]).

Using the proposed input values, you can follow this example in the IDES system!

Create production order
A production order that is created with reference to an available production order is used as the basis for the scenario.

Choose Logistics → Production → Production control → Order → Create → With material.

The Production Order Create: Initial Screen is then displayed.

Choose any order in the Order field in the Copy from box. Use the F4 input help.

Choose ☑.

The Production Order Create: Header screen is then displayed.

For Basic Dates Finish, enter a date that is a few weeks in the future.

Choose ☑ and note the number displayed. The number is assigned automatically.

The Production Order Create: Initial Screen is then displayed.

Display status information for production order
Choose Order → Display and enter the number of the production order you created in the Order field.

Click the Display overview flag and choose ☑.

The Production Order Display: Header screen then appears.

Note the entry in the Plant field.

You will need this entry to release the order later on in this unit.

Choose ☑.

A list of the system statuses that are set is then displayed. The status REL (released) is not yet set.
However, the release is part of the business transactions that are allowed as the next step. An overview of the possible and permitted order status is shown on the Business Processes tab.

Choose Extras \(\rightarrow\) Overview.

The Display Status: Overview dialog box then appears. From the entries in the displayed popup, you can see that the statuses refer to the status object type ORH (production order). This information is important for creating events.

**Release production order**

Choose Logistics \(\rightarrow\) Production \(\rightarrow\) Production control \(\rightarrow\) Order \(\rightarrow\) Release.

Enter the plant in the Plant field and the order number in the Order field.

Delete the entries in the date fields.

Choose \(\mathbb{Q}\).

The Release Production Orders: List screen is then displayed.

Select the order and choose \(\mathbb{P}\) Order.

In the Status column, you see that the status REL is set.
Unit 2: Identifying and Extending Object Types

Object Type BUS2005 (Production Order)
Execute the following steps, in order to learn about the object type BUS2005


Enter BUS2005 in the Object/interface type field and choose .

The screen Display object type BUS2005 is then displayed.

If the folder symbol in front of one of the entries Interfaces, Key fields, Attributes, Methods or Events contains a plus sign, elements are available and defined for this object type component. Open the folders to display an overview of the existing elements.

Interface
In addition to the IFSAP interface, which is supported by every object type, this object type supports three further interfaces.

Key fields

Double-click the entry ProductionOrder.Number to display detailed information on this key field.

The key field [Extern] of the object type BUS2005 is Number (Order number) and refers to the table field AUFK-AUFNR. The key field of an object of the type BUS2005 identifies it uniquely and enables read access to its attributes.

Attributes
Attributes [Extern] are defined for the object type BUS2005.
The two database field attributes LastChangedBy (name of person who made last change) and ChangeDate (date of last change) are used later on, since they are to be included in the request text (along with the key field).

Methods

The Display method is used in this scenario.

Enhancements to the Object Type BUS2005

The definition of the object type BUS2005 is incomplete and cannot be used in this scenario. In other words, this object type must be extended.

You need an event [Extern] to publish the change to the material master data throughout the system. All events which are to be used must be defined beforehand as elements in their object type.

Create Subtype for Available Object Type

Since you cannot make any direct changes to the SAP object type BUS2005, first create a customer-specific object type as a sub-type [Extern] of this object type. This sub-type inherits all attributes and methods of its super type.

Choose .

You are now in the Business Object Builder: Initial Screen. The object type BUS2005 is still entered.

Choose Subtype.

The dialog box Create object type appears.
Enter the following data:

Object type:  \textit{Zini}_2005 \\
Object name:  \textit{<Production order>} \\
Name:  \textit{<Production order>} \\
Description:  \textit{<Production order extension BUS2005>} \\
Program:  \textit{Zini}_2005 \\
Application:  \textit{Z}

Choose \textbullet{} and save the subtype as a local object.

You are now on the \textit{Change Object Type \textit{Zini}_2005} screen and can edit the new object type you created.

Ensure that this object type has inherited all methods and attributes from the super-type \textit{BUS2005}. Inherited elements are marked in red.

\textbf{Creating an Event}

The status change should be broadcast with an event with the name \textit{released1}. This event can only be generated if it is defined for the object type. This is not yet the case.

Position the cursor on the \textit{Events} node and choose \textbullet{}.

The \textit{Change Object Type \textit{Zini}_2005} dialog box is then displayed.

Enter the following data:

\begin{itemize}
  \item \textit{Event}: \textit{released1} \\
  \item \textit{Name}:  \textit{<Order released>} \\
  \item \textit{Description}:  \textit{<Production order released>}
\end{itemize}

Choose \textbullet{}.

Position the cursor on the event \textit{released1} and choose \textit{Edit} \rightarrow \textit{Change release status} \rightarrow \textit{Object type component} \rightarrow \textit{To implemented}.

Until now, within the object type definition, you have only described the fact that the event \textit{released1} is intended to be used with object type \textit{Zini}_2005. You must ensure that the event is actually created. This is described in one of the following chapters.

Choose \textbullet{} to check the object type.

Choose \textbullet{}.

\textbf{Implementing and generating an object type}

You are on the screen \textit{Business Object Builder: Initial Screen}. You object type is entered in the \textit{Object/interface type} field.

Choose \textbullet{} \rightarrow \textit{Change status to} \rightarrow \textit{Implemented}.

Choose \textbullet{} to generate the object type.
**Delegation**

In order to be able to work through this tutorial more than once or with a number of different users at the same time and on the same system, edit the new object type `Zini_2005`, which you created as a sub-type of the object type `BUS2005`. This procedure is particularly appropriate for training situations.

For extensions to object types and their use in productive workflow scenarios, SAP recommends an extended procedure in which you define the sub-type as the delegation type `[Extern]` of the object type. For further information, please refer to Extending and Adapting Object Types [Seite 1157].

Delegation functionality is not used in this tutorial.

**Result**

In this unit, you were introduced to the *Business Object Builder* and the object type `BUS2005`. For the object type `BUS2005`, you have created a subtype `Zini_2005` and extended this by adding an event.
Unit 3: Adding Further Attributes

Use
In this unit, you will add further attributes that can be used in a workflow definition to the object type.

Procedure

"Last Changed by" as an Object Reference

The inherited attribute `LastChangedBy (name of person who made last change)` of the object type `Zini_2005` returns the contents of the database field `AUFK-AENAM`. This is the user name of the person who made the last change.

You can also use the name of the user from the user defaults in task descriptions or work item texts. To do so, you create a new attribute for your object type.

Instead of the user ID `MILLERP`, which is returned as the value of the attribute `LastChangedBy`, the name `Peter Miller` from is be used from the user defaults in the texts.

The name of the user is an attribute of the object type `USR01 (SAP User)`. You must create an attribute for the object type `Zini_2005 (production order)` with a data type reference to the object type `USR01 (SAP user).

1. Open your object type to edit it in the Business Object Builder.

   The Change Object Type `Zini_2005` screen is then displayed.

2. Position the cursor on the Attributes entry and choose `1`

   The query Create with ABAP Dictionary field proposals? appears.

3. Choose `No`.

4. Enter the following data:

   Attribute: `ChangedBy`
   Name: `Changed by (object)`
   Description: `Last changed by (object reference)`
   Reference table: `AUFK`
   Reference field: `AENAM`
   Object type: `USR01`

1. Select Database field in the Source frame.
2. Select Object type in the Data type reference frame.
3. Choose `2`.
As this is a database field attribute (source database field) no implementation is carried out for this attribute.

Status of an Object Status as an Attribute

If the status (set or not set) of an object is to be available as an attribute of an application object, you must add the interface IFSTATUS (status management) and implement the attributes StatusObjNumber and StatusObjTyp that it contains. Then add the ReleaseStatus attribute to your object type:

1. Position the cursor on the Interfaces entry and choose .
   
   The Enter Interface Type dialog box appears.

2. Enter the IFSTATUS interface in the input field and choose .
   
   This interface provides the StatusObjNumber and StatusObjTyp attributes.

3. Position the cursor on the StatusObjType attribute (object type of status management) in the list.

4. Choose .
   
   The entry changes color. You can now edit the attribute.

5. Position the cursor on the StatusObjType attribute and choose Program.
   
   The dialog box “Do you want to have a template generated automatically for the missing section?” is displayed.

6. Select Yes.
   
   The source code, which is generated automatically, is displayed.

7. Make the following changes to the source code:

   ```
   GET_PROPERTY STATUSOBJTYPE CHANGING CONTAINER.
   OBJECT-STATUSOBJTYPE = 'ORH'.
   SWC_SET_ELEMENT CONTAINER 'StatusObjType' OBJECT-STATUSOBJTYPE.
   END_PROPERTY.
   ```

8. Choose , and then .

9. Position the cursor on the StatusObjNumber attribute (object number of the status management).

10. Choose .
   
    The entry changes color. You can now edit the attribute.

11. Position the cursor on the attribute StatusObjNumber and choose Program.
    
    The dialog box “Do you want to have a template generated automatically for the missing section?” is displayed.

12. Select Yes.
    
    The source code, which is generated automatically, is displayed.

13. Make the following changes to the source code:

    ```
    GET_PROPERTY STATUSOBJNUMBER CHANGING CONTAINER.
    OBJECT-STATUSOBJNUMBER = 'OR'.
    ```
Unit 3: Adding Further Attributes

SWC_GET_PROPERTY SELF 'Number' OBJECT-STATUSOBJNUMBER+2.
SWC_SET_ELEMENT CONTAINER 'StatusObjNumber'
                     OBJECT-STATUSOBJNUMBER.
END_PROPERTY.

14. Choose ☑, ☑ and then ☑.
15. Position the cursor on the Attributes entry and choose ☑.

The query Create with ABAP Dictionary field proposals? appears.

16. Choose No.
17. Enter the following data:
   Attribute: ReleaseStatus
   Name: Object status
   Description: Status of an object status
   Object status: I0002
18. Select Object status in the Source frame.
19. Choose ☑.

The system enters SWCEDITOR as the reference table and OBJSTATUS as the reference field.

20. Choose ☑.
As this is an object status attribute, no implementation is required for this attribute.

Generate and Test Object Type

1. Choose Edit → Change release status → Object type → To implemented.
2. Choose ☑ to generate the object type.
3. Position the cursor on the Object type entry and choose ☑.
4. Chose FIND ☑ in the line. Enter the number of a production order.

You can enter the number of the production order that you used in Unit 1.
The list displays the values of all attributes for this object type. For the attributes that are created with a data type reference to an object type, you can navigate further to the attributes of the referenced object type.
Unit 4: Flexible Event Creation

Use
You want to establish a relationship between the status change of an object (here: release of a production order) and the subsequent reaction (here: start a workflow). Before you define the workflow, you must make sure that an event is created if the status REL is set for the status object type ORH. However, you must first ensure that an event is really created when the status REL is set for the object type ORH, that is, when the status change has taken effect on the object.

Prerequisites
You have defined the released1 event as an enhancement of the object type Zini_2005. This object type was, in turn, created as a sub-type of the SAP object type BUS2005, as it is otherwise not possible to make changes to object types delivered by SAP. This is the precondition for generating this event and using it as a triggering event.

Procedure
Event Creation During Status Change
The event released1 should be generated if the status REL is set.


   The Event creation status management dialog box appears.

2. Choose Customer settings.

   The View “Display Events for Status/User Status”: Initial Screen is displayed.

3. Choose to switch to change mode.


5. Enter the following data:
   
   StatusOT: ORH

   BusinessOT: Zini_2005

   Event: released1

6. Choose .

7. Select your entry and double-click the Status restrictions node to go to the associated view.

8. Choose New entries.

9. Enter I0002 in the System status field.

   The entry I0002 is the internal representation of the status REL. You can also use the F4 possible entries pushbutton to select the status REL.

10. Choose and exit status management.
Test the Event Creation

Test whether the events are created successfully.

1. Check whether the event log is activated. Choose Tools → Business Workflow → Development → Utilities → Events → Event Trace → Switch Event Trace On/Off and switch on the event trace.

2. Create a production order and release it. The relevant procedure is described in Unit 1: Testing the Existing Application Functionality [Seite 1649].


   The screen Display Event Trace is displayed.

4. Restrict the selection by entering values in the Creation date and Creation time fields so that the event you created appears in the display.

   The list contains the object type you created with the created event. No receiver is entered here.
Unit 5: Finding Agents - Organizational Structure

Use

Certain employees are notified by mail as a response to the release of the production order. In addition to the existing organizational plan of your enterprise, all employees who are to receive the mail are to be grouped together in a separate, isolated organizational unit. The organizational unit is used as a distribution list in this scenario. The organizational unit is assigned existing positions, which are also used in the organizational plan of the enterprise. As a result, the recipient group is determined dynamically from the holders of the positions.

Procedure

You create an organizational unit and assign it positions. These positions are assigned users.

Creating an organizational unit


18. Confirm the validity period proposed in the dialog box Creating a Root Organizational Unit.

This takes you to the Create Organization and Staffing (Workflow) screen. This user interface is divided into four screen areas:

19. On the Basic Data tab in the details area, enter an abbreviation and a name in the Organizational unit input fields.

   Abbreviation:  Zini_OrderRel

   Name:  Release production order
1. Choose 🔄.

**Assigning positions**
In the staff assignments, you assign the organizational unit positions that are also used in the organizational plan of your enterprise.

1. Change to the overview area in the staff assignments [Exterm] of the organizational unit, in order to assign positions, jobs and holders. Choose the arrow 🔄 on the right next to the 🔄 and then the staff assignments (list).

   The staff assignments of the new organizational unit are displayed.

1. Select positions in the find area and enter a search range.

   The positions that were found are shown in the selection area.

2. Select the position that you want to copy to the staff assignments and drag it to the staff assignments table in the overview area. Repeat this procedure until all of the necessary positions have been assigned to the organizational unit.

   Note that you also select a position, that you are assigned as a user, so that you can check the successful execution of the scenario.

3. Choose 🔄 and exit the screen for editing the organizational plan.
Unit 6: Defining a Workflow

Use
A mail is to be sent once the production order has been released. This is carried out by a workflow which you will define in this unit.

Procedure

   The Workflow Builder screen is then displayed. If you are starting the Workflow Builder for the first time, an initial workflow definition is displayed there. Otherwise, the workflow that was processed last is opened.

2. Choose .

   The Workflow Builder - Create 'Undefined' [new, not saved] screen is then displayed. An initial workflow definition is displayed.

Create container element in the workflow container
You must now add a container element to the workflow container that can store an object reference to a production order.
1. Choose <Double-click to create> by double-clicking in the Workflow Container.

   The dialog box for creating a new container element is displayed.

2. Enter the following data:

   Element: ProductionOrder
   Name: Production order
   Short Description: Production order
   Object type: Zini_2005

3. Set the Import and Mandatory flags.
4. Select Object type as the Data type reference.
5. Choose ✓.

   The new container element is displayed in the workflow container. The symbol shows that the new container element is an import parameter of the workflow and thus belongs to the workflow interface.

Saving the workflow
1. Choose .

   The first time a workflow is saved, a multistep task of the type workflow template is automatically created.

2. Enter the following data:

   Abbreviation: Zini_Mail
3. Choose Local Object.

**Determine Triggering Event**

If a workflow is to be started by an event, you must define this as a triggering event. In this scenario, the workflow is to be started as a response to the released1 event of your object type Zini_2005. You must activate the event linkage [Extern] for this event and define a binding.

1. Choose 
   The basic data of the workflow is displayed.
2. Go to the Start tab.
3. Enter the following data:
   - **Object type**: Zini_2005
   - **Event**: released1

**Activate event linkage**

4. Click on in the event column.
   Active event linkages are marked with .

**Defining the binding between the event container and the workflow container**

The event parameter _Evt_Object of the event released1 contains the object reference to the released sales order.

The event parameter _Evt_Creator of the event released1 contains the name of the person whose system user was used to generate the event. The name is stored in the form US<Name> in this element. This information must be transferred by means of a binding between the event container and the workflow container.

The workflow container contains the production order (technical name: ProductionOrder) element you created as well as the standard initiator element (technical name: _WF_Initiator).

5. Choose released1 in the event line.
   The Task: Binding for Triggering Events dialog box is then displayed. The workflow system has automatically assigned the contain element _Evt_Object of the event container to the ProductionOrder element of the workflow container.

6. Choose to display all of the container elements in the workflow container to which you can define a binding.

7. Assign the _Evt_Creator container element in the event container to the _WF_Initiator element in the workflow container. Use the F4 input help.

8. Choose .

9. Exit the basic data of the workflow and choose .
Unit 7: Editing Workflow Definitions

Use
You define a mail step in this unit, which when executed, sends a mail to the members of the newly created organizational unit. For this purpose, use the step type Send mail.

Procedure
1. If you are no longer in the Workflow Builder, choose Tools → Business Workflow → Development → Definition Tools → Workflow Builder. The Workflow Builder is then launched with the workflow that was processed last. If the required workflow is not opened, choose 🕵️ and enter the abbreviation of your workflow. You can use the F4 input help and enter an abbreviation to search for your workflow. Choose ✅.

As well as the initial workflow definition, your workflow also contains the defined triggering event [Extern]. If you are not in change mode, choose 🕵️.

2. Choose Send mail in the step type area and click on the Undefined step.

All necessary entries for sending a mail are made on the tab page Mail.

3. Specify the person(s) to whom the mail is to be sent. Do not change the presetting recipient type organizational object. Choose the entry organizational unit in the area Recipient and enter the identification of your organizational unit. You can use the F4 input help.

   Enter the subject of the mail.

   Production order released

3. Enter the text for the mail.

   This text should contain text variables that can be replaced at runtime by values taken from the task container. Text variables are surrounded by a & symbol. To insert a text variable, choose 🕵️. The system fields and the container element of the workflow container are displayed in the new dialog box. Double-click the expression. Choose 🕵️ to toggle between the technical name and the description. When you select a container element, the system creates the necessary container element in the task container and defines a binding definition from the workflow to the workflow container.

   General information:
   The production order &ProductionOrder.Number& was released by &ProductionOrder.ChangedBy.Name& on &ProductionOrder.ChangeDate&.

4. Set the Send express flag.

4. Choose ✅.

The system automatically creates a task that the sending of the mail executes. You must enter another abbreviation and name for the task:

   Abbreviation: Zini_Mail
Name: Mail: Production order released

In the Create Object Directory Entry dialog box, choose Local object.

The graphical display is displayed again.

Concluding step definition

Choose ✱.

The workflow definition is checked, saved, and, provided it does not contain any errors, activated. The system displays a message if problems are encountered during the test.

To view the task created by the mail step, carry out the following steps:

1. Open the step definition for the step Production order released by double-clicking it.
2. On the tab page Control, double-click the task ID to open the task definition.

The screen Standard task: Display is displayed. If you want to test the task (see below), make a note of the task number assigned by the system.

Test the defined task

You can also test the task you defined by starting it in dialog.

2. In the field Task, enter the number of the standard task you defined in the form TS<8-digit number>.

You can also use the F4 input help pushbutton.

3. Choose Input data to fill the ProductionOrder element in the task container with a value for the production order before execution.
4. In the Initialize Container dialog box, assign the number of the production order you created to the ProductionOrder container element. To do so, use the F4 input help. Do not assign a value to the _WI_Object_ID container element.
5. Choose ✱ to display all the container elements.
6. Assign your user name as the email recipient to the ADDRESSSTRINGS container element of the task container.
7. Assign an X to the Express container element.
8. Choose ✱.
9. Choose ✱ to start the test.

The mail is sent in the background. Open your Business Workplace to check that the mail has arrived.

Further notes on sending mails

You have followed the above procedure to identify the recipient of the mail by a user name and assigned the task container to the ADDRESSSTRINGS container element. This container element is identified as a multiline element and can also store a list of mail recipients.

To assign more than one element to the ADDRESSSTRINGS container element in the Container Editor, position the cursor in the ADDRESSSTRINGS line and choose ✱.
The ADDRESSSTRINGS container element can contain other values in addition to a user name. For further information on entry types, see Address Type [Extern]. You assign the recipient type to the TypeID container element of the task container.
Unit 8: Executing the Scenario

Use
In this unit you will execute the scenario created in this tutorial.

Prerequisites
You have completed all the units. You have:
- Linked the creation of an event to a status change
- Declared the event as a triggering event for the workflow
- Defined a workflow that executes the step "send mail"

Procedure

Executing the scenario
Release a production order. Follow the description in Unit 1: Testing the Existing Application Functionality (Seite 1649).

Business Workplace
When a production order is released, an event is triggered that starts your workflow. The express mail Production order <number> released is sent to the Business Workplace inboxes of the recipients.

1. Log on to the system under a user name contained in the list of recipients.
2. Choose Office → Workplace to start the Business Workplace.
3. Check whether the express mail is in your inbox.
Tutorial: Workflow Programming

Purpose
This tutorial provides a step-by-step introduction to defining and implementing object types in the Business Object Repository (BOR). You should carry out these steps in the system yourself. The tutorial is intended for both customers and consultants who want to modify the scenarios provided by SAP. It is also designed for developers within SAP who want to connect their application component to the workflow functionality. This tutorial is not designed to replace the documentation on the Business Object Repository, which you can consult for further details. For further information, refer to Business Object Repository [Seite 1100] in the SAP Business Workflow documentation.

Prerequisites
To be able to work through this tutorial, you must be familiar with the basic ABAP programming principles. You should have initial practical experience using SAP Business Workflow and have worked through the Tutorial: Workflow Modeling [Seite 1583].

Process Flow
The individual sections of this tutorial provide a step-by-step introduction to working with the Business Object Builder, creating, maintaining and implementing object types, and also point out the aspects that require consideration. These steps are explained using the example of the object type definition sales order. If you do not yet have any experience of object types, carry out these examples directly in the system. In particular, use the options given to test “your” object type to see whether your work has been successful.

If you want to maintain your own object types in the Business Object Repository, go through this tutorial using the individual sections as a type of “check list”. If you use this tutorial to create your own object types, knowledge of the application environment of the object type to be maintained is absolutely essential. In particular, you should know in which tables the data (attributes) of the object type is stored and which transactions or function modules are used to call the main operations (methods) for objects of this type.

Tutorial Structure
The tutorial assumes that no object type sales order exists in the system or is to be maintained in the Business Object Repository. (However, this object type does already exist. Do not let this irritate you.) Read Unit 1: Creating Object Types in the Business Object Repository [Seite 1671] for information on how to create object types.

In the following units of this tutorial, you extend this object type by adding selected attributes, methods and events:

<table>
<thead>
<tr>
<th>Attributes</th>
<th>Methods</th>
</tr>
</thead>
<tbody>
<tr>
<td>DocumentDate</td>
<td>Unit 7: Creating the Method Edit Without Parameters [Seite 1686]</td>
</tr>
<tr>
<td>SoldToParty</td>
<td></td>
</tr>
<tr>
<td>TextWithDocumentNo</td>
<td>Unit 4: Creating Virtual Attributes [Seite 1679]</td>
</tr>
<tr>
<td>SalesGroup</td>
<td>Unit 5: Creating Object Virtual Attributes [Seite 1681]</td>
</tr>
<tr>
<td>Items</td>
<td>Unit 6: Creating Multiline Virtual Attributes [Seite 1683]</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Event</th>
<th>Unit Number</th>
</tr>
</thead>
<tbody>
<tr>
<td>Edit</td>
<td>Unit 7: Creating the Method Edit Without Parameters [Seite 1686]</td>
</tr>
<tr>
<td>ExistenceCheck</td>
<td>Unit 8: Creating the ExistenceCheck Without Parameters [Seite 1689]</td>
</tr>
<tr>
<td>Display</td>
<td>Unit 9: Creating the Display Method Without Parameters [Seite 1689]</td>
</tr>
</tbody>
</table>
In production operation, you will often encounter situations in which the object type to be maintained already exists in the Business Object Repository and is (only) to be extended, because you think customer-specific extensions or adaptations are required in particular areas.

For situations of this kind, refer to Extending Object Types: Inheritance and Delegation [Seite 1669] to discover how to derive a "new" object type from an existing object type as a subtype, which can then be extended. Creating an object type as a sub-type "saves" you maintaining the basic data and creating key fields. The further maintenance of this object type (creating methods, attributes, and events) is exactly the same as for maintaining a completely new object type.

**Implementation program for example**

Every object type defined in the Business Object Repository has one implementation program [Extern] containing the implementation of the key fields, methods and attributes of this object type. The implementation program of the object type created in this example is listed in the Appendix: Implementation Program for Object Type Z_BUS2032 [Seite 1700] of this tutorial. Reference is made in the individual units of this tutorial to the appropriate lines of this program.
Extending Object Types: Inheritance and Delegation

Use

In particular situations you will need to modify an object type in conjunction with delegation rather than create a new object type. This applies whenever you want to add components to an object type which are not provided in the standard version and at the same time need to ensure that productive scenarios with the original SAP object type remain operational. If no modifications are required, you can use any released object type provided by SAP as is.

Conventions for the other sections of the tutorial

The procedure described below consisting of “create subtype” and “delegation” is always required when you have to extend an SAP object type to meet your requirements. However, within this tutorial it is always assumed that the object type to be maintained did not exist in the Business Object Repository and therefore is created from scratch. This section therefore informs you about the procedure involved in certain extensions, but does not really belong to the tutorial.

Procedure

Creating a sub-type

Create a new object type as the sub-type of the object type you want to extend. This subtype automatically inherits all the components (methods, attributes, and events) of the original object type including its implementation.


2. In the field Object type, enter the object type you want to extend. This is then the supertype in the inheritance hierarchy.

3. Choose Subtype.

4. For the new object type, enter:
   - A unique name that can be used to identify it
   - An object name
   - A name
   - A short description

5. Enter the name of the implementation program of the object type and assign the code letter of your application.

6. Confirm your entries with ✔ and save the object type created as a local object or with development class in a transport request.

Entering a delegation type

Before you process “your” object type any further, define it as the delegation type [Extern] of the supertype.

1. Exit maintenance of the object type and go back to the Business Object Builder. Choose Settings → Delegation → System-wide.

2. Add a new entry to the table. To do this, choose Table view → Display → Change, then Edit → New entries.
3. Enter the name of the object type (super type) for which you want to specify a delegation type.

4. Enter the name of the sub-type as the delegation type.

5. Choose \[button\].

The delegation entered is client-independent.

In all SAP Business Workflow definition tools, you must still use the “old” object type (supertype). However, when the definition is read and evaluated (at runtime, for F4 input help, and so on), the definition of the delegation type (subtype) is used.
Unit 1: Creating an Object Type in the Business Object Repository

Use

In this unit, you will learn how to create the object type sales order and how to maintain the key fields of this object type. You will see that the system supports you in the selection of key fields and the implementation in the program.

Procedure

Creating an Object Type

2. Enter the name Z_BUS2032 of your new object type in the Object type field.
3. Choose Create.
   The dialog box Create object type is displayed.
4. Enter the following texts:
   - Object name: Sales order
   - Name: Sales order (mod.)
   - Short description: Sales order (modification for tutorial)
5. Enter Z_BUS2032 as the program name. This is the name of the implementation program of the object type.
6. Enter Z as the code letter of your Application.
7. Confirm your entries with .
8. Save the object type you just created as a local object or with development class in a transport request.

"Analyzing" the object type definition

First take a look at the "initial" object type definition. To do this, expand the hierarchies under the entries Interfaces, Attributes and Methods by clicking on the plus sign.

Interface IFSAP (SAP standard interface)

You see that your new object type already supports the interface IFSAP (SAP standard interface). Every object type supports this interface as standard.

The attribute and the two methods, which are also already in your object type, are defined for the interface IFSAP and are inherited from it. (The attribute and methods are therefore only there because the interface IFSAP is supported. If you were to delete the interface IFSAP, the attribute and the two methods would also "disappear" again.) The following object type components are inherited from the interface:

- Method ExistenceCheck (check existence of an object)
  Method without dialog, which is called to check whether the object exists.
Unit 1: Creating an Object Type in the Business Object Repository

- **Method** Display (*Display*)
  
  Method that displays the object in an object-specific way.

- **Attribute** ObjectType (*Object type*)
  
  Attribute containing the type of the object.

The methods Display and ExistenceCheck still have to be implemented in the implementation program on an object-specific basis. This can also be seen by the background color. You will also do this in a later unit.

The implementation of the attribute ObjectType does not need to be changed.

### Maintaining Key Fields

The next thing you have to do is create the key fields [Extern] of your new object type, so that the system knows how to uniquely identify objects of this type.

1. Position the cursor on the entry **Key fields**.

2. Choose ☑️ and answer the query *Create with ABAP Dictionary field proposals?* with Yes. This simplifies the entry of the data type reference.

   The dialog box *Create with Data Dictionary field proposals* is displayed.

3. Enter the table VBAK.

   The table VBAK (*Sales Document: Header Data*) contains the data for a sales order. The only key field of this table, identifying a sales order, is the field VBELN (*sales document*).

4. Choose ☑️.

   The only key field VBELN of this table is now displayed.

5. Select this field. (In the case of object types that have several key fields, you should of course select all the key fields.)

6. Choose ☑️.

   You go to the dialog box *Create* with text proposals for the key field to be created.

7. Check the proposed texts. Changes do not have to be made (but they could be made).

8. Choose ☑️. This is final confirmation that you want to transfer this key field into the object type definition.

   The key field is created with the automatically-proposed name SalesDocument for the object type Z_BUS2032.

9. Choose ☑️.

### Adding Interfaces

In addition to the interface IFSAP (*SAP standard interface*), which is automatically supported by the newly created object type, you have the option of assigning the object type additional interfaces with methods, attributes and events, which this object type must then support.

1. Position the cursor on the entry **Interfaces**.

2. Choose ☑️.

   The *Insert Interface Type* dialog box is then displayed.
3. Enter **IFCREATE** as the Interface type and choose ✔️.

4. Choose ✔️.

5. Repeat this procedure for the interface types **IFEDIT** and **IFFIND**.

You have now added three new interface types to the object type Z_BUS2032.

- **Create** interface (**IFCREATE**)
  
  This interface adds the method **Create** to your object type.

  Every object type that provides a function for creating should support the interface **Create**. Using the interface ensures that names are assigned in a standard manner, and makes it possible to set up a generic search help.

- **Edit** interface (**IFEDIT**)

  This interface adds the method **Edit** (change) to your object type.

  Every object type that provides a function for changing should support the interface **Create**. Using the interface ensures that names are assigned in a standard manner, and makes it possible to set up a generic search help.

- **Find** interface (**IFFIND**)

  This interface adds the method **Find** to your object type.

  This method determines an object which is available with this type and returns a reference to the object found.

  Every object type should support the interface **Find**.

The implementation of the attributes and methods inherited from the interface is usually incomplete and not adapted to the current conditions of the object type. This will be discussed in a later unit.

**Implementing object type**

At this point go to the implementation program of the object type you have created.

Choose Program.

Since methods and attributes have not yet been implemented, the implementation program only contains the data declaration for the object key. This section is generated automatically from the information you have specified for the key fields of the object type.

Do not change this part of the implementation program!

**Key fields in the implementation program**

Analyze the implementation program of your object type as it stands. You can use the implementation program in the appendix [Seite 1700] as a comparison.

Note that the implementation program in the appendix already contains the complete implementation of object type **Z_BUS2032**.

Some of the data declarations and program components there have not yet been included in your example, and are not required or created until the examples in the units still to come.

The key fields of the object type are declared in the program between the two macro commands **BEGIN_DATA OBJECT** and **END_DATA OBJECT**.
These macro commands, together with the commands in the implementation program, declare a structure for all key fields, which always starts with `OBJECT-KEY-` and whose fields are derived from the defined key fields of the object type.

The (only) key field of the object is hence available in the variable `OBJECT-KEY-SALESDOCUMENT` in the program.
Unit 2: Creating Database Field Attributes

Use

In this unit you create and implement the database field attribute [Extern] document date. This attribute contains the document date as a field value as it is stored on the database. The system helps you implement this attribute by generating source text automatically.

Procedure

1. Position the cursor on the entry Attributes.
2. Choose and answer the query Create with ABAP Dictionary field proposals? with Yes. This simplifies the entry of the data type reference.
3. Enter VBAK in the Table field. All database field attributes of an object type refer either to the table used to define its key fields or to a table with an identical key structure.
4. Select the AUDAT field (document date) from the table fields displayed.
5. Choose . The Create dialog box is then displayed.
6. Check the proposed texts. You do not need to make any changes.
7. Choose . The attribute is transferred into the object type definition with the automatically-proposed name DocumentDate.

Implementing attributes in the implementation program

Check the definition made so far. To do so, choose . The system detects that the database field attribute has not been implemented in the program of the object type and allows you to generate a template automatically for the missing source text. Make sure you choose this option. The automatic generation of the implementation program has satisfactory results for database field attributes so that no revision is necessary. The system then automatically goes to the implementation program of the object type you created. Here, you can see additional program components that control access to the table with the attributes. Do not change this part of the implementation!

Database field attribute

Database field attributes are implemented between the macro commands GET_TABLE_PROPERTY <table name> and END_PROPERTY. To program one access for all attributes that refer to a table, the tables are buffered initially in an internal table. This internal table is created from the database table in the subroutine FORM SELECT_TABLE_<table name> USING SUBRC LIKE SY-SUBRC. Analyze the implementation program as it stands. You can use the Implementation program in the appendix [Seite 1700] as a comparison.

- In the data declaration part, the variable OBJECT-_VBAK is declared as a structure VBAK (line 16).
Unit 2: Creating Database Field Attributes

- The table access is implemented between the two macro command `GET_TABLE_PROPERTY VBAK` and `END_PROPERTY` (lines 21-28).
- The automatically generated subroutine `FORM SELECT_TABLE_VBAK` that builds the internal table for buffering purposes is in lines 31-45.

**Result**

The object type must first be generated and released before it can be instantiated and tested.

1. Choose 🔄.
   
The system informs you if the object type still contains errors. Try to correct these errors in the error overview (Goto → Error list).

2. Choose Edit → Change release status → Object type → To implemented.
   
The attributes created up to now can then be tested.

**Testing the DocumentDate attribute**

1. Choose 🔄.
   
The Test Object Type <Object Name>: No Instance screen appears.

2. Choose 🔄 Instance.
   
   Identify an object of the type sales order by entering the number of a sales order of your choice. Use the F4 input help if necessary.

3. Choose 🔄.
   
The Test Object Type <Object Name> screen then appears, in which you can test your object type (execute methods, check attribute values).
Unit 3: Creating Object Database Field Attributes

Use
In this unit you create and implement the object database field attribute [Extern] sold-to party. Database field attributes are attributes that evaluate the content of a database field directly. In this case, the attribute reads the sold-to party from the database and creates from it a reference to an object of the type customer. The system helps you implement this attribute by generating source text automatically.

Procedure
1. Position the cursor on the entry Attributes.
2. Choose and answer the query Create with ABAP Dictionary field proposals? with Yes. This simplifies the entry of the data type reference.
3. Enter VBAK in the Table field.
4. Select the field KUNNR (sold-to party) from the table fields displayed.
5. Choose ✓.
6. Check the proposed texts. You do not need to make any changes.
7. Since the value of the database field is to be formatted and returned as an object reference, enter KNA1 (customer) in the object type field.
   Make sure that the object type entered here only has one key field and can be identified via the database field entered.
8. Choose ✓.
   The attribute is transferred into the object type definition with the automatically-proposed name SoldToParty.

Implementing attributes in the implementation program
Check your object type. To do so, choose 
If you have completed the previous unit, you will see that no further implementation is required. Database field attributes that refer to fields in the table VBAK are already fully implemented.

Result
Testing the SoldToParty attribute
1. Choose ✓.
   This regenerates your object type.
2. Choose ✓.
   The Test Object Type <Object Name>: No Instance screen appears.
3. Choose Instance.
   Identify an object of the type sales order by entering the number of a sales order of your choice. Use the F4 input help if necessary.
4. Choose ✓.
The Test Object Type <Object Name> screen then appears, in which you can test your object type (execute methods, check attribute values).
Unit 4: Creating Virtual Attributes

Use
In this unit you create and implement the virtual attribute [Extern] TextWithDocumentNo. This attribute returns the text sales order <order number> of <date>. This text must be “assembled” in the program of the object type.

A virtual attribute always requires a supplement to the proposed implementation in the implementation program.

Prerequisites
You use the macro commands provided by the system to implement these attributes.
For further information on the macro commands provided, see:

- Macro Commands for Processing a Container [Seite 1133]
- Macro Commands for Accessing Objects, Attributes and Methods [Seite 1146]

Procedure
1. Position the cursor on the entry Attributes.
2. Choose and answer the query Create with ABAP Dictionary field proposals? with No.
3. Enter the following texts for the attribute to be created:
   - Attribute: TextWithDocumentNo
   - Name: Text and number
   - Short description: Text and number of sales order
4. Select Virtual in the area Source.
5. Select Dictionary for the area Data type reference.
6. Enter SYST in the Reference table field.
7. Enter TITLE in the field Reference field.
   (You could also specify any character field with the required length here. Note that it is not the content of the field that is of interest, but its data type.)
8. Choose .

Implementing attribute in the implementation program
Check the definition of the object type so far. To do so, choose .
The system detects that the implementation is missing and allows you to generate a template automatically for the missing source text.

Make sure you choose this option.
The source text generated automatically for implementing virtual attributes is always incomplete and restricted to setting the relevant container element. You must make changes here and implement the read procedure in the implementation program, which determines the attribute value by evaluating the database contents at runtime.
The system then automatically goes to the implementation program of the object type you created.

Virtual attributes
A virtual attribute is implemented between the macro commands GET_PROPERTY <attribute
name> CHANGING CONTAINER and ENDPROPERTY. The program code here determines how the value of the virtual attribute is derived at runtime.

The value determined is then stored in a container. This container forms the interface of the object type.

When you implement a virtual attribute, you must write the attribute value determined to the container of the object. Use one of the following macros for this purpose:

- **SWC SET ELEMENT CONTAINER \'<ElementName>'<Attribute>** for single-line attributes.
- **SWC SET TABLE CONTAINER \'<ElementName>'<Attribute>** for multiline attributes.

These macro commands are already in the automatically-generated source text.

Analyze the implementation program as it stands. You can use the implementation program in the appendix [Seite 1700] as a comparison.

The variable **OBJECT-TEXTWITHDOCUMENTNO** is declared for the attribute (line 13 of the program, generated automatically).

You carry out the implementation yourself between the two macro commands **GET_PROPERTY TEXTWITHDOCUMENTNO CHANGING CONTAINER and END_PROPERTY** (lines 47-57).

- The text is constructed from the values of the attribute already created and the key field of the object type. These attributes are read with the macro command **SWC GET_PROPERTY** with reference to the object itself (object reference **SELF**). (Lines 51-52).

- Assemble the text and assign it to the container element **TextWithDocumentNo** using the macro command **SWC SET ELEMENT**. (Lines 53-56).

**Result**

**Testing virtual attributes**

In order to test the attribute, it must first be released and then the object type must be generated.

1. Position the cursor on the attribute **TextWithDocumentNo**.
2. Choose **Edit \(\rightarrow\) Change release status \(\rightarrow\) Object type component \(\rightarrow\) To implemented**.
3. Choose ✅.

The system informs you if the object type still contains errors. Try to correct these errors in the error overview (Goto \(\rightarrow\) Error list).

You can now test the attributes created up to now:

1. Choose ✅.

   The **Test Object Type <Object Name>: No Instance** screen appears.

2. Choose ✅ **Instance**.

   Identify an object of the type **sales order** by entering the number of a sales order of your choice. Use the F4 input help if necessary.

3. Choose ✅.

   The **Test Object Type <Object Name>** screen then appears, in which you can test your object type (execute methods, check attribute values).
Unit 5: Creating Object Virtual Attributes

Use
In this unit you create and implement the object virtual attribute `virtual attribute [Extern] sales group`. This attribute returns the sales group of the sales order as an object reference.

Objects of the type `sales group` have only one key field. Therefore, it is not actually necessary to implement this attribute as a virtual attribute. You could also create it as an object database field attribute.

This basic example is simply intended to demonstrate how a virtual attribute is implemented.

A virtual attribute always requires an implementation in the implementation program.

Prerequisites
You use the macro commands provided by the system to implement these attributes.
For further information on the macro commands provided, see:
- Macro Commands for Processing a Container [Seite 1133]
- Macro Commands for Accessing Objects, Attributes and Methods [Seite 1146]

Procedure
1. Position the cursor on the entry `Attributes`.
2. Choose `>Add Attributes` and answer the query `Create with ABAP Dictionary field proposals?` with `No`.
3. Enter the following texts for the attribute to be created:
   ```
   Attribute: SalesGroup
   Name: Sales group
   Short description: Sales group
   ```
4. Select `Virtual` in the area `Source`.
5. Select `Object type` in the `Data type reference` area.
6. Enter `TVKGR` in the `Object type` field.
7. Choose `✓`

Implementing attributes in the implementation program
Check the definition of the object type so far. To do so, choose `✓`

The system detects that the implementation is missing and allows you to generate a template automatically for the missing source text.

Make sure you choose this option.

The source text generated automatically for implementing virtual attributes is always incomplete and restricted to setting the relevant container element. You must make changes here and implement the read procedure in the program of the object type, which determines the attribute value by evaluating the database contents at runtime.

The system then goes to the program of the object type you created.
**Virtual attribute**

A virtual attribute is implemented between the macro commands `GET_PROPERTY <attribute name> CHANGING CONTAINER and END_PROPERTY`. The program code here determines how the value of the virtual attribute is derived at runtime. The value determined is then stored in a container. This container forms the interface of the object type. When you implement a virtual attribute, you must create an object reference. To create an object reference with given key fields and a known object type, use the following macro command:

- `SWC_CREATE_OBJECT <Object> <ObjectType> <ObjectKey>`.

The variable `<Object>` must first be declared with:

- `DATA: <Object> TYPE SWC_OBJECT.`

This data declaration is already in the automatically-generated source text. Analyze the implementation program as it stands. You can use the implementation program in the appendix [Seite 1700] as a comparison.

- Since the attribute is to return an object reference, a variable `OBJECT-SALESGROUP` with type `SWC_OBJECT` is declared automatically.

- The actual implementation must be carried out manually. It must be between the two macro commands `GET_PROPERTY SALESGROUP CHANGING CONTAINER` and `END_PROPERTY`. The system inserts the macro command between these macro commands, with which the object reference to be created is written to the container.

Add the following two lines to the program:

```
SELECT SINGLE * FROM VBAK WHERE VBELN = OBJECT-KEY-SALESDOCUMENT.
SWC_CREATE_OBJECT OBJECT-SALESGROUP 'TVKGR' VBAK-VKGRP.
```

- Save your program.

**Result**

**Testing the virtual attribute SalesGroup**

1. Position the cursor on the attribute just defined.
2. Choose `Edit → Change release status → Object type component → To implemented.`
3. Choose 📇.

   The system informs you if the object type still contains errors. Try to correct these errors in the error overview (Goto → Error list).

Test the attributes created so far.

1. Choose 📜.

   The Test Object Type `<Object Name>`: No Instance screen appears.

2. Choose 📜 `Instance`.

   Identify an object of the type sales order by entering the number of a sales order of your choice. Use the F4 input help if necessary.

3. Choose 📜.

   The Test Object Type `<Object Name>` screen then appears, in which you can test your object type (execute methods, check attribute values).
Unit 6: Creating Multiline Virtual Attributes

Use
In this unit you create and implement the multiline object virtual attribute\[Extern\] sales document positions. This attribute evaluates the items in the sales order, and returns a list with references to objects of the type VBAP (sales document item).
A virtual attribute always requires an implementation in the implementation program.

Prerequisites
You use the macro commands provided by the system to implement these attributes.
For further information on the macro commands provided, see:
- Macro Commands for Processing a Container [Seite 1133]
- Macro Commands for Accessing Objects, Attributes and Methods [Seite 1146]

Procedure
1. Position the cursor on the entry Attributes.
2. Choose No and answer the query Create with ABAP Dictionary field proposals? with No.
3. Enter the following texts for the attribute to be created:
   - Attribute: Items
   - Name: Order items
   - Short description: Sales order items
4. Select Virtual in the area Source.
5. Select multiline in the area attribute property.
6. Select Object type in the Data type reference area.
7. Enter VBAP in the Object type field.
8. Choose .

Implementing an attribute in the program of the object type
Check the definition of the object type so far. To do so, choose .
The system detects that the implementation is missing and allows you to generate a template automatically for the missing source text.
Make sure you choose this option.
The source text generated automatically for implementing virtual attributes is always incomplete and restricted to setting the relevant container element. You must make changes here and implement the read procedure in the program of the object type, which determines the attribute value by evaluating the database contents at runtime.
The system then goes to the implementation program of the object type you created.

Virtual attribute
A virtual attribute is implemented between the macro commands GET_PROPERTY <attribute name> CHANGING CONTAINER and END_PROPERTY. The program code here determines how the value of the virtual attribute is derived at runtime.
Analyze the implementation program as it stands. You can use the implementation program in the appendix [Seite 1700] as a comparison.

- As the attribute is to return a multiline list of object references, an internal table OBJECT-ITEMS with type SWC_OBJECT is declared (line 14 of the program, generated automatically).
- The actual implementation must be carried out manually. It must be done between the two macro commands GET_PROPERTY ITEMS CHANGING CONTAINER and END_PROPERTY (lines 65-96). The system inserts the macro command between these macro commands, with which the object reference to be created is written to the container.
- Data declaration (lines 69-78)
  - Refresh table OBJECT-ITEMS
  - Auxiliary variable ITEM with type SWC_OBJECT to hold one object reference
  - Structure VBAP_KEY with the two fields VBELN and POSNR. These fields correspond to the key fields of the table VBAP (sales document: item data).
  - Internal table VBAP_TAB with structure VBAP. The items for an order are “collected” in this table.
- Data selection (lines 81-91)
  - All items for the sales order identified by the key field of the object type are read from table VBAP and written to table VBAP_TAB (lines 81-82).
  - The entries in table VBAP_TAB are used in a loop (lines 87-91) as follows:
    - The structure VBAP_KEY is filled
    - An object reference ITEM is created
    - ITEM is appended to table OBJECT-ITEMS.
- Assignment (line 94)
  - The internal table OBJECT-ITEMS with the list of object references is assigned to the container element Items with the container macro SWC_SET_TABLE.

Result

Testing the virtual attribute items

1. Position the cursor on the attribute just defined.
2. Choose Edit → Change Release Status → Object Type Component → To Implemented.
3. Choose .
   - The system informs you if the object type still contains errors. Try to correct these errors in the error overview (Goto → Error list).

The attributes created up to now can then be tested.

1. Choose .
   - The Test Object Type <Object Name>: No Instance screen appears.
2. Choose Instance.
Unit 6: Creating Multiline Virtual Attributes

Identify an object of the type sales order by entering the number of a sales order of your choice. Use the F4 input help if necessary.

3. Choose ✔.

The Test Object Type <Object Name> screen then appears, in which you can test your object type (execute methods, check attribute values).

**Note: Object type VBAP (sales document item)**

The object type VBAP describes the individual items of a sales document. The key fields of the object type VBAP are:

- **SalesDocumentNo (sales document)**
- **ItemNo (sales document item)**

These key fields reference the key fields VBELN and POSNR in the table VBAP (sales document: item data).

**Alternative implementation**

An alternative implementation without an additional internal table is shown below.

```plaintext
get_property items changing container.

tables vbap.
refresh object-items. clear object-items.

data item type swc_object.
data: begin of vbap_key,
    vbeln like vbap-vbeln,
    posnr like vbap-posnr,
end of vbap_key.

vbap_key-vbeln = object-key-salesdocument.

select * from vbap where vbeln = object-key-salesdocument.
  vbap_key-posnr = vbap-posnr.
  swc_create_object item 'VBAP' vbap_key.
  append item to object-items.
endselect.

  swc_set_table container 'Items' object-items.
end_property.
```
Unit 7: Creating the Method Edit Without Parameters

Use

In this unit, you will create and implement the method Edit as a method without a parameter. You must implement this method if your object type supports the IFEDIT interface.

Prerequisites

The Edit method enables you to edit the object. This is one of the methods that your object type inherits from the IFEDIT interface.

Although this method is defined for the interface, it is not yet implemented. In other words, the program of your object type must be extended accordingly. To do this, the object-specific transaction (or function module) is called in the implementation of the method Edit, with which the object can be edited. For sales orders, this is transaction VA02.

Inherited Object Methods

If you want to edit a method that your object type has inherited from an interface or a supertype, note the following:

- An inherited method can only be edited if you release this method for overriding.
- You cannot change the name of the inherited method any longer.
- You cannot delete any parameters. You can only add non-mandatory parameters to the interface.
- When you maintain inherited methods, it is important that the entries you make for the ABAP functionality, call attributes, and result type are correct, since automatic program generation depends on this.

The following properties and exceptions of the method Edit are inherited from the interface IFEDIT:

- Method properties

  The method Edit is a synchronous method [Extern], which runs with dialog, is instance-dependent and does not return a result. (The task in which this method is referenced will usually have the property Confirm end of processing.)

Procedure

1. Position the cursor on the method Edit.
2. Choose OK.
   The color of the entry changes.
3. Double-click the method name to open the method definition.
4. On the tab page ABAP/4, enter the transaction VA02 in the field Name and select Transaction.
5. Choose OK.
6. Position the cursor on the method Edit.
7. Choose Program. Let the system create the template for the missing method.
An (almost) usable implementation is created from the specifications on the ABAP function type:

The method is implemented between the macro instructions
BEGIN_METHOD <method name> CHANGING CONTAINER and END_METHOD.

When the calls are implemented, the unique ID of the object is available in the structure of the key fields under the variable name OBJECT-KEY-SALESDOCUMENT. Using SET/GET parameters, the input field in the first screen of the transaction is filled from this key field. This first screen is then skipped when the transaction is called (... AND SKIP FIRST SCREEN).

8. Add the line SWC_REFRESH_OBJECT SELF to the implementation of the method Edit. You can find the source text in the appendix in lines 98-102.

The standard buffering of attribute values can give rise to the situation where attribute changes that occur when an object is edited are not "noticed" by the runtime system. Therefore, after the execution of methods that could have changed attributes (methods Edit, Update, Change, and so on), call the macro in the form SWC_REFRESH_OBJECT SELF before END_METHOD. This means that the attributes will be refreshed before the next read access, and not read from the internal table.

9. Choose and exit the Program Editor.

Result

Testing a method without parameters

1. The object type must first be generated so that it can be instantiated and tested. Choose .

   The system informs you if the object type still contains errors. Try to correct these errors in the error overview (Goto → Error list).

2. To test the method, choose .

   The Test Object Type <Object Name>: No Instance screen appears.

3. Choose Instance.

   Identify an object of the type sales order by entering the number of a sales order of your choice. Use the F4 input help if necessary.

4. Choose .

   The Test Object Type <Object Name> screen then appears, in which you can test your object type (execute methods, check attribute values).

Notes on the calling and reporting behavior of this method

In order for the method to be used in a workflow, it must be encapsulated in a task. This task is represented at runtime by a work item. If this work item is executed by one of its recipients [Extern], the runtime system calls the method and thus, indirectly, the transaction. As the method Edit is a synchronous method without results, the work item is completed as soon as the method has been executed once. This system behavior is not generally desirable:

- The current agent [Extern] does not receive any information as to whether anything was changed or not.
It is not possible to call the method several times for multiple changes. If you want to be sure that the work item only assumes the status *completed* if changes were actually made to the sales order, the method should not be defined as a synchronous object method. Instead, you must define the method as an *asynchronous method* and provide a terminating event for the task, which is created when a change is made to an object.

This event can be implemented by linking the event creation to the writing of change documents.

If the work item is to be executed several times, you must set the flag *Confirm end of processing* in the task definition.
Unit 8: Creating the ExistenceCheck Method Without Parameters

Use
In this unit, you will create and implement the method ExistenceCheck as a method without a parameter.
You must implement this method if your object type supports the SAP standard interface.

Prerequisites
The ExistenceCheck method checks whether the object identified by the key fields actually exists. The ExistenceCheck method is one of the methods inherited by every object type from the SAP standard interface (technical name: IFSAP).
This method is not yet implemented for the interface. The implementation program must be extended correspondingly.
The following properties and exceptions of the method ExistenceCheck are inherited from the interface IFSAP:

- Method properties
  The ExistenceCheck method is a synchronous method, which runs without dialog, is instance-dependent and does not return a result. If the object does not exist, the method is terminated with an exception.

- Exceptions
  The exception The object does not exist is defined for the method ExistenceCheck with the number 0001.
This exception must be implemented at the appropriate point in the source text. Use the macro command EXIT_RETURN 0001 SPACE SPACE SPACE SPACE for this purpose.

Defining exceptions
Exceptions can be defined for each method. For each exception, you must specify whether a temporary error or application or system error is involved. Each exception is identified by a 4-digit number and linked to a message text that is supplemented by up to four variables at runtime. You can use the following number ranges:
- 1001-7999: Application-specific exceptions (for SAP development)
- 9000-9999: Exceptions defined by customer

Implementing exceptions
You trigger the exception within the implementation of the method, that is within the macro commands BEGIN_METHOD and END_METHOD, if an error occurs when the method is processed. Use the macro command with the call: EXIT_RETURN Code Variable1 Variable2 Variable3 Variable4.
As the code (ABAP Dictionary reference SWOTINVOKE-CODE), enter the four-digit number of the exception which was specified when the exception was defined. The four variables (maximum) are determined from the definition of the message that is linked to the exception.
For further information, please refer to Programming Exceptions [Seite 1156].
Unit 8: Creating the ExistenceCheck Method Without Parameters

Procedure
1. Position the cursor on the method ExistenceCheck.
2. Choose
   The color of the entry changes.
3. Double-click the method name to open the method definition. You do not need to change anything here for this example. Choose ✔.
4. Position the cursor on the method ExistenceCheck.
5. Choose Program. Let the system create the template for the missing method.
6. The actual implementation must be carried out manually. It must be done between the two macro commands BEGIN_METHOD EXISTENCECHECK CHANGING CONTAINER and END_METHOD. There is no additional source text initially.
   Add the implementation program of the ExistenceCheck method, which you can take from the implementation program in the appendix [Seite 1700] (lines 104-111), to the program. The VBAK table, which contains the header data of a sales order, is scanned with the key entered. If this was not successful, exception 0001 is triggered.
7. Choose ✔ and exit the Program Editor.

Result
Testing a method without parameters
1. The object type must first be generated so that it can be instantiated and tested. To do so, choose
   The system informs you if the object type still contains errors. Try to correct these errors in the error overview (Goto → Error list).
2. To test the method, choose ✔.
   The Test Object Type <Object Name>: No Instance screen appears.
3. Choose Instance.
   Identify an object of the type sales order by entering the number of a sales order of your choice. Use the F4 input help if necessary.
4. Choose ✔.
   The Test Object Type <Object Name> screen then appears, in which you can test your object type (execute methods, check attribute values).
Unit 9: Creating the Display Method Without Parameters

Use
In this unit, you will create and implement the method Display as a method without a parameter. You must implement this method if your object type supports the SAP standard interface.

Prerequisites
The Display method displays the object. It is one of the methods inherited by every object type from the SAP standard interface (technical name: IFSAP). The method is already implemented for the interface, but this inherited implementation only displays the key fields of the object. To use the object-specific display functionality for the object from the application, the implementation must be redefined for your object type, and the transaction or function module of the application must be called, with which the object is displayed. For sales orders, this is transaction VA03. The following properties and exceptions of the method Display are inherited from the interface:

- Method properties
  The Display method is a synchronous method, which runs with dialog, is instance-dependent, and does not return a result.

Procedure
1. Position the cursor on the method Display.
2. Choose .
   The color of the entry changes.
3. Double-click the method name to open the method definition.
4. On the ABAP/4 tab, enter the transaction VA03 in the Name field and select Transaction.
5. Choose .
6. Position the cursor on the method Display.
7. Choose Program. Let the system create the template for the missing method.
   You can find the source text in the appendix in lines 113-116.
8. Choose and exit the Program Editor.

Result

Testing a method without parameters
1. The object type must first be generated so that it can be instantiated and tested. Choose .
   The system informs you if the object type still contains errors. Try to correct these errors in the error overview (Goto → Error list).
2. To test the method, choose .
   The Test Object Type <Object Name>: No Instance screen appears.
3. Choose Instance.
Unit 9: Creating the Display Method Without Parameters

Identify an object of the type sales order by entering the number of a sales order of your choice. Use the F4 input help if necessary.

4. Choose ✓.

The Test Object Type <Object Name> screen then appears, in which you can test your object type (execute methods, check attribute values).

Further note (can be ignored during the first read operation)
The exception The object does not exist is specified in the method definition. This exception is not implemented automatically. It should not normally occur.
To implement it anyway, the following program lines must be inserted immediately after BEGIN_METHOD DISPLAY CHANGING CONTAINER:

```
SWC_CONTAINER CONT.
SWC_CREATE_CONTAINER CONT.

SWC_CALL_METHOD SELF 'ExistenceCheck' CONT.
IF SY-SUBRC NE 0.
   EXIT_RETURN 0001 SPACE SPACE SPACE SPACE.
ENDIF.
```

First an empty container is created, which the subsequent macro expects as an input. The ExistenceCheck method is then called and evaluated referring to itself (reference to the object SELF).

Notes on the calling and reporting behavior of this method
In order for the method to be used in a workflow, it must be encapsulated in a task. This task is represented at runtime by a work item.
If this work item is executed by one of its recipients [Extern], the runtime system calls the method and thus, indirectly, the transaction. Since the Display method is a synchronous method without results, the work item is completed as soon as the method has been executed once. If the work item is to be executed several times, you must set the flag Confirm end of processing in the task definition.
Unit 10: Creating a Method with Parameters

Use
In this unit, you will create and implement the method `Create` as a method with parameters. Methods may require parameters [Extern] in order to be executed or return them after they have been executed. You must implement this method if your object type supports the IFCREATE interface.

Prerequisites
The method `Create` creates an object of the specified type. This is one of the methods that your object type inherits from the IFCREATE (Create) interface. This method is not yet implemented for the interface. The implementation program must be extended correspondingly. To do this, the transaction (or function module) of the application is called in the implementation of the method `Create`, with which the object is edited. For sales orders, this is transaction VA01. The following properties and exceptions of the method `Create` are inherited from the interface IFCREATE:

- Method properties
  - The method `Create` is a synchronous, instance-independent method, which runs with dialog and does not return a result.

Procedure
Analyzing application functionality
1. Choose Logistics → Sales and Distribution → Sales → Order → Create.
2. Choose System → Status.
   - You see that you have called transaction VA01.
3. Choose 📰.
   - The screen Create Sales Order: Initial Screen is displayed again.
   - The Order type is a required entry field. If you want to create a sales order using the method `Create`, you must either carry out dialog or pass the order type as a parameter when calling.
4. Position the cursor in the field.
5. Press F1.
   - You go to the dialog box with the field help.
   - You can see from the section Field data, that the input field involves the table field VBAK-AUART.
7. Leave the dialog box and the screen for creating sales orders. Return to the Business Object Builder in the processing of your object type.
Defining a method
1. Position the cursor on the method Create.
2. Choose . The color of the entry changes.
3. Double-click the method name to open the method definition.
4. On the tab page ABAP, enter the transaction VA01 in the field Name and select Transaction.
5. Choose .

Creating parameters
6. Position the cursor on the method Create.
7. Select Parameters.
   The method container, which is still empty, is then displayed.
8. Choose .
9. Answer the query Create with ABAP Dictionary field proposals? with Yes.
10. Enter VBAK in the Table field.
11. From the table fields displayed, select the field AUART (sales document type).
12. Choose .
   The dialog box for creating a parameter is displayed.
13. Create the parameter SalesDocumentType as proposed by choosing .
15. Position the cursor on the method Create.
16. Choose Program. Let the system create the template for the missing method.
   The implementation of the method Create, which was taken automatically from your specifications, remains unchanged.
   Using the provided container macros SWC_GET_ELEMENT (for single-line elements) and SWC_GET_TABLE (for multiline elements), import parameters of the method are read from the container (CONTAINER) and passed to the function module or transaction or its processing parameters (“SET/GET parameters”).
   Export parameters are set in the containers (CONTAINER) when a method is implemented. To do this, use the container macro SWC_SET_ELEMENT for a single-line return parameter or the container macro SWC_SET_TABLE for a multiline return parameter.
   You can find the source text in the appendix in lines 118-125.

Result
Testing a method with parameters
1. The object type must first be generated so that it can be instantiated and tested. Choose .
The system informs you if the object type still contains errors. Try to correct these errors in the error overview (Goto → Error list).

2. To test the method, choose 
   
   The Test Object Type <Object Name>: No Instance screen appears.

3. Choose Instance.
   
   Identify an object of the type sales order by entering the number of a sales order of your choice. Use the F4 input help if necessary.

4. Choose ✔️.
   
   The Test Object Type <Object Name> screen then appears, in which you can test your object type (execute methods, check attribute values).
Unit 11: Events and Their Creation

Use
In this unit, you create the event [Extern] delivery block cancelled.

Prerequisites
Events indicate the status changes of an object in the runtime system. Events always belong to an object and are therefore described as components in the object type definition.

Keep in mind that when you create an event for an object type, this is only the first step. You still need to ensure that the event is actually triggered.

Procedure
1. Position the cursor on the entry Events.
2. Choose .
3. Enter the following texts for the event to be created:

   Event: DelBlockCancelled
   Name: no delivery block
   Short description: delivery block cancelled

   Irrespective of whether you use uppercase and lowercase in the event name, the name is always entered in uppercase letters in the event receiver linkage table by the system.
4. Choose .
   The event has now been created and is available in the Business Object Repository. You can now use this event in the other definition tools in SAP Business Workflow, it is displayed via the F4 input help.
5. Position the cursor on the new event and choose Edit → Change release status → Object component → To implemented.
6. The object type must first be generated so that it can be instantiated and tested. Choose .
   The system informs you if the object type still contains errors. Try to correct these errors in the error overview (Goto → Error list).

Scenario for this event
The event DelBlockCancelled (delivery block cancelled) states that an existing delivery block has been deleted.
The event can therefore be used as a terminating event of a task, in which the asynchronous method DeliveryBlockReset (reset delivery block) is referenced. This method does not yet exist, but you could easily define it yourself as a copy of the Edit method. You must ensure that you define an asynchronous method.
At runtime, you can execute the task that encapsulates this method as often as you want and make changes to the sales order. Only when you really cancel the delivery block for your changes, is the event created and the task terminated.

**Creating the event**

This event can be created using a change document linkage referring to the field VBAK-FAKSK (*billing block in SD document*).

In the table for assigning change documents to events, assign the *change document object* VERKBELEG to the *object type* Z_BUS2032 and to the *event* DelBlockCancelled. **Also enter a field restriction** for this assignment, which states that the event is only to be created when a change is made to the field VBAK-LIFSK changing it from any value to the initial value.

For further information on creating events, please refer to *Wizards for Creating Events* [Seite 1324].
Unit 12: Basic Data

Use
The basic data contains the general administrative information on the object type.

Procedure

Maintaining Basic Data

To maintain the basic data, choose Goto → Basic data on the screen Display object type Z_BUS2032.

On the tab page Defaults, leave the method Display as default method [Extern].

On the tab page Defaults, enter the following the attribute TextWithDocumentNo as the default attribute.

Choose .
Unit 13: Final Steps

Use
To be able to use the object type you have created operationally, you need to generate and release it.

Procedure

Generate
To generate the object type and make it available to the runtime system, choose \( \text{Generate} \).

An object type can only be generated if all higher-level object types do not contain any errors. If these object types have not yet been generated, they are generated as well. When an object type is generated, all subordinate sub-types are generated as well as long as they do not have any errors.

Release
If you have saved your object type as a temporary object, it can never be released. You can only use it as a “full” object type in your system.

To release an object type, position the cursor on the object type and choose Edit \( \rightarrow \) Change release status \( \rightarrow \) Object type \( \rightarrow \) To released.

An object type should only be released if the basic functionality is ensured.

Test/Execute

1. To test the object type, choose \( \text{Test} \).
   The Test Object Type <Object Name>: No Instance screen appears.

2. Choose \( \text{Instance} \).
   Identify an object of the type sales order by entering the number of a sales order of your choice. Use the F4 input help if necessary.

3. Choose \( \text{Execute} \).
   The Test Object Type <Object Name> screen then appears, in which you can test your object type (execute methods, check attribute values).
Appendix: Program for Object Type Z_BUS2032

***** Implementation of object type Z_BUS2032 *****

INCLUDE <OBJECT>.

BEGIN_DATA OBJECT. " Do not change.. DATA is generated
* only private members may be inserted into structure private
DATA:
" begin of private,
" to declare private attributes remove comments and
" insert private attributes here ...
" end of private,
BEGIN OF KEY,
   SALESDOCUMENT LIKE VBAK-VBELN,
END OF KEY,
SALESGROUP TYPE SWC_OBJECT,
..ITEMS TYPE SWC_OBJECT OCCURS 0,
   TEXTWITHDOCUMENTNO LIKE SYST-TITLE,
   _VBAK LIKE VBAK.
END_DATA OBJECT. " Do not change.. DATA is generated

TABLES VBAK.

* GET_TABLE_PROPERTY VBAK.
DATA SUBRC LIKE SY-SUBRC.
* Fill TABLES VBAK to enable Object Manager Access to Table Properties
   PERFORM SELECT_TABLE_VBAK USING SUBRC.
   IF SUBRC NE 0.
      EXIT_OBJECT_NOT_FOUND.
   ENDIF.
END_PROPERTY.

* Use Form also for other(virtual) Properties to fill TABLES VBAK
FORM SELECT_TABLE_VBAK USING SUBRC LIKE SY-SUBRC.
* Select single * from VBAK, if OBJECT-_VBAK is initial
   IF OBJECT-_VBAK-MANDT IS INITIAL
      AND OBJECT-_VBAK-VBELN IS INITIAL.
      SELECT SINGLE * FROM VBAK CLIENT SPECIFIED
      WHERE MANDT = SY-MANDT
      AND VBELN = OBJECT-KEY-SALESDOCUMENT.
      SUBRC = SY-SUBRC.
      IF SUBRC NE 0. EXIT. ENDIF.
      OBJECT-_VBAK = VBAK.
   ELSE.
      SUBRC = 0.
      VBAK = OBJECT-_VBAK.
   ENDIF.
ENDFORM.

GET_PROPERTY TEXTWITHDOCUMENTNO CHANGING CONTAINER.
DATA:
   SALESDOCUMENT LIKE VBAK-VBELN,
DOCUMENTDATE LIKE VBAK-AUDAT.
SWC_GET_PROPERTY SELF 'SalesDocument' SALESDOCUMENT.
SWC_GET_PROPERTY SELF 'DocumentDate' DOCUMENTDATE.
CONCATENATE 'sales order' SALESDOCUMENT 'of' DOCUMENTDATE
  INTO OBJECT-TEXTWITHDOCUMENTNO SEPARATED BY SPACE.
SWC_SET_ELEMENT CONTAINER 'TextWithDocumentNo'
  OBJECT-TEXTWITHDOCUMENTNO.
END_PROPERTY.

GET_PROPERTY SALESGROUP CHANGING CONTAINER.
SELECT SINGLE * FROM VBAK WHERE VBELN = OBJECT-KEY-SALESDOCUMENT.
SWC_CREATE_OBJECT OBJECT-SALESGROUP 'TVKGR' VBAK-VKGRP.
SWC_SET_ELEMENT CONTAINER 'SalesGroup' OBJECT-SALESGROUP.
END_PROPERTY.

GET_PROPERTY ITEMS CHANGING CONTAINER.
* Declare data
TABLES VBAP.
REFRESH OBJECT-ITEMS.
DATA ITEM TYPE SWC_OBJECT.
DATA:
  BEGIN OF VBAP_KEY,
  VBELN LIKE VBAP-VBELN,
  POSNR LIKE VBAP-POSNR,
  END OF VBAP_KEY.
DATA BEGIN OF VBAP_TAB OCCURS 0.
  INCLUDE STRUCTURE VBAP.
DATA END OF VBAP_TAB.

* Select data
SELECT * FROM VBAP INTO TABLE VBAP_TAB
  WHERE VBELN = OBJECT-KEY-SALESDOCUMENT.
VBAP_KEY-VBELN = OBJECT-KEY-SALESDOCUMENT.

* Create object reference
LOOP AT VBAP_TAB.
  VBAP_KEY-POSNR = VBAP_TAB-POSNR.
  SWC_CREATE_OBJECT ITEM 'VBAP' VBAP_KEY.
  APPEND ITEM TO OBJECT-ITEMS.
ENDLOOP.

* Assign object reference to container element
SWC_SET_TABLE CONTAINER 'Items' OBJECT-ITEMS.
END_PROPERTY.

BEGIN_METHOD EDIT CHANGING CONTAINER.
SET PARAMETER ID 'AUN' FIELD OBJECT-KEY-SALESDOCUMENT.
CALL TRANSACTION 'VA02' AND SKIP FIRST SCREEN.
SWC_REFRESH_OBJECT SELF.
BEGIN_METHOD EXISTENCECHECK CHANGING CONTAINER.
  SELECT SINGLE * FROM VBAK
  WHERE VBELN = OBJECT-KEY-SALESDOCUMENT.
  IF SY-SUBRC NE 0.
    EXIT_RETURN 0001 SPACE SPACE SPACE SPACE.
  ENDIF.
END_METHOD.

BEGIN_METHOD DISPLAY CHANGING CONTAINER.
  SET PARAMETER ID 'AUN' FIELD OBJECT-KEY-SALESDOCUMENT.
  CALL TRANSACTION 'VA03' AND SKIP FIRST SCREEN.
END_METHOD.

BEGIN_METHOD CREATE CHANGING CONTAINER.
  DATA:
    SALESDOCUMENTTYPE LIKE VBAK-AUART.
    SWC_GET_ELEMENT CONTAINER 'SalesDocumentType' SALESDOCUMENTTYPE.
    SET PARAMETER ID 'AAT' FIELD SALESDOCUMENTTYPE.
    CALL TRANSACTION 'VA01' AND SKIP FIRST SCREEN.
    GET PARAMETER ID 'AUN' FIELD OBJECT-KEY-SALESDOCUMENT.
END_METHOD.
BC - Workflow Scenarios in Applications (BC-BMT-WFM)

Purpose
With SAP Business Workflow, SAP AG provides an efficient cross-application tool enabling integrated electronic management of business processes. SAP Business Workflow is a solution which has been integrated fully in the R/3 System and which enables customer-specific business process flows to be coordinated and controlled on a cross-application and cross-work center basis. SAP Business Workflow therefore enhances "ready-made" application software. The SAP Business Workflow definition environment can represent business processes simply and can respond to changing external conditions quickly, even in a live system, by adapting the existing business processes.

Workflow Scenarios
Many SAP applications use SAP Business Workflow enabling preconfigured workflow scenarios to be reused in various situations. The scenarios can either be implemented without any changes or configured for your business processes by making minor adjustments. These workflow scenarios reduce implementation time significantly and have been optimally configured for the respective application functions.

Many workflow scenarios are integrated in IDES (International Demonstration and Education System). It is possible to simulate the business processes of a model company in this fully-configured system.

Features
The workflow scenarios can be divided into three categories:

Creating events
Events are created to report status changes for an application object and to allow a reaction to the changes.

- Document 4711/98 posted
- Material XYZ created

These events can be used as triggering events for your own tasks or workflows. The events are therefore “connected” in a flexible and customer-specific way to application events, without having to modify the standard part of the application.

In some cases, the triggering of these events is not activated in the standard version, but depends on the Customizing settings. You can find further information in the application scenario documentation.

Providing SAP tasks
A task contains a task description and the connection to the application logic via the method for a business object. Before you can use a task productively, you must assign the tasks to its possible agents.

The tasks provided by SAP are generally used as steps in SAP workflows, but you can use them for your own developments as well.

- Release change request
- Change purchase order

If a workflow scenario only involves one task, the scenario can usually be regarded as a minimal solution for showing the connection between application functionality and SAP Business Workflow. For differentiated control, this SAP task should be replaced by a customer-specific task.
You can find further information in the application scenario documentation.

**Providing SAP workflows**

A workflow contains a complete workflow definition covering several steps. An SAP workflow has a complete workflow definition, but must still be adapted to the organizational environment of the customer.

- Release a purchase requisition
- Recruitment

In cases in which SAP workflows describe business processes which also occur in your company, or in cases in which changes should not be made to the SAP workflow for technical reasons, the SAP workflows supplied can be used without any changes or adapted using workflow configuration. In all other cases, the SAP workflows can be used as templates for your own developments. The existing process structures of the business application components, which are often represented within a transaction, are generally not replaced. SAP Business Workflow is seen as an integration level "above" the standard business functions and uses the existing transactions, function modules, and reports.
Aim and Structure of this Documentation

Use
This documentation describes the workflow scenarios which can be found in the standard system (SAP workflows). Each workflow scenario has its own description. The descriptions contain information about the features available in the standard system, as well as the options relating to settings and modifications for your own developments.

Prerequisites
This documentation does not replace the SAP Business Workflow documentation.
For information about using and calling the individual tools of SAP Business Workflow and about taking advantage of the complete functionality for your own enhancements and developments, refer to the SAP Business Workflow documentation [Seite 972].

Features
Each description of a workflow scenario is assigned to one of the application components in the R/3 System.
Each description has the same structure and usually contains the following topics:

<table>
<thead>
<tr>
<th>Features</th>
<th>Details</th>
</tr>
</thead>
</table>
| Process description             | You learn which business processes are supported by the workflow scenario, which employees are involved, and what advantages and benefits can be derived from using the workflow system for this scenario.  
                                  | This information is sufficient if you are interested in an initial overview of the options available in a particular application.         |
| Technical implementation       | You learn, in detail, which steps were carried out by SAP in order to represent the business process in the system.  
                                  | You are familiarized with the object types which form the basis of the scenario and you discover which tasks are defined to describe the activities to be carried out, which roles are resolved, and which events are used. The structure of the workflow used, as well as the interface and container, are described in greater detail.  
                                  | This information is particularly useful if you are planning to enhance and modify the scenario and if you must identify the points at which changes can be made.  
                                  | This section is not available or is very short if no enhancements are planned or appropriate.                                           |
| Preparation and customizing     | You learn which application-specific or scenario-specific preparations and settings are required to adapt the scenario to the business conditions in your company.  
                                  | This section contains notes and recommendations regarding the organizational plan and agent assignment.  
                                  | For further information refer to Settings and Preparations [Seite 1706].                                                          |
| Operation and connection to     | You learn how the technical and business connection between the scenario and the application is controlled, how the scenario can be started and how the corresponding application functionality must be called. |
| application functionality       |                                                                                                                                         |
Settings and Preparations

Purpose

As far as possible in the technical and business sense, the workflow scenarios can be run directly without any modifications. However, there are always some settings which must be entered in your system and which cannot be supplied by SAP.

Prerequisites

"Technical" Customizing for the workflow system

The Auto-Customizing function can be used to configure basic technical settings for the workflow system. The corresponding functionality is available both in SAP Business Workflow Customizing and via Tools → Business Workflow → Development → Utilities → Customizing.

Process Flow

Maintaining the organizational plan

The company-specific organizational plan describes the organizational assignment of an employee. This allows the responsibilities of employees for performing individual business activities to be defined in the form of activity profiles.

The organizational plan is part of the PD component "Organization and Planning". The organizational plan is maintained on a client-dependent basis. An organizational plan which was created for personnel management purposes (or is to be created) can be used in SAP Business Workflow without any changes, provided the workflow functionality and the personnel management application run in the same client.

Even if you wish to use one of the supplied workflow scenarios without making any changes, you must always define the organizational plan of your company. You only need to define those areas of the company in which you wish to use SAP Business Workflow. You should only differentiate individual jobs to the extent required by the different activity profiles.

The corresponding sections of this documentation contain proposals for organizational plans which have been adapted to the relevant scenario.

Agent assignment for single-step tasks

The SAP tasks supplied by SAP as part of the workflow scenarios are client-independent and their validity period does not expire.

Tasks must always be linked to their possible agents from an organizational point of view. It may therefore be the case that several employees with the same authorization on an organizational basis are offered the same task for execution. One of these employees assumes responsibility for and processes the task. This assignment principle supports automatic load distribution within work groups with the same activity profile.

The corresponding sections of this documentation contain proposals for linking the task in a scenario to the relevant employees.
Activating the triggering events for a task

Tasks can be started via events in response to status changes for an object. A task can have several triggering events. If one of the triggering events is generated in the system, the relevant task is started ("OR" operation).

If the task is used in a workflow, the workflow that contains the task as a step is not started by the event. The triggering events of a task are entered in its definition. If you activate the triggering events, the system automatically activates the associated event receiver linkage.

If the workflow scenarios described in this documentation are started via triggering events, this fact is mentioned in the relevant sections. If this is the case, the linkage between the triggering event and the receiver (task or workflow) must always be activated. This is usually carried out in Customizing, but can also be carried out later during production operation.

Customizing for individual workflow scenarios

Additional customizing activities will generally be necessary within the environment of the individual application scenarios. These are described in the following sections of this documentation.
Organizational Plan

Definition
Representation of the task-related, functional structure of your enterprise, created using tools from the Organizational Management component.
This functional organizational plan differs from the administrative enterprise structure and the personnel structure whose elements are relevant to Payroll Accounting (company code, personnel subarea or employee group, for example). These structures are found in their corresponding components.

Use
You can create several organizational plans in different plan versions, this provides you with the following options in Organizational Management:
- in one plan version, you depict your current valid organizational plan which you use for your current business processes (evaluations, Workflow, personnel planning, for example).
- In additional plan versions, you can depict organizational plans as planning scenarios (for Business Process Re-Engineering, for example).
You can compare the current organizational plan with the planning scenarios and transfer data from the simulated structures into the current organizational plan.

Structure
Organizational plans are normally created by assigning objects of the following types to each other:
- Organizational Unit
- Position
- Job
- Task
If you are using your organizational plan for Workflow, the following object types are also available:
- Standard task
- Workflow template
- Task group
The main elements of an organizational plan are

- an organizational structure [Extern], with which the reporting structure and task distribution are created using organizational units (departments, for example)

- Staff assignments [Extern] for each organizational unit, in which the current persons (employees), users and vacancies are listed

**Integration**

Organizational plans are generally related to objects from other components.

- If Integration with Personnel Administration [Extern] is active, the personal data for the staff assignments comes from the Personnel Administration component.

  These functional (organizational plan) and administrative (personnel and enterprise structures) structures come into contact if a person is assigned to an organizational plan (as the holder of a position) as well as an enterprise or personnel structure (that is, to a personnel subarea etc.).

- If the enterprise structure [Extern] is active, account assignment data can be obtained from Controlling.
Customizing for Tasks and Workflows

Use
In this Customizing activity you activate triggering events of tasks and workflows and assign possible agents to them.

Integration
This Customizing activity enables you to set up centrally in the implementation phase all the tasks and workflows that you will use. However, you can also carry out the steps required individually within the relevant definitions. You only have to call the definition in display mode to carry out the activities described. You do not have to call the definition in change mode.

Prerequisites
You must know:
Which of the tasks and workflows belong to scenarios that you want to use.
The application component to which these tasks and workflows are assigned.

Features
The Customizing of tasks and workflows involves the following activities:

Activation of triggering events
If SAP’s usage of the task or workflow includes starting with a triggering event, the linkage is supplied inactive and you have to activate it if required. This is the only way to restrict the workflow functions to the areas in which you actually want to use them.

Assignment of possible agents
All tasks that are represented by dialog work items [Seite 1372] when they are executed must be assigned to their possible agents [Extern], or be classified as general tasks [Extern]. Whether a task is executed as a single step or as a step in a workflow is immaterial.
All workflows that are to be started in dialog also have to be assigned to their possible agents.

Activities
This function for customizing tasks and workflows can be found in SAP Business Workflow Customizing under Perform Task-Specific Customizing [Extern]. You access the Customizing for particular tasks and workflows via the relevant application component.
SAP Business Workflow

Purpose

*SAP Business Workflow* can be used to define business processes that are not yet mapped in the R/3 System. These may be simple release or approval procedures, or more complex business processes such as creating a material master and the associated coordination of the departments involved. SAP Business Workflow is particularly suitable for situations in which work processes have to be run through repeatedly, or situations in which the business process requires the involvement of a large number of agents in a specific sequence.

You can also use *SAP Business Workflow* to respond to errors and exceptions in other, existing business processes. You can start a workflow when predefined events occur, for example an event can be triggered if particular errors are found during an automatic check.

SAP provides several workflows that map predefined business processes. These workflows do not require much implementation. For an overview of these SAP workflows, refer to *Workflow Scenarios in Applications* [Seite 1703].

Integration

*SAP Business Workflow* uses the existing transactions and functions of the R/3 System and does not change the functions. You can combine the existing functions of the R/3 System to form new business processes with *SAP Business Workflow*. The workflow system takes over control of the business processes. If you are already using *SAP Organizational Management*, you can use the organizational structure created there to have the relevant agents carry out the individual activities. It is possible to have an activity carried out by a position. This ensures that the respective occupiers of the position can carry out the individual activities during execution of the workflow. This means that personnel changes in your organization are taken into account immediately in the execution of a workflow.

Features

*SAP Business Workflow* provides a number of tools for defining and analyzing workflows as well as for monitoring operation.

The *Workflow Builder* is for displaying and making changes to workflows. You can make small extensions directly to the original workflows supplied by SAP, such as carrying out your own agent assignments or changing deadline monitoring.

There are several Workflow Wizards to support you in the definition of workflows, with which you can create specific parts of a workflow. The *Workflow Wizard Explorer* gives you an overview of the existing Workflow Wizards.

In order to make the functions of the R/3 Systems available to a workflow, you use business objects, which you can define and analyze in the *Business Object Builder*. These business objects are made available to the workflow in reusable tasks. The *Business Wizard Explorer* gives you an overview of all existing tasks.

The end user receives information about the activities they are to carry out in their *Business Workplace*. This provides them with a central overview of all the activities that they are authorized to carry out. They can commence the activities from here.

Several tools are available to the workflow system administrator, with which they can control and analyze the current workflows. The workflow system administrator is notified of problems automatically by the system.
Technical Principles of SAP Business Workflow

The definition and execution of a workflow can be divided into four main areas. A user executes workflows in their Business Workplace where the work items that they can execute are displayed. A workflow must be defined to be executed. To this end, a workflow definition is created in the Workflow Builder. This definition contains steps that are executed at runtime. The step either control the workflow directly or they contain a reference to a task. The task refers to a method of an object type in the Business Object Repository (BOR) and can be executed at runtime either automatically (background task) or by a user (dialog task).

Business Workplace and work items

Work items [Seite 1371] are displayed to the user for execution in their Business Workplace. Work items are instances of a workflow at runtime. Their are various types of work item. Only certain types are displayed in the Business Workplace.

Workflow and workflow definition

A workflow must be defined before it can be executed. This workflow definition s made up of steps that control the workflow or refer to the tasks to be executed. You can make additional specifications about agents and deadline monitoring for a step. These specifications are evaluated at runtime by the work item manager [Extern]. The workflow is started either manually or by the system at runtime. For the system to start a workflow, the workflow definition must contain a triggering event (for example the event "material created"). When the event occurs, the relevant workflow is started automatically.

When you activate a workflow definition, you automatically generate a runtime version. When the workflow is started (manually or automatically), the relevant runtime version is used for the execution. If the workflow definition is changed later and a new runtime version is generated, these changes do not affect workflows that are already being executed.
Tasks
Tasks describe elementary business activities. Tasks always refer to a method of an object type. Possible agents are defined for tasks. Tasks can refer to automatically executable methods (background tasks) or they can need a user to execute them (dialog tasks).

Object types and objects
An object type describes the data with which you want to work in a workflow, for example the object type Material. An object is an individual data record of an object type. Attributes are defined for an object type, which make up its data record (for example, material name or material number). Each object type has methods, in which activities are defined, which can be executed with the data (for example, create material). The transactions and functions of the R/3 System can be called in a method as can your own transactions or other applications. The last important component of an object type are its events. These describe the status changes that an object can undergo (for example, material deleted or material changed). A workflow can be started by an event of this kind being triggered.

The Business Object Repository provides an overview of all object types in the R/3 System. You can use or extend the existing object types as well as create new object types.
General Procedure Model

Purpose

With SAP Business Workflow, you can map business processes in the R/3 System and process them (several times if required) under the control of the workflow system. A workflow management system can process and monitor structured processes that:

- Contain a sequence of activities
- Reoccur in the same or similar forms
- Involve several people of groups of people
- Require a lot of coordination

In addition to this general procedure model for working with SAP Business Workflow, there are also role-specific procedure models. For descriptions of these procedure models, refer to Workflow Roles [Seite 94].

Prerequisites

Before working with SAP Business Workflow, you should have a clear idea of the areas in question and the extent to which you want to change and adapt business processes, and pass them over to electronic process control.

Process Flow

Use the following procedure model as orientation when working with SAP Business Workflow.

Organizational plan

The enterprise-specific organizational plan describes the organizational assignment of the employees. The aim of this is to determine the responsibility of employees for the execution of individual business activities in the form of activity profiles.

You maintain the organizational plan on a client-specific basis. You can use an organizational plan that was created (or is still being created) for HR purposes in SAP Business Workflow as well without making any changes, as long as the workflow functionality and the HR application are used in the same client. Generally, however, in each client you (only) map those sub-areas and organizational structures of your enterprise, in which you also coordinate business processes using SAP Business Workflow.

For further information, refer to Enterprise-Specific Organizational Plan [Seite 1708].

Objects and object type definition

You identify all objects involved in your business process. You sort out which business function you want to map in your scenario and which attributes you want to access for control purposes.

You check whether the relevant object types with their methods, attributes and events are defined in the Business Object Repository. The grouping of object types in the application component hierarchy and the option of searching generically for parts of a name help when looking for object types.

- If you find an object type whose definition meets your requirements, you can use it without making any modifications.
- If you find an object type whose definition does not quite meet your requirements, you extend its definition.
- If you do not find a suitable object type, you define your own object type.

For further information, refer to Extending and Adapting Object Types [Seite 1157].
Single-step tasks
You identify the single-step tasks involved in your business process. Establish which object method is to be executed with the single-step task and who is responsible for executing it. You then define the single-step tasks by specifying object type and method, and determine the possible agents of the single-step tasks.

In the test and development stage, you should specify all single-step tasks used as general tasks [Extern].

You check whether single-step tasks are already defined. The single-step tasks available are not usually sufficient, meaning that you define additional single-step tasks to meet your requirements. For further information, refer to Single-Step Task [Seite 1175].

Standard roles
You identify possible agent roles in your business process. You use roles when agents are to be found using specific, business, functionally-oriented criteria.

Specifying roles is always optional.

You can initially do without roles in the test and development stage.

You check whether you can use any existing roles. If you cannot use any existing roles, you define your own.

For further information, refer to Role [Seite 1277].

Events
You identify the events required to initiate and control the workflow and check whether these events are defined for the relevant object types.

You require an event that is not incorporated in the standard version, you must add the event to the relevant object type definition and ensure that the event is created.

For further information, refer to Event [Seite 1321].

Multistep tasks
To describe a business process, which normally comprises several steps, you first create a workflow task and then reference a workflow definition in it. You can use workflow templates as a basis for your own developments, helping you to make a start. You can, of course, also use the definition tools to define new multistep tasks from scratch.

For further information, refer to Multistep Task [Seite 1194].

Demo example and tutorials
You can learn the basic principles of workflow definition and execution in the demo example for processing a notification of absence.

For further information, refer to Demo Example: Processing a Notification of Absence [Seite 52]. There are also self-teach tutorials that introduce you step-by-step to selected functions. See SAP Business Workflow - Tutorials [Seite 1581].
Transport Workflow

Purpose

The transport workflow provides a framework for transporting enhancements or new developments of existing business functions in a system landscape. It provides a direct connection between development and transport administration. The transport workflow manages the transport process, determines the user for each individual step automatically, and then displays an interface which they can use to perform the task directly.

It is an efficient method of transporting a selected number of requests into a group of transport targets, and uses clearly defined approval steps to ensure the quality of your target systems. The requests can be transportable change requests, Customizing requests, relocation transports or transports of copies. The transport targets do not need to be located on defined transport routes. However, the transport workflow can involve some risks, caused by the dependencies between transport requests:

Import sequence
- It is important that you import requests in the correct order, so that development work is up-to-date in the target system.

Incompleteness
- It is important that the functions transported in the transport proposal are complete; otherwise errors may occur in the import system.

A request is not imported, but it contains an important data element. You use another request to transport a table that references this data element. Since the referenced data element does not exist in the target system, activation errors will occur when you import the second request.

The transport workflow is a generic workflow. Its ability to process the transport route configuration in TMS enables it to adapt itself to any system landscape. This means you can transport multiple requests into multiple targets, even if these targets are not located on the transport routes.

This reduces the amount of work for the transport administrator significantly, and also reduces the number of possible errors in the transport process.

You can use the transport workflow in two different ways.

Transport workflow as a transport strategy

If you have production systems in your landscape that can only accept approved transports, we recommend that you use the transport workflow to organize and coordinate the transport process.

To do this, set Workflow-controlled transports as your transport strategy and configure the transport workflow.

When you release a transport request, the transport workflow starts automatically and the screen Create Transport Proposal appears. The requests are then released implicitly when the transport proposal is sent to the transport administrator.

Special transport workflow (mass transports)

You can use the special transport workflow to make transports that do not follow the defined transport routes or that take place outside the normal transport schedule (part of the mass transport strategy). These transports may be corrections made in the development system that have to be transported into the production system without delay.
To use the special transport workflow, set *Mass transports* as your transport strategy [Extern] and configure the transport workflow [Extern].

**Prerequisites**

You have configured the transport workflow [Extern] for your system.

The users involved in the transport workflow have a user in the Workflow Engine [Extern] system/client.

One or more users have transport administration authorization [Extern].

**Process Flow**

The developer creates a transport proposal [Extern] in the Transport Organizer. This proposal contains the required transport requests. The transport proposal then appears in the TMS worklist [Extern] of the transport administrator. The administrator can then approve or reject the transport proposal [Extern]. The transport administrator can also make changes to the transport proposal, for example change its contents and the transport target.

After a transport proposal has been approved, the TMS imports the transport requests automatically into the specified target systems. If the proposal is rejected, it is sent back to the transport proposal inbox [Extern] for revision [Extern] by the responsible developer. If the import is successful, the proposal is sent back to the transport proposal inbox to be confirmed [Extern] by the creator of the proposal. The developer can complete the proposal by confirming it, or apply to have it transported into other systems.

We recommend that you only use the transport workflow to transport into those target systems defined by the direct transport routes. Only in the next step should you work out which are the next direct target systems, and then apply to transport into them. This is the best way to keep the transport landscape consistent and complete.

The transport workflow writes an action log for each transport proposal. This log contains all development and transport activities, allowing you to check on the entire process.

Developers and transport administrators can communicate directly by writing notes.

For more information on transport administration, see Transport Workflow (Administration) [Extern].

For more information on the development team, see Transport Workflow (Development) [Extern].
Form Integration with SAPforms (BC-BMT-WFM)

Purpose

The form integration functionality allows employees or customers who have never worked with R/3 to use the R/3 System. This includes those users, for example, who only have a MAPI mail client (MS Exchange, MS Outlook) or a Web browser and do not have access to SAPGUI. Using SAPforms these users can also carry out simple, recurring activities using forms [Seite 1721] that are specially tailored to their needs. With SAPforms and a suitable form development environment (Visual Basic, MS Outlook, MS Word, MS Excel), you can easily create these forms and link them to the R/3 System.

You can edit certain types of form offline (that is, without a connection to the R/3 System) using your mail client (provided that this is supported by your e-mail system). The data is transferred immediately the next time you connect to the R/3 System.

The form integration functionality, therefore, can be used to:

- Replace paper-based procedures with electronic forms that purposely have restricted functionality
- Simplify and speed up request and approval processes
- Facilitate simple self-service scenarios
- Provide better support for mobile computing (offline data entry)
- Start and control workflows in the R/3 System.

SAPforms is the interface that allows the integration of electronic forms [Seite 1722]

The form integration concept, however, also includes the integration of web forms [Seite 2284].
From a technical perspective, the functions are completely autonomous. The SAPforms interface only supports integration of electronic forms. Since both integration types have similar functions, we have combined them under the term “form integration”.

**Implementation considerations**

To use the electronic form integration functions, you must

- Install SAPforms
- Use a Windows 32-bit platform.
- You also need a development environment, such as Visual Basic, MS Outlook, MS Word, or Excel, to create the forms.

To use the Web form integration functions, you must install the SAP@Web Studio.

**Features**

**Starting a Workflow and Executing a Work Item**

SAPforms is an interface whose primary function is to start workflows and execute work items:

You can start a workflow by executing a form. The data entered in the form is then supplied to the workflow.

You can execute a work item as a form. You can display and edit the data from the work item container in the form.

This function is supported by electronic and web forms. See also:

- Starting workflows with electronic forms
- Starting workflows with Web forms
- Executing a work item as an electronic form
- Executing work items with Web forms
BAPI Call
You can also call BAPIs from an electronic form. This function is only available online. See also:
Calling a BAPI [Seite 2260]

Using IDocs
Finally, you can fill and use IDocs with electronic forms:
Using IDocs [Seite 2268]
Forms

Definition
Copy of a paper form with fields for displaying and entering data. Forms are made available electronically for general use and are processed on a computer.

There are two types of form:

Electronic forms [Seite 1722], such as those created with Visual Basic
Web forms [Seite 2284] for displaying and processing with an Internet browser

Use
Forms serve as a means of

displaying information from the R/3 System in a structured manner

entering data that is used and evaluated in the R/3 System

An employee, for example, can call a leave application form from his or her PC, fill in the necessary data, and send the form to the R/3 System. Here, a workflow is started to process the leave application.

A company publishes an order form with product information and input fields on its Internet homepage. A workflow in the R/3 System is also started here when the form is filled in and submitted.

See also:
Workflow examples [Seite 2258]

By distributing forms electronically, you can make sure that the latest version of the form is always available and used. Changes to the form can be made easily and quickly from a central department.

To avoid input errors, you can provide a set of possible input values or help texts to the users of the form.

You can even check the inputs in the form automatically.

Integration
To create Web forms, you need the SAP@Web Studio as a development environment.
To create electronic forms, you need the SAPforms Designer [Seite 1735] as a development environment.
Electronic Form Integration

Use
By using electronic forms, those users who do not use the SAPGUI can access the functions of the R/3 System.
Using electronic forms you can:
- start workflows [Seite 2234],
- execute work items [Seite 2245],
- call BAPIs [Seite 2260], and
- fill and use IDocs [Seite 2268]

The data can be transferred to the R/3 System either online (synchronously) via RFC or offline (asynchronously) via Internet mail.
The form integration functionality supports so-called “approval workflows” where a workflow is started and an associated work item executed with just one form. Compare the two application examples below.

Prerequisites
To use the functions described here, you must install SAPforms [Seite 1725] on your PC. This also applies to local PCs, such as notebooks used by external staff, which are used to access the R/3 functionality offline via mail.
You also need a development environment (Visual Basic, MS Outlook, MS Word, or Excel, to create the forms) to create the electronic forms [Seite 1729].
Specific default settings are set for data exchange via mail. This also applies to starting a workflow, executing a work item, and using IDocs.
For more information, see SAPforms Administration [Seite 1727].
Finally, some functions are only available in certain releases:

Start Workflow

<table>
<thead>
<tr>
<th></th>
<th>RFC</th>
<th>MAPI</th>
<th>SAPMAPI</th>
<th>SMTP</th>
</tr>
</thead>
<tbody>
<tr>
<td>Visual Basic</td>
<td>3.1G</td>
<td>3.1G</td>
<td>4.5A</td>
<td>3.1G</td>
</tr>
<tr>
<td>MS Outlook</td>
<td>4.5A</td>
<td>4.5A</td>
<td>4.5A</td>
<td>4.5A</td>
</tr>
<tr>
<td>MS Word</td>
<td>4.5A</td>
<td>4.5A</td>
<td>4.5A</td>
<td>4.5A</td>
</tr>
<tr>
<td>MS Excel</td>
<td>4.5A</td>
<td>4.5A</td>
<td>4.5A</td>
<td>4.5A</td>
</tr>
</tbody>
</table>

Execute Work Item

<table>
<thead>
<tr>
<th></th>
<th>RFC</th>
<th>MAPI</th>
<th>SAPMAPI</th>
<th>SMTP</th>
</tr>
</thead>
<tbody>
<tr>
<td>Visual Basic</td>
<td>3.1G</td>
<td>3.1G</td>
<td>4.5A</td>
<td>3.1G</td>
</tr>
<tr>
<td>MS Outlook</td>
<td>4.5B</td>
<td></td>
<td>4.5B</td>
<td></td>
</tr>
<tr>
<td>MS Word</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>MS Excel</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

BAPI Call

<table>
<thead>
<tr>
<th></th>
<th>RFC</th>
<th>MAPI</th>
<th>SAPMAPI</th>
<th>SMTP</th>
</tr>
</thead>
<tbody>
<tr>
<td>Visual Basic</td>
<td>3.1G</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>MS Outlook</td>
<td>4.5A</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>MS Word</td>
<td>4.5A</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>MS Excel</td>
<td>4.5A</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

From Release 4.0A "SAPforms Transmit Plug-in for SMPT" has been replaced by the "SAPforms Transmit Plug-in for SMPT via WinSock".
Electronic Form Integration

### IDoc Support

<table>
<thead>
<tr>
<th>Visual Basic</th>
<th>RFC</th>
<th>MAPI</th>
<th>SAPMAPI</th>
<th>SMTP</th>
</tr>
</thead>
<tbody>
<tr>
<td>4.5A</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>MS Outlook</td>
<td>4.5A</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>MS Word</td>
<td>4.5A</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>MS Excel</td>
<td>4.5A</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Blank boxes indicate that the function is not available (yet).

### Features

**The SAPforms Interface**

After you have created the electronic form with Visual Basic or MS Outlook, for example, you must prepare it for communication with the R/3 System. This is the task of the [SAPforms Designer](Seite 1735).

When you are working with the SAPforms Designer, you use [plug-ins](Seite 1746) to define the following:

**Interface** to the R/3 System:

With the [SAPData type](Seite 1895) plug-in, you specify the interface to the R/3 System that is to be used by the form. The Workflow Container, BAPIs, and IDocs are supported as interfaces.

**Form type**:

With [SAPForm type](Seite 1755) plug-in, you specify the development environment (Visual Basic, MS Outlook, MS Word, MS Excel) in which the form was created.

The SAPforms plug-in Release 4.5A does not support the import of IDocs to earlier R/3 back-end releases.
Communication method:

With the SAPXmit type [Seite 2123] plug-in, you specify the communication method that is to be used to exchange data with the R/3 System. Both online communication via RFC and offline communication via mail (MAPI, SMTP) are supported. Note that the BAPI interface only supports online communication.

The binding between the form and the R/3 System is also defined using the SAPforms Designer. The SAPforms Designer then inserts a few lines of program code in the form when it is generated to evaluate the binding definition at runtime and communicate with the R/3 System. You can also call the SAPforms interface [Seite 2231] directly, that is, without using the SAPforms Designer.

Examples

Examples of the scenarios described here are provided in the directory ...\SAPforms\Samples.

Workflow examples [Seite 2258]

Example: Create customer master record [Seite 2264]

Example: Create customer master record via IDoc [Seite 2280]
Installing SAPforms

Procedure
To install SAPforms, run the SAPSETUP.EXE program from the SAP Presentation CD. Choose the Forms SDK component.

⚠️

SAPforms will only run on a Windows 32-bit platform!
The setup program will then guide you through the installation process. During this process, the SAPforms ActiveX components are installed to the subdirectory SAPforms\Bin. The SAPforms ActiveX components are registered.

some shared components are installed in the Windows system directory.

the SAPforms\Samples directory is installed that contains the examples referred to in this documentation.

subdirectories are installed under SAPforms that contain the source code of some of the SAPforms components.

Once you have installed SAPforms, you should start with the introductory example Getting Started [Seite 1739].

Result
To verify that installation succeeded please execute the following programs:

TestOCX1.EXE in the SAPforms\Samples directory.

This program checks whether the SAP components WDTREE.OCX and WDTVOCX.OCX were correctly installed.

TestOCX2.EXE in the SAPforms\Samples directory.

This program checks whether the SAP components WDTLOG.OCX and WDTFUNCS.OCX for remote communication with R/3 were correctly installed. You do not need to run this program, and indeed will not be able to, if your R/3 System is not available.

To perform the check, choose R3Logon and log on to an available system. Then choose Call Function Module to run a test RFC.

WDFDES.EXE in the SAPforms\Bin directory.

If the binding dialog screen of the SAPforms Designer then appears, SAPforms was installed correctly. WDFDES.EXE is the stand-alone version of the SAPforms Designer [Seite 1735] and has its own desktop icon (sapfff.ico).

If you want to start an SAP Business Workflow via Internet mail or MAPI mail, the corresponding components should be configured. To do so:

Start the OPTIONS.EXE program in SAPforms\Samples.
Select the SAPformsXSock.SAPXmit and SAPformsXMAPI.SAPXmit plug-ins [Seite 1746].
Choose Options and enter the corresponding settings.
The following applications can be accessed via the Start menu of your PC:
SAPforms - Administration

Use

You use the functions for the administration of SAPforms, in particular, when you want to communicate via the Internet with electronic forms. A trace function and a diagnosis tool are also available.

Features

Default Settings for Starting a Workflow

Check Customizing for Workflow

The Customizing settings for the workflow system must be complete. Customizing for Workflow is largely automated. For further information, see Customizing [Extern].

Check System Administration

The R/3 System must be configured in such a way that Internet mails can be sent and received. Unlike the same functions via RFC, a greater number of components is involved here: mail client, mail server, gateway to SAP R/3, SAPconnect, inbound distribution, and the R3F inbound processing functions (workflow).

Entry in the inbound distribution function (transaction SO28)

Communication between the R/3 System and the external mail system requires one user on the R/3 side that functions as a sender and receiver of the Internet mails. The user who is also the user for the logical destination WORKFLOW_LOCAL_<Mdt.> is used here. (<Mdt.> = R/3 System client) is entered. If auto-customizing was carried out without any manual intervention, this user is WF-BATCH.

You can view the settings by calling transaction SO28 Inbound Distribution: settings for recipient determination.

The standard user WF-BATCH forwards inbound R3F messages to the “handler for messages to work items” as an attachment to an Internet mail. The handler then calls a function module to forward the transferred data to the container. WF-BATCH, therefore, must be assigned an Internet mail address and the entry “handler for messages to work items” must be entered in the New recipient field. Use the possible entries pushbutton for this purpose (F4).

Configuring Automatic Forwarding

The automatic forwarding function must be set up for every user who wants to start a workflow with a form, wants to process a work item as a form in an external mail system via Internet mail.

The Internet mail address that is entered for automatic forwarding is used as the receiver address of the Internet mail for processing the form. When an Internet mail is received, the function ensures that the workflow (work item) is started (processed) by a certain agent who is known in the organizational plan.

Default Settings for Processing a Work Item

The following settings must be defined in addition to the default settings for starting a workflow.
Scheduling a Report for Periodic Transmission

The Internet mail address that is maintained for automatic forwarding is used as the recipient address for the Internet mails that are created for processing forms externally. In addition, a report must be scheduled to search the system at regular intervals for work items that fulfill the following criteria:

The work items represent a single-step task that executes the `Process` method for the `FORM` object type.

The work items have been created since the last time this report was run. The time at which the report was last run is entered in the table `SWU_WLSCAN` for this purpose.

The work items are “ready”.

If a work item that fulfills the above criteria is found and if one of the selected agents of this work item has activated the automatic forwarding function, an Internet mail is sent for the work item to this Internet mail address.

Scheduling a report for transmitting R3F messages is an activity in Workflow Customizing [Extern]. You must define these steps if you want to process a work item as a form in an external mail system.

You can also define this setting in normal operation using the transaction `SWUX` (SAPforms Administration).

Diagnosis Tool and Trace Function

The diagnosis tool provides various reports for error analysis. Documentation can be viewed for the individual functions via [1].

In particular, the results of the SAPforms trace can also be analyzed here. The trace must have been activated previously.

Activities

Choose `Tools` → `Business Workflow` → `Development` → `Administration` → `SAPforms` →

<table>
<thead>
<tr>
<th>What?</th>
<th>Where?</th>
</tr>
</thead>
</table>
| Entry in the inbound distribution | → `Mail enabling` → `Periodic`.  
The Scheduling Report for Sending SAPforms Messages screen is then displayed.  
Here you choose `Environment` → `Enter inbound processing`.  
The entry is then made in inbound distribution. |
| (De-)activate SAPforms trace | → `Trace on/off` |
| Diagnosis tool | → `Diagnosis` |
| Schedule reports (periodically) | → `Schedule batch job periodically` |
| Schedule report (once, test) | → `Start batch job once` |
| Workflow Customizing | `Tools` → `Business Workflow` → `Development` → `Utilities` → `Customizing` |
Designing Electronic Forms

Purpose
An employee does not have a SAPGUI on his or her local PC but still wants to start a workflow, execute a work item, create a customer (BAPI call), or use an IDoc in the R/3 System. These activities should be possible both online and offline (for example, on an external notebook).

Prerequisites
You must install SAPforms. You also need a development environment for electronic forms (Visual Basic, MS Outlook, MS Word, or MS Excel). You must also be familiar with the functionality of the SAPforms Designer (Getting Started).

Process Flow
To design an electronic form, you must work through the following steps:

In R/3:
Define a workflow, define a single-step task to be executed as a form, choose a BAPI or IDoc. Specific default settings are required for communication via mail. For more information, see SAPforms Administration.

In the form development environment:
Design the form
- Designing forms with Visual Basic
- Designing forms with MS Outlook
- Designing forms with MS Word and MS Excel

Call the SAPforms Designer or include the submit control or call the SAPforms interface directly.

In the SAPforms Designer:
- Maintain the plug-ins
- Define and generate the binding

Result
Once your form has been designed and processed by the SAP Forms Designer, you can make the form accessible to the end users.
Since SAPforms does not have its own mechanism for distributing forms to end users, you must either use existing mechanisms or create your own. For example, you can pull down your forms from public folders in Exchange or from your intranet.
Forms that are directly launched by the end user, for example a form starting a purchase requisition workflow, can be stored locally on the user’s PC.
Forms that are launched indirectly, for example by executing a work item in the R/3 System, are normally started via the SAPforms Helper. In this case, the forms must be stored in the active directory or under /SAPforms/Forms.
Designing Forms with Visual Basic

Use
To use the form integration technology, you need both SAPforms and a development environment for electronic forms.
Visual Basic is one such development environment.
It is also possible to create forms in MS Outlook [Seite 1731], MS Word [Seite 1733] and MS Excel [Seite 1733].
In this documentation, it is assumed that you are using Visual Basic. This applies to the application examples in particular.

Prerequisites
You must install SAPforms [Seite 1725].
You should have a basic knowledge of programming with Visual Basic.

Features
SAPforms largely consist of ActiveX components.
If you are using the SAPforms Designer [Seite 1735], these components are addressed directly when the form is generated. On one hand, the SAPforms Designer inserts code in your form (Code inserted by the SAPforms Designer [Seite 2230]) that accesses these components.
On the other hand, all of the important form information is stored in the TAG attribute of the form. This applies in particular to the bindings between the form controls and the interface parameters in the R/3 System.
You can also address the SAPforms interface directly with Visual Basic by programming your own ActiveX components. See also: Programming forms [Seite 2228]

Activities
Choose File → New Project to create a new Visual Basic project. Choose the project type Standard.EXE.
Insert the controls that you need to provide the required functionality in the form (“form 1”). If you want to want to display or fill multiple-line elements from the R/3 System, use the SAP Table View Control or the MS FlexGrid Control.
Call the SAPforms Designer via add-ins, choose an interface, define the bindings, and generate the form.
Or: Program [Seite 2228] the interface directly.
Or: Use the SAP Submit control [Seite 1747].
Compile an EXE file from your Visual Basic project and distribute it to the users.
Designing Forms with MS Outlook

Use
To use the form integration technology, you need both SAPforms and a development environment for electronic forms. MS Outlook is one such development environment. It is also possible to create forms in Visual Basic, MS Word, and MS Excel. In this documentation, it is assumed that you are using Visual Basic. This applies to the application examples in particular.

Prerequisites
You need MS Outlook 97 or higher. You need the VBScript Engine 2.0 or higher. You must install SAPforms. You should be familiar with the basic functions in MS Outlook, particularly with regard to designing forms.

Features

General
SAPforms largely consist of ActiveX components. If you are using the SAPforms Designer, these components are addressed directly when the form is generated. Unlike Visual Basic, the SAPforms Designer cannot be called as an add-in from MS Outlook. The Submit control must be used instead.

Form template
To display work items, the form template WORKITEM.OFT is available as a standard template and is found in directory \SAPforms\Forms. This template is then always started when you want to display a work item in your Outlook inbox (via SAPMAPI). Double-click on the file and publish the form under the name SAP.WORKITEM in any folder. To execute each work item the form maintained in the workflow definition is started. This can be an Outlook form or also a Visual Basic form.

You can find examples of such forms in the directory \SAPforms\Forms.

The template WFAPPROV2.OFT allows the user to display and execute work items without the standard template being started (WORKITEM.OFT). You can use this form template as a template for your own forms.

When you publish the form, you must enter the following message class:

SAP.WORKITEM.DIALOG.<task ID>

Starting workflows with attachments
An Outlook form, which should have a workflow started with it, can have an attachment inserted in the usual way. The attachment is then automatically included in the attachment list when the workflow is started.
Activities

Design a form by choosing Tools → Design Outlook Form on the Untitled Message screen.

If you want to display or fill multiple-line elements from the R/3 System, use the SAP Table View Control or the MS FlexGrid Control.

To insert the Submit control in your form, activate the Control Toolbox. If the Submit control is not in the toolbox, you can add it by choosing Custom Controls... (SAPforms Submit Control) from the context menu.

Choose Save.

Choose the Properties entry in the context menu of the control (right mouse click). Choose the property (user-defined) in the following dialog box. The Properties dialog box appears.

Choose the SAPforms-Designer pushbutton on the General tab page.

Edit and generate the form.

Exit the SAPforms Designer.

Exit the SAPforms Designer and publish the completed form in the desired folder.

If you want to modify the form later using the stand-alone version of the designer, remember to store the form in *.OFT format first. You can then load the form in the SAPforms Designer with Open Form.
Designing Forms with MS Word and MS Excel

Use
To use the form integration technology, you need both SAPforms and a development environment for electronic forms. 
**MS Word** and **MS Excel** are examples of such development environments.
It is also possible to create forms in **MS Outlook** and **Visual Basic**.

In this documentation, it is assumed that you are using Visual Basic. This applies to the application examples in particular.

Prerequisites
You need MS Office 97 or higher.
You need the VBScript Engine 2.0 or higher.
You must install **SAPforms**.
You should be familiar with the basic functions in MS Word and MS Excel.

Features
SAPforms largely consist of ActiveX components. If you are using the **SAPforms Designer**, these components are addressed directly when the form is generated.
Unlike Visual Basic, the **SAPforms Designer** cannot be called as an add-in from MS Word and MS Excel.
The **SAPforms Submit control** must be used instead.

Activities
Using MS Word
Design a form in MS Word. Use the input fields of the **text form fields** form (**View** → **Toolbars** → ...). You can define bindings for these text form fields when you are working with the **SAPforms Designer**.

Insert a new section.

Use the **Control toolbox** (**View** → **Toolbars** → ...) to insert the Submit control in the new section.
Choose the **Other controls** symbol in the **SAPforms Submit Control** list displayed.

Choose the **Properties** entry in the context menu of the control (right mouse click). Choose the property **(user-defined)** in the following dialog box. The **Properties** dialog box appears.
Choose the **SAPforms-Designer** pushbutton on the **General** tab page.

Edit and generate the form.
Exit the **SAPforms Designer**.

Protect the first section of your document.

You can now use the form. You can also maintain the **document properties** with **File** → **Properties**. These properties are also recorded by the **SAPforms Designer** and can be edited.
This means that the form can be used to transfer background data, for example.
Using MS Excel

Design a form in MS Excel. Use the input fields of the named cells form (Insert → Names→ ...). You can define bindings for these cells when you are working with the SAPforms Designer.

Use the Control toolbox (View → Toolbars → ...) to insert the Submit control. Choose the Other controls symbol in the SAPforms Submit Control list displayed.

Insert the control in the form.

Choose the Properties entry in the context menu of the control (right mouse click). Choose the property (user-defined) in the following dialog box. The Properties dialog box appears. Choose the SAPforms-Designer pushbutton on the General tab page.

Edit and generate the form.

Exit the SAPforms Designer.

Cancel the block for the named cells and protect the worksheet.

Exit the draft mode. You can now use the form.

You can also maintain the document properties with File → Properties. These properties are also recorded by the SAPforms Designer and can be edited. This means that the form can be used to transfer background data, for example.
SAPforms Designer

Use

The SAPforms Designer is a tool that allows you to edit electronic forms so that they can communicate with the R/3 System. In this way, elements from a workflow container can be displayed, for example, and modified via the form. These elements can also contain several lines.

The SAPforms Designer allows you to define and generate the necessary bindings between the controls on an electronic form and the elements of a certain interface in the R/3 System.

When the form is executed (at runtime), the interface elements are displayed or modified by the controls of the form according to this binding definition. The interface is also selected in the SAPforms Designer.

Prerequisites

Since the SAPforms Designer is an integral component of SAPforms, you must install SAPforms [Seite 1725].

You will also need a development environment for your electronic forms. Visual Basic, which is also used in the introductory example (Getting Started [Seite 1739]), is an example of such a development environment. However, you can also design forms with MS Outlook, MS Word, and MS Excel.

There are three ways of using the SAPforms Designer:

If you design the form in a Visual Basic environment, you can use the SAPforms Designer as an add-in. In Visual Basic, choose Add-Ins → Add-In Manager and select the entry SAP Forms Designer.

The second option is to call the SAPforms Designer directly. This method works with all form types, even those whose development environment does not support add-ins.

The third option is to use the Submit control [Seite 1747] of SAPforms. The SAPforms Designer is also launched when you use this control.
**Features**

**Screen Layout**

**Interface as an overview tree**

The top part of the screen displays the selected interface description as a tree structure. The individual elements in this structure correspond to the elements of your selected workflow container, for example.

**Form fields**

The bottom part of the screen displays the bindings in a table. The form controls supported by SAPforms are displayed in the Form Field column. If you have not defined any bindings yet, the SAP data field column is empty.

**Allocation using "drag and drop"**

To define the binding, you must fill the individual cells with the elements from the tree structure displayed above. This can be performed using either "drag and drop" or via the Binding menu.

**Menu functions**

**The file menu**

<table>
<thead>
<tr>
<th>Function</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Open form</strong></td>
<td>Opens an existing electronic form. This menu option is only displayed if you call up the SAPforms Designer directly. By choosing the SAPforms Form Plug-In [Seite 1755], you can define whether a Visual Basic form (extension FRM), Outlook form (extension OFT), or an office form (extension DOC or XLS) is opened. If you have already generated this form, the binding information is displayed when you open the form.</td>
</tr>
<tr>
<td><strong>Open SAP structures</strong></td>
<td>Opens a SAPforms-compatible interface to the R/3 System. The following interfaces are supported: Workflow containers (container structure)</td>
</tr>
</tbody>
</table>
The interface type last selected is displayed here.

Consistency checks
This function checks whether the binding within an open form is still consistent.

Generate
This function generates the selected form.

Generate and Close
This function generates the form and closes the SAPforms Designer.

Close
Choose this function to quit the SAPforms Designer.

The Display Menu

Node details
This function displays detailed information on the selected node.

The binding menu
The binding menu contains various functions with regard to the "binding definition". The binding definition controls the data that is transferred to the R/3 System via the form controls.

To the data element
Defines a binding between a form control and a structure element. You can also do this with drag and drop.

When you use the menu function, first select a structure element and then a form control. It is possible to fill several structure elements via a form control.

Release
Deletes a single binding.

Release all
Deletes all bindings.

Reset
Deletes all bindings as well as the current interface structure. This function deletes the entry in the TAG property of the form in particular.

The Tools Menu
The Options menu (Tools → Options) contains functions for maintaining plug-ins.

Plug-Ins Tab
You can enter the required plug-in in the Active Plug-In field. The SAPforms Designer uses plug-ins to support various interfaces in the R/3 System, various environments for designing forms, and various methods for communicating and exchanging data with the R/3 System.

You can also maintain Properties for all plug-ins.

Only the possible SAPXmit-plug-ins are displayed for the SAPData-plug-in selected.

General Tab
You can maintain the RFC user ID and the RFC Language here.
By choosing the pushbutton *Restore default settings* you can reset all the settings to their default values.

**Generate Tab**

If you choose the *Insert initialization code into Load method*, the SAPforms Designer inserts a few lines of program code in the form to evaluate the binding definition at runtime and communicate with the R/3 System. If you do not select this option, you must enter the code yourself.

You can add a *Submit* button to your form via the *button to trigger the insert action*. This will then be used to start communication with the R/3 System.

This information and all other relevant information is stored in the *TAG* property of the form when the form is generated.

### The Help Menu

**Tips when displaying start**

Via this function you can activate or deactivate the *Tips&Tricks* which are displayed at the start of a SAPforms Designer terminal session.

### Activities

To launch the Designer as an *add-in* under Visual Basic, open a form in the Visual Basic development environment and choose *Add-Ins → SAPforms → Designer*. The main window of the Designer is then displayed.

You can also start the SAPforms Designer as a stand-alone version or via the Submit control.

All of the supported form controls are displayed in the table control in the bottom part of the window. If you have already processed the form with the Designer, the structure information of the SAP interface and the existing binding are also displayed.

Working with the SAPforms Designer usually entails the following steps:

- **Maintain Plug-Ins**
  - *Maintaining plug-ins and opening a structure [Seite 1744]*

- **Define a binding**

- **Generate form**
  - *Define and generate binding [Seite 2226]*
Getting Started

This chapter provides a brief example of how to design a Visual Basic form for starting a workflow in the R/3 System. This form could start a workflow that handles approval of a leave application. The form that we are about to design represents the leave application.

Prerequisites

You have installed SAPforms [Seite 1725].

Procedure

To familiarize yourself with Visual Basic gradually, you should start with a simple Visual Basic project that contains a form with controls and no Visual Basic code. SAP Forms Designer will add all the definitions you need for starting a workflow with this form. You do not need to write any code for this purpose. Work through the following steps:

Start Visual Basic.

Check whether the Add-Ins menu contains the SAPforms entry. If this is the case, skip the next step.

Choose Add-Ins → Add-In Manager and select the SAPforms entry.

⚠️ In some cases, the SAP Forms add-in might not be listed in the Add-Ins menu of Visual Basic or in the Add-In Manager. This can happen, for example, if you installed Visual Basic after you installed SAPforms. In this case please do the following:

a) Close Visual Basic
b) Open the MD DOS command prompt.
c) Open the Bin directory under SAPforms
d) Enter WDFDES.EXE /R..
e) Start Visual Basic again.

Open the project Demo3a in the ..\SAPforms\Samples directory.

Open the form frmTst3a.frm in the project window. This is an absence form that triggers a workflow in the R/3 System.

Choose Add-Ins → SAPforms → Designer.

The main window of the SAPforms Designer is then displayed. This window essentially consists of two large controls.

The top control is still empty.

The bottom control - a table control - contains a list of the elements in the “leave request” form.

Choose Tools → Options.

The Options dialog box of the SAPforms Designer is then displayed.

Select the tab page Plug-Ins.
Enter SAP Forms Data Plug-In Demo in the Active Plug-In field in the SAP Forms Data Plug-In box. Choose this entry from the list of possible entries (F4).

In the frame SAP Transmit Plug-In the value for the SAP Forms Transmit Plug-In Demo is now entered automatically in the Active Plug-In field.

Click OK to exit this dialog box.

The main window of the SAPforms Designer is then displayed again.

Open the SAP data structure via File → Demo Structure.

The top control of the main window contains a tree structure of the demo data. Click on the “+” to display the fields in the structure.

Since you chose the SAPforms Data Plug-In Demo plug-in in step 8, a data structure stored for demonstration purposes is now displayed in the top control.

If you had chosen the SAPforms Data Plug-In for Workflow Container plug-in in step 8, the structure displayed would comprise the contents of the workflow container.

Define which form field is to be linked to which field in the SAP data structure (“Binding Definition”).

Choose a data field in the structure tree of the top control and drag it to the line in the table that contains the required control. If the binding was successful, the name of the SAP data field that is now linked to the control in the form appears in the Data Element column.

Generate the binding by choosing the File → Generate.

A dialog box with generation options is then displayed. Select the first two options Insert initialization code in Load()- method and button to trigger the insert action.

Choose OK.

Choose File → Exit to exit the main window of the SAPforms Designer.

You will then return to the Visual Basic development environment.

Choose Run → Start to start the form directly from the Visual Basic development environment.

The runtime version of the form is displayed. The controls are empty initially.

Enter your data in the fields.

Click the Submit button to send the form.

Result

Normally this would start the SAP Workflow functionality.

Since you chose the SAPforms Data Plug-In Demo in step 8, the form is running in Demo mode. This means that a window displaying the data that would normally be transferred to your R/3 System appears when you send the form.

If you had chosen the SAPforms Data Plug-In for Workflow Container in step 8, sending this form would have started a workflow. Since this would require extensive changes to settings in SAPforms and in the R/3 System, you should only use the Demo mode initially. The workflow examples [Seite 2258] demonstrate how a workflow is started.
Use

The programs WDFR3FH.EXE are linked to the file extension R3F when they are installed. These files are stored in the "\SAPforms\Bin" directory. They are configured via the SAPforms Helper. R3F (R/3 Form) is a previously unpublished file format that is used to transfer workflow information from an R/3 System to a form and vice versa. One of the two programs is launched each time you open a file with this extension. This happens, for example, when the appropriate attachment in a mail is opened or when the file is downloaded by the SAP frontend. The R/3 Release determines which of the two programs is started.

Integration

You must configure the SAPforms Helper application if you do not want to store your forms in the "\SAPforms\Forms" directory or in the active directory. When a work item is executed by an electronic form, the FORM.Process method supplies the name of the form to the SAPforms Helper application. This is the name you maintained when you defined the task. The SAPforms Helper application then searches for this form either in the active directory or in the "\SAPforms\Forms" directory.

Activities

To store your form files in a directory other than "\SAPforms\Forms", proceed as follows:

Start the SAPforms Helper.

The R3F-Interpreter Administration dialog box appears.

Enter the path where you want to store your forms in the Form Path field. Enter EXE in the Priority list field.

Choose OK.
SAP Trace Log Viewer

Use
The SAP Trace Log Viewer allows you to track any step in SAPforms to the level of detail required or to write it to a file. This helps you find runtime errors.

Prerequisites
The Trace Log Viewer is part of SAPforms. For this reason, you must install SAPforms [Seite 1725].

Features
When the SAP Trace Log Viewer is activated, all of the steps involving SAPforms are listed. You can display detailed information on a step by double-clicking it in the list. The functions of the SAP Trace Log Viewer are listed below:

<table>
<thead>
<tr>
<th>Menu</th>
<th>Function</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>File</strong></td>
<td>Open Trace File</td>
<td>Open existing trace file.</td>
</tr>
<tr>
<td></td>
<td>Save as</td>
<td>Save trace file.</td>
</tr>
<tr>
<td></td>
<td>Clear</td>
<td>Display or delete file.</td>
</tr>
<tr>
<td></td>
<td>Exit</td>
<td>Exit Trace Log Viewer</td>
</tr>
<tr>
<td><strong>Tools</strong></td>
<td>Options</td>
<td>Write Trace Log to:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Window (standard), <em>Trace Window</em></td>
</tr>
<tr>
<td></td>
<td></td>
<td>Temporary file, <em>File</em></td>
</tr>
<tr>
<td></td>
<td></td>
<td><em>File</em>, <em>File Name</em></td>
</tr>
<tr>
<td></td>
<td>Trace On</td>
<td>Level 0 (Trace off) - 3 (max. details)</td>
</tr>
<tr>
<td></td>
<td>Restore Default Settings</td>
<td>Standard window size</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Standard column width</td>
</tr>
<tr>
<td><strong>Filter</strong></td>
<td>Filter by Application</td>
<td>Display filter</td>
</tr>
<tr>
<td></td>
<td>Filter by Components</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Filter by Modules</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Filter by Procedure</td>
<td></td>
</tr>
<tr>
<td></td>
<td>No Filter</td>
<td></td>
</tr>
</tbody>
</table>

The *Clear*, *Open Trace File*, *Save as*, and the details setting in the *Trace* function can also be executed from the toolbar of the SAP Trace Log Viewer.

The toolbar also displays the file name in which the trace is stored.

You can move the columns to suit your requirements.
Maintaining Plug-Ins and Opening a Structure

Prerequisites
You have created a form with Visual Basic. You have launched the SAPforms Designer as a Visual Basic add-in and are in the main window of the SAPforms Designer. The form controls are shown in the table displayed in the bottom control.

The procedure described below is the same if you are using the Submit control or the stand-alone version of the SAPforms Designer.

Procedure

Maintain Plug-Ins
By maintaining the plug-ins, you define how the form communicates with the R/3 System.

Choose Tools → Options.
Select the tab page Plug-Ins.
In the SAP Data Plug-In box, enter the SAPData type plug-in for the R/3 interface you want to address.

The SAPforms Form Plug-Ins value is set by default according to the form development environment you are using. If you are using the stand-alone version of the SAPforms Designer, you must also maintain this plug-in.

In the SAP Transmit Plug-In box, enter the SAPXmit type plug-in for the communication method you want to use. Only the Plug-Ins of type SAPXmit which correspond to the SAPData Plug-In selected are available here.

Exit the dialog box.

The main window of the Designer is then displayed again.

The list below contains the plug-ins that are required for the various R/3 functions:

<table>
<thead>
<tr>
<th>R/3 function</th>
<th>Entry in the SAP Data Plug-In box.</th>
<th>Entry in the SAP Transmit Plug-In box.</th>
</tr>
</thead>
<tbody>
<tr>
<td>Starting tasks</td>
<td>SAPforms Data Plug-In for Workflow [Sie 1897]</td>
<td></td>
</tr>
<tr>
<td>Execute work items</td>
<td>SAPforms Transmit Plug-In for RFC [Sie 2125]</td>
<td></td>
</tr>
<tr>
<td></td>
<td>SAPforms Transmit Plug-In for MAPI [Sie 2146]</td>
<td></td>
</tr>
<tr>
<td></td>
<td>SAPforms Transmit Plug-In for SAPMAPI [Sie 2167]</td>
<td></td>
</tr>
<tr>
<td></td>
<td>SAPforms Transmit Plug-In for SMTP via WinSock [Sie 2188]</td>
<td></td>
</tr>
<tr>
<td>Calling a BAPI</td>
<td>SAPforms Data Plug-In for Function Modules [Sie 1954]</td>
<td>- No entry -</td>
</tr>
</tbody>
</table>
## Maintaining Plug-Ins and Opening a Structure

<table>
<thead>
<tr>
<th>Use an IDoc</th>
<th>SAPforms Data Plug-In for IDoc [Seite 2006]</th>
<th>SAPforms Transmit Plug-In for RFC [Seite 2125]</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>SAPforms Transmit Plug-In for MAPI [Seite 2146]</td>
</tr>
<tr>
<td></td>
<td></td>
<td>SAPforms Transmit Plug-In for SAPMAPI [Seite 2167]</td>
</tr>
</tbody>
</table>

### Opening a Structure

Choose *File* → *Open SAP Structures* and then the structure that you want to open.

A dialog box is then displayed prompting you to identify (for example, by the workflow number) the structure you want to start. For more information, see the detailed procedures.

- Maintaining Plug-Ins and Opening a Structure: BAPI [Seite 2262]
- Maintaining Plug-Ins and Opening a Structure: IDoc [Seite 2277]
- Maintaining Plug-Ins and Opening a Structure: Workflow [Seite 2254]

Identify the structure.

### Result

The interface description is exported from R/3 and displayed as a tree in the top control of the main SAPforms Designer window. You can now define the binding.

*Defining and generating a binding [Seite 2226]*
Plug-ins

Definition

Central components of SAPforms. SAPforms consists of several autonomous Visual Basic classes each of which is implemented by a plug-in. There are three types of plug-in, each of which contains a defined ActiveX interface.

The SAPforms Object Model

Use

All of the form and SAP-specific data operations within a form are carried out by the plug-ins. The SAPforms Designer uses plug-ins to communicate with the form and the R/3 System. From the perspective of the forms programmer, the plug-ins are objects whose properties and methods can be used to manipulate the forms.

See also programming forms [Seite 2228].

The tree plug-in types cover the following areas:

- Interfaces in the R/3 System: SAPData plug-in type [Seite 1895]
- Form systems: SAPForm plug-in type [Seite 1755]
- Communication method and data exchange with the R/3 System: SAPXmit plug-in type [Seite 2123]

SAPforms also provides the Submit control [Seite 1747] in addition to these plug-in types.
Submit Control

Use

The SAPforms Submit control allows you to use the SAPforms Designer [Seite 1735] in all form development environments that support ActiveX controls. You can also use the control in Visual Basic.

It also makes it easier for you to edit and generate electronic forms.

Integration

Using the Submit control is only one way to utilize the functions of the SAPforms Designer. You can also start the Designer as an add-in from Visual Basic or as a stand-alone version.

The Submit control in relation to the various plug-in types [Seite 1746].

Prerequisites

To work with the Submit control in Visual Basic, you must activate the corresponding ActiveX component. To do so, choose Project → Components and select the SAPforms Submit Control entry on the Controls tab.

To work with the Submit control in MS Outlook, you must also activate the corresponding ActiveX component first. Start the Toolbox in the form design environment of MS Outlook. Call the Additional Controls dialog box by choosing Custom Controls... from the context menu (right mouse click) of the Toolbox. Choose the SAPforms Submit Control entry here too.

You do not need to make any additional settings if you want to use the Submit control in MS Word or MS Excel. After you have installed SAPforms [Seite 1725], the control will be included in the Control toolbox (Other controls) under MS Word or MS Excel.
Features
The SAPforms Submit control contains all of the information required for communication between the
form and the R/3 System. Unlike working with the SAPforms Designer normally, the TAG property of the
form is no longer used to store this information in the form. Furthermore, no code is inserted in the form’s
load method.

Classification
Like every ActiveX control, the Submit control contains properties, methods, and results that can be used
to design a form. In addition to the typical properties for this control type (“pushbutton”), the following
properties and methods can be maintained:

Submit control
- AfterSubmit (method) [Seite 1750]
- BeforeSubmit (method) [Seite 1751]
- Custom (property) [Seite 1752]
- ErrorMessage (property) [Seite 1753]
- SuccessMessage (property) [Seite 1754]

Activities
Working with the Submit control entails the following steps:

Start SAPforms Designer
Create your form.
Insert the submit control into your form.
Open the context menu of the control (right mouse click).
Choose Properties.
Now choose SAPforms Designer from the dialog box displayed.
The SAPforms Designer is launched.

Generating a Binding
There are two ways of defining the binding between the controls on the form and the R/3 data fields.
Choose File → Generate and Exit to exit the SAPforms Designer. The dialog box from where the
SAPforms-Designer was started is displayed. Here, choose Use data binding if supported.
This allows you to define the binding via the properties of the individual controls.

Choose the DataSource SapformsSubmit property.

Using the DataMember and DataField properties, choose the R/3 System
data fields (from the open interface structure) that you want to
link with the corresponding form control.

This procedure is suitable if you want to define a large number of bindings and therefore want to make this definition in the development environment for forms and not in the SAPforms Designer.
Define the binding within the SAPforms Designer using "drag and drop". Then choose File → Generate. The binding is then generated in the usual way [Seite 2226].

You can then compile the form in Visual basic and convert it to an EXE file or publish it in a folder in MS Outlook.
AfterSubmit (Method)

Definition
Inserts an action that takes place directly after the Submit event.

Remarks

Classification
AfterSubmit

← Submit control [Seite 1747]
BeforeSubmit (Method)

Definition
Inserts an action that takes place directly before the Submit event.

Remarks
It could be useful, for example, to program an error handling routine [Seite 2233] before the Submit event.

Classification
BeforeSubmit

⬅ Submit control [Seite 1747]
Custom (Property)

Definition
Calls the context menu of the Submit control, which can be used to launch the SAPforms Designer.

Remarks
When using the Submit control, you will usually launch the SAPforms Designer by choosing Define from the context menu of the control (right mouse click).
See also
Designing forms with MS Outlook [Seite 1731]
Designing forms with MS Word and MS Excel [Seite 1733]

Classification
Custom

↩ Submit control [Seite 1747]
ErrorMessage (Property)

Definition
Error message text that can be used in a message box.

Remarks
You can assign this property a text that is displayed in a message box each time an error occurs when the form data is transferred to the R/3 System.

Classification
ErrorMessage

←Submit control [Seite 1747]
**SuccessMessage (Property)**

**Definition**
Text that can be used in a *message box*.

**Remarks**
You can assign this property a text that is displayed in a *message box* each time the form data is successfully transferred to the R/3 System.

**Classification**
SuccessMessage

←Submit control [Seite 1747]
**SAPForm Plug-In Type**

**Definition**
This plug-in type is provided for each form type that is to be supported by the SAPforms Designer. The main purpose of this plug-in is to populate its own controls with the controls contained in the original form. It provides access to controls and specific features of the form through a form-independent interface.

The plug-in type SAPform in relation to the various plug-in types [Seite 1746].

**Use**
The SAPforms Designer [Seite 1735] is shipped with the following plug-ins of this type:

<table>
<thead>
<tr>
<th>Plug-In</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>SAPforms Form Plug-In for Visual Basic [Seite 1756]</td>
<td>Plug-in for Visual Basic forms. The source code for this plug-in is supplied on the CD and serves as an example of how plug-ins can be implemented for other form systems.</td>
</tr>
<tr>
<td>SAPforms Form Plug-In for MS Outlook [Seite 1796]</td>
<td>Plug-in for MS Outlook forms.</td>
</tr>
<tr>
<td>SAPforms Form Plug-In for MS Office 97 [Seite 1856]</td>
<td>Plug-in for forms designed with MS Word and MS Excel.</td>
</tr>
</tbody>
</table>
SAPforms Form Plug-In for Visual Basic

Definition
Form plug-in that implements an interface between Visual Basic forms and SAPforms.

Use
This plug-in implements the SAPforms interface that provides the SAPforms Designer [Seite 1735] and other SAPforms components with generic access to the properties and controls of a certain form type (Visual Basic, MS Outlook, MS Office). In other words, this plug-in can be regarded as being synonymous with the associated SAPforms class. Since the source code is supplied, you can include support for your own control types. If you want to map a SAP data element on a group of option button, you associate the data element to each of the desired option buttons. You must also assign a value to the Tag property of each option button. This value is assigned to the data element when the corresponding option button is selected.

Classification
SAPforms Form Plug-In for Visual Basic

→ Link class [Seite 1857]
→ Links class [Seite 1862]
→ SAPCtrl class [Seite 1767]
→ SAPform class [Seite 1772]
Link Class

Definition
Represents a binding between a generic form control and an R/3 data field.

Remarks
A form control may be linked to several data fields in an SAPData object. All such bindings of a form control are contained in the Links list in the SAPCtrl class. The bindings are established when the form is designed by means of the SAPForms Designer and are stored in the Save method of the SAPForm class. Furthermore, they are reconstructed in the Load method of the SAPForm class every time the form is loaded and are used by the PutData and GetData methods to populate the SAP fields with the form fields and vice versa.
The form control referenced in the link is identified by means of its display name. The data field referenced in the link is identified by means of the field qualifier and optionally by a field selector plus selector value.
Links are m:n relationships, that is, a form field may be linked to several data fields and a certain data field may be linked to several form fields.
The Link class is only required for developing plug-ins.

Classification
SAPforms Form Plug-In for MS Office [Seite 1856]
SAPforms Form Plug-In for Visual Basic [Seite 1756]
SAPforms Form Plug-In for MS Outlook [Seite 1796]

⇒Link Class

⇒Load (Method) [Seite 2213]
Load (Method)

Definition
Loads structure information to SAPforms objects.

Remarks
Classification 1
Load

Properties
Loads the property values of an SAPData object from a structured string. As a result, the status of the SAPData object that was previously saved permanently with the Save method is restored. This saves the additional overheads that would result if the structure information had to be loaded each time from the original source. This method is usually called within the Load method of the SAPForm class.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>From</td>
<td>Character string containing the structure information</td>
</tr>
</tbody>
</table>

Classification 2
Load

Properties
Loads the property values of a field object from a structured string. As a result, the status of the field object that was previously saved permanently with the Save method is restored. For this method to be called successfully, the next character at the Pos position must be keyword “Fld”.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>From</td>
<td>Structured string</td>
</tr>
<tr>
<td>Pos</td>
<td>Current position within the string. Should point to the number of remaining properties to read</td>
</tr>
</tbody>
</table>

Classification 3
Load
Properties

Loads the property values of a fields object from a structured string. As a result, the status of the fields object that was previously saved permanently with the Save method is restored.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>From</td>
<td>Character string that provides the structure information.</td>
</tr>
<tr>
<td>Pos</td>
<td>Current search position within the character string.</td>
</tr>
</tbody>
</table>

Classification 4

Load

Properties

Reconstructs the segment object from a structured string. Counterpart of the Save method.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>From</td>
<td>Character string containing structure information.</td>
</tr>
<tr>
<td>Pos</td>
<td>Current search position within the character string.</td>
</tr>
</tbody>
</table>

Classification 5

Load

Properties

This method reconstructs the segment list from a structured string. Counterpart of the Save method.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>From</td>
<td>Character string containing structure information.</td>
</tr>
<tr>
<td>Pos</td>
<td>Current search position within the character string.</td>
</tr>
</tbody>
</table>

Classification 6

Load

Properties

This method reconstructs an object of the Link class from the serialized representation. It assumes that the next token is located at the Pos position in the serialized representation of the “Fld” keyword.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>From</td>
<td>Serialization string.</td>
</tr>
</tbody>
</table>
Load (Method)

Current position within the character string.

Classification 7

Load

- SAPForm class [Seite 1772]
- SAPForms Form Plug-In for Visual Basic [Seite 1756]
- SAPForm class [Seite 1812]
- SAPForms Form Plug-In for MS Outlook [Seite 1796]
- SAPForm class [Seite 1872]
- SAPForms Form Plug-In for MS Office [Seite 1856]

Properties

Method for loading form controls into Controls Collection from a form or form template.

Note that the form information is not stored in the Tag property of this object, but reconstructed each time the form is loaded. This method is called at design time and at runtime. It has to populate the Controls collection for both cases. At runtime, the method is called by the user code in the form's Load method.

When the form is designed, the method is called by the SAPForms Designer or from outside SAPForms. If you are using the SAPForms Designer as a stand-alone application, the Load method is called with Mode=2. This instructs the Load method to load the form from a file. If you are using the SAPForms Designer as a plug-in, the Load method is invoked from the hosting environment and receives the proprietary form description as a parameter. The populated SAPForm object should be passed to the SAPForms Designer by the PlugIn method of the Designer class.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Frm</td>
<td>form or form template containing controls.</td>
</tr>
<tr>
<td>Mode</td>
<td>0=design time (add-in), 1=design time (file), 2=runtime.</td>
</tr>
</tbody>
</table>

Classification 8

Load

- SAPXmit class [Seite 2106]
- SAPForms Data Plug-In for Demo [Seite 2077]
- SAPXmit class [Seite 1837]
- SAPForms Form Plug-In for Outlook [Seite 1796]
- SAPXmit class [Seite 2126]
- SAPForms Transmit Plug-In for RFC [Seite 2125]
- SAPXmit class [Seite 2168]
- SAPForms Transmit Plug-In for MAPI [Seite 2146]
- SAPXmit class [Seite 2147]
- SAPForms Transmit Plug-In for SAPMAPI [Seite 2167]
- SAPXmit class [Seite 2189]
- SAPForms Transmit Plug-In for WinSock [Seite 2188]
- SAPXmit class [Seite 2209]
- SAPForms Transmit Plug-In for Demo [Seite 2208]
Properties
Loads the property values of an SAPXmit object from a structured string. As a result, the status of the SAPXmit object that was previously saved permanently with the Save method is restored.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>From</td>
<td>Character string containing the structure information</td>
</tr>
</tbody>
</table>
Links Class

Definition
Collection of all the bindings in a form containing an object of the Link class.

Remarks
The content of this list is stored in serialized form in the Tag property of the form. The list is reconstructed from this property in the Load method of the SAPforms object.

Classification
SAPforms Form Plug-In for MS Office [Seite 1856]
SAPforms Form Plug-In for Visual Basic [Seite 1756]
SAPforms Form Plug-In for MS Outlook [Seite 1796]

→Links Class
→Save (Method) [Seite 2219]
Save (Method)

Definition
Saves the updated form definition.

Remarks
This method is used by the SAPforms Designer when a form is designed to save the modified form definition after the binding has been defined. The modifications can affect the binding definition or possibly the source code entered in the form.

Classification 1
Save

<table>
<thead>
<tr>
<th>Classification</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Classification 1</td>
<td>Description</td>
</tr>
<tr>
<td>Save</td>
<td></td>
</tr>
</tbody>
</table>

Properties
Saves the current property values of an SAPData object in a structured string. Counterpart of the Load method.
When a form is created, the SAPforms Designer uses this method to save the structure description retrieved from the R/3 System permanently in the form.
The R/3 structure is restored from the structured string at runtime by the Load method.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Tgt</td>
<td>String in which the status is stored.</td>
</tr>
</tbody>
</table>

Classification 2
Save

Field Class [Seite 2007]
SAPforms Data Plug-In for Workflow [Seite 1897]
SAPforms Data Plug-In for Function Modules [Seite 1954]
SAPforms Data Plug-In for IDoc [Seite 2006]
SAPData class [Seite 2077]

Properties
Saves the current property values of an SAPData object in a structured string. Counterpart of the Load method. The string is saved with a form so that the structure information is available when the form is executed.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Tgt</td>
<td>Stores the string structure.</td>
</tr>
</tbody>
</table>
Classification 3

Save

Properties
Loads the property values of a fields object from a structured string. As a result, the status of the fields object that was previously saved permanently with the Save method is restored. The string is saved with the form so that the structure information is available when the form is executed.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Tgt</td>
<td>Stores the string structure.</td>
</tr>
</tbody>
</table>

Classification 4

Save

Properties
This method saves the structure of a segment, including the field list, in a structured string.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Tgt</td>
<td>Stores the string structure.</td>
</tr>
</tbody>
</table>

Classification 5

Save

Properties
This method saves the segment list in a structured string. Counterpart of the Load method.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Tgt</td>
<td>Stores the string structure.</td>
</tr>
</tbody>
</table>

Classification 6

Save

Properties
Properties
Method that writes the content of all links to a file. Counterpart of the Load method.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Tgt</td>
<td>Stores the string structure.</td>
</tr>
</tbody>
</table>

Classification 7
Save

- SAPForm class [Seite 1772]
- SAPforms Form Plug-In for Visual Basic [Seite 1756]
- SAPForm class [Seite 1872]
- SAPforms Form Plug-In for MS Office [Seite 1856]
- SAPForm class [Seite 1812]
- SAPforms Form Plug-In for MS Outlook [Seite 1796]

Properties
Method that saves a form to a file if it was created from a file.
Reads the old file and writes updated lines to the new file.

Classification 8
Save

- SAPXmit class [Seite 2106]
- SAPforms Data Plug-In for Demo [Seite 2077]
- SAPXmit class [Seite 1837]
- SAPforms Form Plug-In for Outlook [Seite 1796]
- SAPXmit class [Seite 2126]
- SAPforms Transmit Plug-In for RFC [Seite 2125]
- SAPXmit class [Seite 2168]
- SAPforms Transmit Plug-In for MAPI [Seite 2146]
- SAPXmit class [Seite 2147]
- SAPforms Transmit Plug-In for SAPMAPI [Seite 2167]
- SAPXmit class [Seite 2189]
- SAPforms Transmit Plug-In for WinSock [Seite 2188]
- SAPXmit class [Seite 2209]
- SAPforms Transmit Plug-In for Demo [Seite 2208]

Properties
Saves the current property values of an SAPXmit object in a structured string. Counterpart of the Load method.
When a form is created, the SAPforms Designer uses this method to save the structure description retrieved from the R/3 System permanently in the form.
The R/3 structure is restored from the structured string at runtime by the Load method.
## Parameter Description

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Tgt</td>
<td>String in which the status is stored.</td>
</tr>
</tbody>
</table>
SAPCtrl Class

Definition
This class represents a generic form control that is contained in the Controls list of the SAPForm class.

Remarks
Implements a dependent object of the SAPForm interface for Visual Basic forms. The Controls list of the SAPForm class consists of objects of this type. An object in the SAPCtrl class usually represents a form control. The table controls are an exception here. With table controls, an object of the SAPCtrl class is added to the Controls list for each column. In the case of control fields, each element of a field is added to the Controls list as if it were a separate control.

This interface provides the SAPForms Designer and other SAPForms components with generic access to form properties and controls. The concept requires that each object that is a possible binding target be interpreted as a separate generic control. With VB, this means that a separate control is to be created for every element of a control array and for every grid column.

Classification

SAPforms Form Plug-In for Visual Basic [Seite 1756]

→ SAPCtrl Class

→ Name (property) [Seite 2088]

→ Value (property) [Seite 2103]
Name (Property)

Definition
Name of the object, field, segment, or control.

Remarks
This name, for example, must be used as the root in the SAP structure tree that is built by the BuildTree method.

Classification 1
Name

Properties
Returns the name of the workflow to whose task the container description belongs.

Classification 2
Name

Properties
Name of the R/3 function module that is to be executed.

Classification 3
Name

Properties
Name of the IDoc as it appears in the control record.

Classification 4
Name

Classification 5
Name

[References]
Properties
Name of the field.

Classification 6

Name

← Segment class [Seite 2055]
← SAPforms Data Plug-In for IDoc [Seite 2006]

Properties
Name of the segment.
Used as a key in the Templates list of the IDoc object. Only the first attribute with this name is added to the list of segments with a key. All other segment attributes with the same name are added without a key.

Classification 7

Name

← SAPForm class [Seite 1772]
← SAPforms Form Plug-In for Visual Basic [Seite 1756]
← SAPForm Class [Seite 1812]
← SAPforms Form Plug-In for MS Outlook [Seite 1796]
← SAPForm class [Seite 1872]
← SAPforms Form Plug-In for MS Office [Seite 1856]

Properties
Original name of the control in the real form (e.g. Visual Basic form).
Value (Property)

Definition
Value of a specific field type.

Remarks

Classification 1
Value

Properties
The value of an SAP data field, defined in greater detail by the Qual parameter. The structure and syntax of the Qual parameter depends on the concrete SAP data structure from which this field originates. In the simplest case, the Qual parameter designates the name of a field whose value is to be set or read. If the field is part of a structure (for example, an IDoc segment), Qual must identify the field uniquely within the overall structure. If the structure is repeated within an instance - as is the case with some IDocs - Qual must also contain a selector that identifies the IDoc segment uniquely.

```
obj.Value("Field1") = "abc".
Obj.Value("Seg1.Field1") = "abc".
X = obj.Value("Seg1.Seg2.Field1").
X = obj.Value("Seg1.Field1 Seg2.Field2 = '123'" ) ' selector.
```

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Qual</td>
<td>Qualifier and selector that uniquely identifies a particular field.</td>
</tr>
</tbody>
</table>

Classification 2
Value


Properties

Returns the value of control.
This property is an abstraction of the different properties used by the original controls to present their content, for example, as Text, Caption, Value, and so on. The implementation of the SAPCtrl must map the original control property in the Value property for each control that is to be supported by SAPforms.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>X</td>
<td>Index (e.g. for grids).</td>
</tr>
<tr>
<td>Return</td>
<td>Value</td>
</tr>
</tbody>
</table>
SAPForm Class

Definition
Original class of this plug-in.

Remarks
This class implements the SAPForm interface for Visual Basic forms, making these forms available to the SAPForms Designer and other components of SAPForms. The Label, TextBox, ComboBox, OptionButton, CheckBox, ListBox, SAPTableView, Grid, as well as MSFlexGrid controls and even control arrays are supported.

The classes of the SAPForm plug-in type implement the SAPforms interface that provides the SAPforms Designer and other SAPforms components with generic access to the properties and controls of a certain form type (Visual Basic, MS Outlook, MS Office). In other words, these classes can be regarded as being synonymous with various plug-ins of this type.

The SAPForm object is used as an anchor to all other SAPforms objects. The SAPData property, for example, contains a reference to the SAPData object used in the form, while the SAPXmit property references the SAPXmit object (see below) used.

If you want to develop your own SAPForm-type plug-ins, you should use the source code supplied for this plug-in as a reference (template). The template is stored in the directory ...\sapforms\sapfvb4.

```vbnet
Option Explicit

Dim objSAPForm As Object

Private Sub Form_Load()
    Set objSAPForm = New SAPForm
    Set objSAPForm.SAPData = New SAPData
    Set objSAPForm.SAPXmit = New SAPXmit
End Sub
```

Classification
SAPforms Form Plug-In for Visual Basic [Seite 1756]

→ SAPForm Class

→ Execute (Method) [Seite 2210]
→ GetData (Method) [Seite 1876]
→ Load (Method) [Seite 2213]
→ LogIt (Method) [Seite 1931]
→ Options (Method) [Seite 2217]
→ PutData (Method) [Seite 1884]
→ SAPData (property) [Seite 1885]
→ SAPXmit (property) [Seite 2093]
→ Save (Method) [Seite 2219]
→ Start (Method) [Seite 2223]
→ Submit (Method) [Seite 1894]
Execute (Method)

Definition
Method that calls the Execute function in the R/3 System. The semantics of this function depend on the SAPData object.

Remarks
The method can be called directly via the SAPData object or indirectly via the Start method of the SAPForm object, whose SAPData property is a reference to the SAPData object.

In the case of the SAPData object for workflow containers, for example, this method executes a work item in an SAP Business Workflow and populates its container elements with values from the form fields.

Classification 1
Execute

Properties
Terminates a work item in the R/3 System and supplies its container elements with the data values in the fields list.

Classification 2
Execute

Properties
Does not execute an action in the R/3 System but rather displays the data of the object in a separate window.

Classification 3
Execute

Properties
Invokes the Execute method in the associated Xmit or SAPData object. The semantics of this method depend on this object.


**Classification 4**

**Execute**

- SAPXmit class [Seite 2126]
- SAPforms Transmit Plug-In for RFC [Seite 2125]

**Properties**

This method executes a work item synchronously via a remote function call (RFC). Supports SAPData object of type *Container*.

This method causes a certain function module to be executed in the R/3 System. This module receives a table with a complete R3F structure containing the work item information and the container data that are to be supplied from the form.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Data</td>
<td>Object representing the container data.</td>
</tr>
</tbody>
</table>

**Classification 5**

**Execute**

- SAPXmit class [Seite 2168]
- SAPforms Transmit Plug-In for MAPI [Seite 2146]
- SAPXmit class [Seite 2189]
- SAPforms Transmit Plug-In for WinSock [Seite 2188]

**Properties**

Sends a mail to the R/3 System that contains the information required to terminate a work item (including container data) as an attachment.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Data</td>
<td>Object representing the container data.</td>
</tr>
</tbody>
</table>

**Classification 6**

**Execute**

- SAPXmit class [Seite 2147]
- SAPforms Transmit Plug-In for SAPMAPI [Seite 2167]

**Properties**

Overwrites the R3F attachment, thereby updating it.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Pdata</td>
<td>Object that represents the <em>mobjData</em> container data</td>
</tr>
</tbody>
</table>

**Classification 7**

**Execute**

- SAPXmit class [Seite 2209]
- SAPforms Transmit Plug-In for Demo [Seite 2208]

**Properties**

Displays the instance data of the associated SAPData object in a window instead of sending it to the R/3 System.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Data</td>
<td>object of class SAPData</td>
</tr>
</tbody>
</table>
GetData (Method)

Definition
Transfers the contents of the SAPData object to the form.

Remarks
This method evaluates the link chain associated with each control in order to make the transfer. The method is called from the code contained in the Load method of a form.

Classification
GetData

← SAPForm class [Seite 1772]
← SAPforms Form Plug-In for Visual Basic [Seite 1756]
← SAPForm class [Seite 1812]
← SAPforms Form Plug-In for MS Outlook [Seite 1796]
← SAPForm class [Seite 1872]
← SAPforms Form Plug-In for MS Office [Seite 1856]
Load (Method)

Definition
Loads structure information to SAPforms objects.

Remarks

Classification 1

Load

Properties
Loads the property values of an SAPData object from a structured string. As a result, the status of the SAPData object that was previously saved permanently with the Save method is restored. This saves the additional overheads that would result if the structure information had to be loaded each time from the original source. This method is usually called within the Load method of the SAPForm class.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>From</td>
<td>Character string containing the structure information</td>
</tr>
</tbody>
</table>

Classification 2

Load

Properties
Loads the property values of a field object from a structured string. As a result, the status of the field object that was previously saved permanently with the Save method is restored. For this method to be called successfully, the next character at the Pos position must be keyword “Fld”.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>From</td>
<td>Structured string</td>
</tr>
<tr>
<td>Pos</td>
<td>Current position within the string. Should point to the number of remaining properties to read</td>
</tr>
</tbody>
</table>

Classification 3

Load
Properties
Loads the property values of a fields object from a structured string. As a result, the status of the fields object that was previously saved permanently with the Save method is restored.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>From</td>
<td>Character string that provides the structure information.</td>
</tr>
<tr>
<td>Pos</td>
<td>Current search position within the character string.</td>
</tr>
</tbody>
</table>

Classification 4
Load

Properties
Reconstructs the segment object from a structured string. Counterpart of the Save method.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>From</td>
<td>Character string containing structure information.</td>
</tr>
<tr>
<td>Pos</td>
<td>Current search position within the character string.</td>
</tr>
</tbody>
</table>

Classification 5
Load

Properties
This method reconstructs the segment list from a structured string. Counterpart of the Save method.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>From</td>
<td>Character string containing structure information.</td>
</tr>
<tr>
<td>Pos</td>
<td>Current search position within the character string.</td>
</tr>
</tbody>
</table>

Classification 6
Load

Properties
This method reconstructs an object of the Link class from the serialized representation. It assumes that the next token is located at the Pos position in the serialized representation of the “Fld” keyword.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>From</td>
<td>Serialization string.</td>
</tr>
</tbody>
</table>
Load (Method)

Current position within the character string.

**Classification 7**

**Load**

- SAPForm class [Seite 1772]
- SAPforms Form Plug-In for Visual Basic [Seite 1756]
- SAPForm class [Seite 1812]
- SAPforms Form Plug-In for MS Outlook [Seite 1796]
- SAPForm class [Seite 1872]
- SAPforms Form Plug-In for MS Office [Seite 1856]

**Properties**

Method for loading form controls into Controls Collection from a form or form template. Note that the form information is not stored in the Tag property of this object, but reconstructed each time the form is loaded. This method is called at design time and at runtime. It has to populate the Controls collection for both cases. At runtime, the method is called by the user code in the form's Load method. When the form is designed, the method is called by the SAPForms Designer or from outside SAPForms. If you are using the SAPForms Designer as a stand-alone application, the Load method is called with Mode=2. This instructs the Load method to load the form from a file. If you are using the SAPForms Designer as a plug-in, the Load method is invoked from the hosting environment and receives the proprietary form description as a parameter. The populated SAPForm object should be passed to the SAPForms Designer by the Plugln method of the Designer class.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Frm</td>
<td>form or form template containing controls.</td>
</tr>
<tr>
<td>Mode</td>
<td>0=design time (add-in), 1=design time (file), 2=runtime.</td>
</tr>
</tbody>
</table>

**Classification 8**

**Load**

- SAPXmit class [Seite 2106]
- SAPforms Data Plug-In for Demo [Seite 2077]
- SAPXmit class [Seite 1837]
- SAPforms Form Plug-In for Outlook [Seite 1796]
- SAPXmit class [Seite 2126]
- SAPforms Transmit Plug-In for RFC [Seite 2125]
- SAPXmit class [Seite 2168]
- SAPforms Transmit Plug-In for MAPI [Seite 2146]
- SAPXmit class [Seite 2147]
- SAPforms Transmit Plug-In for SAPMAPI [Seite 2167]
- SAPXmit class [Seite 2189]
- SAPforms Transmit Plug-In for WinSock [Seite 2188]
- SAPXmit class [Seite 2209]
- SAPforms Transmit Plug-In for Demo [Seite 2208]
Load (Method)

Properties
Loads the property values of an SAPXmit object from a structured string. As a result, the status of the SAPXmit object that was previously saved permanently with the Save method is restored.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>From</td>
<td>Character string containing the structure information</td>
</tr>
</tbody>
</table>
LogIt (Method)

Definition
Allows form-specific data to be written to the Trace Log Viewer [Seite 1742].

Remarks

Classification

LogIt

← SAPData [Seite 1918]
← SAPforms Data Plug-In for Workflow [Seite 1897]
← SAPForm [Seite 1772]
← SAPforms Form Plug-In for Visual Basic [Seite 1756]
← SAPForm [Seite 1812]
← SAPforms Form Plug-In for MS Outlook [Seite 1796]
← SAPForm [Seite 1872]
← SAPforms Form Plug-In for MS Office [Seite 1856]
Options (Method)

Definition
Method that outputs the object-specific options dialog box.

Remarks
This dialog box displays either global settings or settings that are valid for the current form. These settings can be changed here. The global settings are read from the registry before they are displayed and are then saved again to the registry.

The method is called within the SAPforms Designer if the required plug-in is active under Properties on the Plug-Ins tab in the Options dialog. You can also use the sample program Options.exe to display the Options dialog of a certain plug-in.

Classification 1
Options

- SAPData class [Seite 1918]
- SAPforms Data Plug-In for Workflow [Seite 1897]
- SAPData class [Seite 1975]
- SAPforms Data Plug-In for Function Modules [Seite 1954]
- SAPData class [Seite 2027]
- SAPforms Data Plug-In for IDoc [Seite 2006]
- SAPData class [Seite 2078]
- SAPforms Data Plug-In for Demo [Seite 2077]

Classification 2
Options

- SAPForm class [Seite 1772]
- SAPforms Form Plug-In for Visual Basic [Seite 1756]
- SAPForm class [Seite 1812]
- SAPforms Form Plug-In for MS Outlook [Seite 1796]
- SAPForm class [Seite 1872]
- SAPforms Form Plug-In for MS Office [Seite 1856]

Classification 3
Options

- SAPXmit class [Seite 2106]
- SAPforms Data Plug-In for Demo [Seite 2077]
- SAPXmit class [Seite 1837]
- SAPforms Form Plug-In for MS Outlook [Seite 1796]
- SAPXmit class [Seite 2126]
Options (Method)

- SAPforms Transmit Plug-In for RFC [Seite 2125]
- SAPXmit class [Seite 2168]
- SAPforms Transmit Plug-In for MAPI [Seite 2146]
- SAPXmit class [Seite 2147]
- SAPforms Transmit Plug-In for SAPMAPI [Seite 2167]
- SAPXmit class [Seite 2189]
- SAPforms Transmit Plug-In for SMTP [Seite 2188]
- SAPXmit class [Seite 2209]
- SAPforms Transmit Plug-In for Demo [Seite 2208]
PutData (Method)

Definition
Transfers the contents of the form to the SAPData object.

Remarks
This method evaluates the link chain associated with each control in order to make the transfer. After the method has been executed, the SAPData object is populated with instance data and can be transferred to the R/3 System.
This method is usually called by the Start or Execute method within an event procedure of a form.

Classification
PutData
← SAPForm class [Seite 1772]
← SAPforms Form Plug-In for Visual Basic [Seite 1756]
← SAPForm class [Seite 1812]
← SAPforms Form Plug-In for MS Outlook [Seite 1796]
← SAPForm class [Seite 1872]
← SAPforms Form Plug-In for MS Office [Seite 1856]
**SAPData (Property)**

**Definition**
Contains a reference to the SAPData object used in the form.

**Remarks**
The SAPData object is reconstructed at runtime by the form's Load method and then made available via this property.

**Classification**
SAPData

- SAPForm class [Seite 1772]
  - SAPforms Form Plug-In for Visual Basic [Seite 1756]
- SAPForm class [Seite 1812]
  - SAPforms Form Plug-In for MS Outlook [Seite 1796]
- SAPForm class [Seite 1872]
  - SAPforms Form Plug-In for MS Office [Seite 1856]
SAPXmit (Property)

Definition
Reference to the SAPXmit object that is used to encapsulate the transport of SAPData to the R/3 System.

Remarks
Classification

SAPXmit

- SAPData class [Seite 1918]
- SAPforms Data Plug-In for Workflow [Seite 1897]
- SAPData class [Seite 1975]
- SAPforms Data Plug-In for Function Modules [Seite 1954]
- SAPData class [Seite 2027]
- SAPforms Data Plug-In for IDoc [Seite 2006]
- SAPData class [Seite 2078]
- SAPforms Data Plug-In for Demo [Seite 2077]
- SAPForm class [Seite 1772]
- SAPforms Form Plug-In for Visual Basic [Seite 1756]
- SAPForm class [Seite 1812]
- SAPforms Form Plug-In for MS Outlook [Seite 1796]
- SAPForm class [Seite 1872]
- SAPforms Form Plug-In for MS Office [Seite 1856]
Save (Method)

Definition
Saves the updated form definition.

Remarks
This method is used by the SAPforms Designer when a form is designed to save the modified form definition after the binding has been defined. The modifications can affect the binding definition or possibly the source code entered in the form.

Classification 1
Save

Properties
Saves the current property values of an SAPData object in a structured string. Counterpart of the Load method.

When a form is created, the SAPforms Designer uses this method to save the structure description retrieved from the R/3 System permanently in the form.

The R/3 structure is restored from the structured string at runtime by the Load method.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Tgt</td>
<td>String in which the status is stored.</td>
</tr>
</tbody>
</table>

Classification 2
Save

Properties
Saves the current property values of an SAPData object in a structured string. Counterpart of the Load method. The string is saved with a form so that the structure information is available when the form is executed.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Tgt</td>
<td>Stores the string structure.</td>
</tr>
</tbody>
</table>
Classification 3

Save

Properties

Loads the property values of a fields object from a structured string. As a result, the status of the fields object that was previously saved permanently with the Save method is restored.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Tgt</td>
<td>Stores the string structure.</td>
</tr>
</tbody>
</table>

Classification 4

Save

Properties

This method saves the structure of a segment, including the field list, in a structured string.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Tgt</td>
<td>Stores the string structure.</td>
</tr>
</tbody>
</table>

Classification 5

Save

Properties

This method saves the segment list in a structured string. Counterpart of the Load method.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Tgt</td>
<td>Stores the string structure.</td>
</tr>
</tbody>
</table>

Classification 6

Save

Properties
Properties
Method that writes the content of all links to a file. Counterpart of the Load method.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Tgt</td>
<td>Stores the string structure.</td>
</tr>
</tbody>
</table>

Classification 7

Save

- SAPForm class [Seite 1772]
- SAPforms Form Plug-In for Visual Basic [Seite 1756]
- SAPForm class [Seite 1872]
- SAPforms Form Plug-In for MS Office [Seite 1856]
- SAPForm class [Seite 1812]
- SAPforms Form Plug-In for MS Outlook [Seite 1796]

Properties
Method that saves a form to a file if it was created from a file.
Reads the old file and writes updated lines to the new file.

Classification 8

Save

- SAPXmit class [Seite 2106]
- SAPforms Data Plug-In for Demo [Seite 2077]
- SAPXmit class [Seite 1837]
- SAPforms Form Plug-In for Outlook [Seite 1796]
- SAPXmit class [Seite 2126]
- SAPforms Transmit Plug-In for RFC [Seite 2125]
- SAPXmit class [Seite 2168]
- SAPforms Transmit Plug-In for MAPI [Seite 2146]
- SAPXmit class [Seite 2147]
- SAPforms Transmit Plug-In for SAPMAPI [Seite 2167]
- SAPXmit class [Seite 2189]
- SAPforms Transmit Plug-In for WinSock [Seite 2188]
- SAPXmit class [Seite 2209]
- SAPforms Transmit Plug-In for Demo [Seite 2208]

Properties
Saves the current property values of an SAPXmit object in a structured string. Counterpart of the Load method.
When a form is created, the SAPforms Designer uses this method to save the structure description retrieved from the R/3 System permanently in the form.
The R/3 structure is restored from the structured string at runtime by the Load method.
<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Tgt</td>
<td>String in which the status is stored.</td>
</tr>
</tbody>
</table>
Start (Method)

Definition
Method that executes the Start function in R/3. The semantics of this function depend on the concrete SAPData object.

Remarks
In the case of the SAPData object for workflow containers, this method starts a workflow and populates its container elements with values from the form fields. This method can be called directly via the SAPData object or indirectly via the Start method of the SAPForm object, whose SAPData property is a reference to the SAPData object. If an SAPXmit object is assigned to the SAPData object, the Start method of the SAPXmit object is called.

Classification 1
Start

Properties
Starts a workflow in the R/3 System and supplies its container elements with the data values in the fields list.

Classification 2
Start

Properties
This method carries out the current R/3 function module call.

Classification 3
Start

Properties
Generates an IDoc file from the data in the segments list and transfers this IDoc to the R/3 System. The transfer method depends on the assigned SAPXmit object. A workflow can be started in the R/3 System when the inbound IDoc is processed.

Classification 4
Start
Properties
Does not execute an action in the R/3 System but rather displays the data of the object in a separate window.

Classification 5
Start

Properties
Invokes the Start method in the associated SAPXmit or SAPData object. The semantics of this method depend on this SAPData object. If this object is the SAPData object for SAP Business Workflow, this method starts a workflow in an R/3 System.

Classification 6
Start

Properties
This method starts a workflow synchronously via RFC. Supports SAPData objects of types "Idoc" and "Container".

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Data</td>
<td>Object of type SAPData.</td>
</tr>
</tbody>
</table>

Classification 7
Start

Properties
Sends a mail to the R/3 System with an attachment containing the information required to start a workflow.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Data</td>
<td>Object of type SAPData.</td>
</tr>
</tbody>
</table>
Classification 8

Start

- SAPXmit class [Seite 2209]
- SAPforms Transmit Plug-In for Demo [Seite 2208]
- SAPXmit class [Seite 2106]
- SAPforms Data Plug-In for Demo [Seite 2077]

Properties
Displays the instance data of the associated SAPData object in a window instead of sending it to the R/3 System.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Data</td>
<td>Object of type SAPData.</td>
</tr>
</tbody>
</table>
Submit (Method)

Definition
Calls the Start() or Execute() method, depending on the internal status of the form.

Remarks
This status must indicate whether the form was called with or without input data. If it was called with input data, the Execute() method is called. This logic is specially designed for integration in SAP Business Workflow. The Execute() method here executes a work item in the R/3 System.

Classification
Submit

← SAPForm class [Seite 1772]
← SAPforms Form Plug-In for Visual Basic [Seite 1756]
← SAPForm class [Seite 1872]
← SAPforms Form Plug-In for MS Office [Seite 1856]
← SAPForm class [Seite 1812]
← SAPforms Form Plug-In for MS Outlook [Seite 1796]
SAPforms Form Plug-In for MS Outlook

Definition
Form plug-in that implements an interface between MS Outlook forms and SAPforms.

Use
You use this form whenever you want to process an electronic form designed with MS Outlook [Seite 1731]. You must set this plug-in if you want to use the stand-alone version of the SAPforms Designer [Seite 1735]. If you use the Submit control [Seite 1747] in MS Outlook, the plug-in is set automatically. This plug-in implements the SAPforms interface that provides the SAPforms Designer and other SAPforms components with generic access to the properties and controls of a certain form type (Visual Basic, MS Outlook, MS Office). In other words, this plug-in can be regarded as being synonymous with the associated SAPforms class.

Classification
SAPforms Form Plug-In for MS Outlook
- Link class [Seite 1857]
- Links class [Seite 1862]
- SAPCtrl class [Seite 1807]
- SAPform class [Seite 1812]
- SAPXmit class [Seite 1837]
Link Class

Definition
Represents a binding between a generic form control and an R/3 data field.

Remarks
A form control may be linked to several data fields in an SAPData object. All such bindings of a form control are contained in the *Links* list in the SAPCtrl class. The bindings are established when the form is designed by means of the SAPForms Designer and are stored in the Save method of the SAPForm class. Furthermore, they are reconstructed in the Load method of the SAPForm class every time the form is loaded and are used by the PutData and GetData methods to populate the SAP fields with the form fields and vice versa.

The form control referenced in the link is identified by means of its display name. The data field referenced in the link is identified by means of the field qualifier and optionally by a field selector plus selector value.

Links are m:n relationships, that is, a form field may be linked to several data fields and a certain data field may be linked to several form fields.

The Link class is only required for developing plug-ins.

Classification
- SAPforms Form Plug-In for MS Office [Seite 1856]
- SAPforms Form Plug-In for Visual Basic [Seite 1756]
- SAPforms Form Plug-In for MS Outlook [Seite 1796]

→ Link Class

→ Load (Method) [Seite 2213]
Load (Method)

Definition
Loads structure information to SAPforms objects.

Remarks

Classification 1
Load

Properties
Loads the property values of an SAPData object from a structured string. As a result, the status of the
SAPData object that was previously saved permanently with the Save method is restored.
This saves the additional overheads that would result if the structure information had to be loaded each
time from the original source. This method is usually called within the Load method of the SAPForm class.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>From</td>
<td>Character string containing the structure information</td>
</tr>
</tbody>
</table>

Classification 2
Load

Properties
Loads the property values of a field object from a structured string. As a result, the status of the field object
that was previously saved permanently with the Save method is restored.
For this method to be called successfully, the next character at the Pos position must be keyword “Fld”.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>From</td>
<td>Structured string</td>
</tr>
</tbody>
</table>
| Pos       | Current position within the string. Should point to the number of
remaining properties to read |

Classification 3
Load
Load (Method)

Properties
Loads the property values of a fields object from a structured string. As a result, the status of the fields object that was previously saved permanently with the Save method is restored.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>From</td>
<td>Character string that provides the structure information.</td>
</tr>
<tr>
<td>Pos</td>
<td>Current search position within the character string.</td>
</tr>
</tbody>
</table>

Classification 4
Load

Properties
Reconstructs the segment object from a structured string. Counterpart of the Save method.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>From</td>
<td>Character string containing structure information.</td>
</tr>
<tr>
<td>Pos</td>
<td>Current search position within the character string.</td>
</tr>
</tbody>
</table>

Classification 5
Load

Properties
This method reconstructs the segment list from a structured string. Counterpart of the Save method.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>From</td>
<td>Character string containing structure information.</td>
</tr>
<tr>
<td>Pos</td>
<td>Current search position within the character string.</td>
</tr>
</tbody>
</table>

Classification 6
Load

Properties
This method reconstructs an object of the Link class from the serialized representation. It assumes that the next token is located at the Pos position in the serialized representation of the “Fld” keyword.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>From</td>
<td>Serialization string.</td>
</tr>
</tbody>
</table>
Load (Method)

| Pos | Current position within the character string. |

Classification 7

Load

- `SAPForm class [Seite 1772]`
- `SAPforms Form Plug-In for Visual Basic [Seite 1756]`
- `SAPForm class [Seite 1812]`
- `SAPforms Form Plug-In for MS Outlook [Seite 1796]`
- `SAPForm class [Seite 1872]`
- `SAPforms Form Plug-In for MS Office [Seite 1856]`

Properties

Method for loading form controls into Controls Collection from a form or form template.

Note that the form information is not stored in the Tag property of this object, but reconstructed each time the form is loaded. This method is called at design time and at runtime. It has to populate the Controls collection for both cases. At runtime, the method is called by the user code in the form's Load method. When the form is designed, the method is called by the SAPForms Designer or from outside SAPForms. If you are using the SAPForms Designer as a stand-alone application, the Load method is called with Mode=2. This instructs the Load method to load the form from a file. If you are using the SAPForms Designer as a plug-in, the Load method is invoked from the hosting environment and receives the proprietary form description as a parameter. The populated SAPForm object should be passed to the SAPForms Designer by the PlugIn method of the Designer class.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Frm</td>
<td>form or form template containing controls.</td>
</tr>
<tr>
<td>Mode</td>
<td>0=design time (add-in), 1=design time (file), 2=runtime.</td>
</tr>
</tbody>
</table>

Classification 8

Load

- `SAPXmit class [Seite 2106]`
- `SAPforms Data Plug-In for Demo [Seite 2077]`
- `SAPXmit class [Seite 1837]`
- `SAPforms Form Plug-In for Outlook [Seite 1796]`
- `SAPXmit class [Seite 2126]`
- `SAPforms Transmit Plug-In for RFC [Seite 2125]`
- `SAPXmit class [Seite 2168]`
- `SAPforms Transmit Plug-In for MAPI [Seite 2146]`
- `SAPXmit class [Seite 2147]`
- `SAPforms Transmit Plug-In for SAPMAPI [Seite 2167]`
- `SAPXmit class [Seite 2189]`
- `SAPforms Transmit Plug-In for WinSock [Seite 2188]`
- `SAPXmit class [Seite 2209]`
- `SAPforms Transmit Plug-In for Demo [Seite 2208]`
Properties

Loads the property values of an SAPXmit object from a structured string. As a result, the status of the SAPXmit object that was previously saved permanently with the Save method is restored.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>From</td>
<td>Character string containing the structure information</td>
</tr>
</tbody>
</table>
Links Class

Definition
Collection of all the bindings in a form containing an object of the Link class.

Remarks
The content of this list is stored in serialized form in the Tag property of the form. The list is reconstructed from this property in the Load method of the SAPforms object.

Classification
SAPforms Form Plug-In for MS Office [Seite 1856]
SAPforms Form Plug-In for Visual Basic [Seite 1756]
SAPforms Form Plug-In for MS Outlook [Seite 1796]

⇒ Links Class
⇒ Save (Method) [Seite 2219]
Save (Method)

Definition
Saves the updated form definition.

Remarks
This method is used by the SAPforms Designer when a form is designed to save the modified form definition after the binding has been defined. The modifications can affect the binding definition or possibly the source code entered in the form.

Classification 1
Save

Properties
Saves the current property values of an SAPData object in a structured string. Counterpart of the Load method.
When a form is created, the SAPforms Designer uses this method to save the structure description retrieved from the R/3 System permanently in the form.
The R/3 structure is restored from the structured string at runtime by the Load method.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Tgt</td>
<td>String in which the status is stored.</td>
</tr>
</tbody>
</table>

Classification 2
Save

Properties
Saves the current property values of an SAPData object in a structured string. Counterpart of the Load method. The string is saved with a form so that the structure information is available when the form is executed.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Tgt</td>
<td>Stores the string structure.</td>
</tr>
</tbody>
</table>
Classification 3

Save

- Fields Class [Seite 2018]
- SAPforms Data Plug-In for Workflow [Seite 1897]
- SAPforms Data Plug-In for Function Modules [Seite 1954]
- SAPforms Data Plug-In for IDoc [Seite 2006]

Properties

Loads the property values of a fields object from a structured string. As a result, the status of the fields object that was previously saved permanently with the Save method is restored. The string is saved with the form so that the structure information is available when the form is executed.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Tgt</td>
<td>Stores the string structure.</td>
</tr>
</tbody>
</table>

Classification 4

Save

- Segment class [Seite 2055]
- SAPforms Data Plug-In for IDoc [Seite 2006]

Properties

This method saves the structure of a segment, including the field list, in a structured string.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Tgt</td>
<td>Stores the string structure.</td>
</tr>
</tbody>
</table>

Classification 5

Save

- Segments class [Seite 2067]
- SAPforms Data Plug-In for IDoc [Seite 2006]

Properties

This method saves the segment list in a structured string. Counterpart of the Load method.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Tgt</td>
<td>Stores the string structure.</td>
</tr>
</tbody>
</table>

Classification 6

Save

- Links class [Seite 1862]
- SAPforms Form Plug-In for Visual Basic [Seite 1756]
- Links class [Seite 1862]
- SAPforms Form Plug-In for MS Office [Seite 1856]
- Links class [Seite 1862]
- SAPforms Form Plug-In for MS Outlook [Seite 1796]
Properties
Method that writes the content of all links to a file. Counterpart of the Load method.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Tgt</td>
<td>Stores the string structure.</td>
</tr>
</tbody>
</table>

Classification 7
Save

- SAPForm class [Seite 1772]
- SAPforms Form Plug-In for Visual Basic [Seite 1756]
- SAPForm class [Seite 1872]
- SAPforms Form Plug-In for MS Office [Seite 1856]
- SAPForm class [Seite 1812]
- SAPforms Form Plug-In for MS Outlook [Seite 1796]

Properties
Method that saves a form to a file if it was created from a file. Reads the old file and writes updated lines to the new file.

Classification 8
Save

- SAPXmit class [Seite 2106]
- SAPforms Data Plug-In for Demo [Seite 2077]
- SAPXmit class [Seite 1837]
- SAPforms Form Plug-In for Outlook [Seite 1796]
- SAPXmit class [Seite 2126]
- SAPforms Transmit Plug-In for RFC [Seite 2125]
- SAPXmit class [Seite 2168]
- SAPforms Transmit Plug-In for MAPI [Seite 2146]
- SAPXmit class [Seite 2147]
- SAPforms Transmit Plug-In for SAPMAPI [Seite 2167]
- SAPXmit class [Seite 2189]
- SAPforms Transmit Plug-In for WinSock [Seite 2188]
- SAPXmit class [Seite 2209]
- SAPforms Transmit Plug-In for Demo [Seite 2208]

Properties
Saves the current property values of an SAPXmit object in a structured string. Counterpart of the Load method.
When a form is created, the SAPforms Designer uses this method to save the structure description retrieved from the R/3 System permanently in the form.
The R/3 structure is restored from the structured string at runtime by the Load method.
<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Tgt</td>
<td>String in which the status is stored.</td>
</tr>
</tbody>
</table>
SAPCtrl Class

Definition
This class implements a generic form control that is contained in the Controls list of the SAPForm class.

Remarks
Represents a dependent object of the SAPform interface for Visual Basic forms. The Controls list of the SAPForm class consists of objects of this type. An object in the SAPCtrl class usually represents a form control. The table controls are an exception here. With table controls, an object of the SAPCtrl class is added to the Controls list for each column. In the case of control fields, each element of a field is added to the Controls list as if it were a separate control.
This interface provides the SAPForms Designer and other SAPForms components with generic access to form properties and controls. The concept requires that each object that is a possible binding target be interpreted as a separate generic control. With VB, this means that a separate control is to be created for every element of a control array and for every grid column.

Classification
SAPforms Form Plug-In for MS Outlook [Seite 1796]

SAPCtrl Class

Name (Property) [Seite 2088]

Value (Property) [Seite 2103]
**Name (Property)**

**Definition**
Name of the object, field, segment, or control.

**Remarks**
This name, for example, must be used as the root in the SAP structure tree that is built by the `BuildTree` method.

**Classification 1**

**Name**

- SAPData class [Seite 1918]
- SAPforms Data Plug-In for Workflow [Seite 1897]

**Properties**
Returns the name of the workflow to whose task the container description belongs.

**Classification 2**

**Name**

- SAPData class [Seite 1975]
- SAPforms Data Plug-In for Function Modules [Seite 1954]

**Properties**
Name of the R/3 function module that is to be executed.

**Classification 3**

**Name**

- SAPData class [Seite 2027]
- SAPforms Data Plug-In for IDoc [Seite 2006]

**Properties**
Name of the IDoc as it appears in the control record.

**Classification 4**

**Name**

- SAPData class [Seite 2078]
- SAPforms Data Plug-In for Demo [Seite 2077]

**Classification 5**

**Name**

- Field Class [Seite 2007]
- SAPforms Data Plug-In for Workflow [Seite 1897]
- SAPforms Data Plug-In for Function Modules [Seite 1954]
- SAPforms Data Plug-In for IDoc [Seite 2006]
Properties
Name of the field.

Classification 6
Name
- Segment class [Seite 2055]
- SAPforms Data Plug-In for IDoc [Seite 2006]

Properties
Name of the segment.
Used as a key in the Templates list of the IDoc object. Only the first attribute with this name is added to the list of segments with a key. All other segment attributes with the same name are added without a key.

Classification 7
Name
- SAPForm class [Seite 1772]
- SAPforms Form Plug-In for Visual Basic [Seite 1756]
- SAPForm Class [Seite 1812]
- SAPforms Form Plug-In for MS Outlook [Seite 1796]
- SAPForm class [Seite 1872]
- SAPforms Form Plug-In for MS Office [Seite 1856]

Properties
Original name of the control in the real form (e.g. Visual Basic form).
Value (Property)

Definition
Value of a specific field type.

Remarks

Classification 1

Value

Properties
The value of an SAP data field, defined in greater detail by the \textit{Qual} parameter.
The structure and syntax of the \textit{Qual} parameter depends on the concrete SAP data structure from which this field originates. In the simplest case, the \textit{Qual} parameter designates the name of a field whose value is to be set or read. If the field is part of a structure (for example, an IDoc segment), \textit{Qual} must identify the field uniquely within the overall structure. If the structure is repeated within an instance - as is the case with some IDocs - \textit{Qual} must also contain a selector that identifies the IDoc segment uniquely.

\begin{verbatim}
obj.Value("Field1") = "abc".
obj.Value("Seg1.Field1") = "abc".
X = obj.Value("Seg1.Seg2.Field1").
X = obj.Value("Seg1.Field1 Seg2.Field2 = '123'" ) ' selector.
\end{verbatim}

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Qual</td>
<td>Qualifier and selector that uniquely identifies a particular field.</td>
</tr>
</tbody>
</table>

Classification 2

Value

Properties

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Qual</td>
<td>Qualifier and selector that uniquely identifies a particular field.</td>
</tr>
</tbody>
</table>
Properties

Returns the value of control. This property is an abstraction of the different properties used by the original controls to present their content, for example, as Text, Caption, Value, and so on. The implementation of the SAPCtrl must map the original control property in the Value property for each control that is to be supported by SAPforms.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>X</td>
<td>Index (e.g. for grids).</td>
</tr>
<tr>
<td>Return</td>
<td>Value</td>
</tr>
</tbody>
</table>
**SAPForm Class**

**Definition**
Original class of this plug-in.

**Remarks**
This class implements the SAPForm interface for MS Outlook forms and makes these forms available to the SAPForms Designer and other components of SAPForms. The Label, TextBox, ComboBox, OptionButton, CheckBox, ListBox, SAPTableView, Grid, as well as MSFlexGrid controls and even control arrays are supported.

The classes of the SAPForm plug-in type implement the SAPforms interface that provides the SAPforms Designer and other SAPforms components with generic access to the properties and controls of a certain form type (Visual Basic, MS Outlook, MS Office). In other words, these classes can be regarded as being synonymous with various plug-ins of this type.

The SAPForm object is used as an anchor to all other SAPforms objects. The SAPData property, for example, contains a reference to the SAPData object used in the form, while the SAPXmit property references the SAPXmit object (see below) used.

If you want to develop your own SAPForm-type plug-ins, you should use the source code supplied for plug-ins for Visual Basic as a reference (template). The template is stored in the directory ...\sapforms\sapfvb4.

**Classification**
SAPforms Form Plug-In for MS Outlook [Seite 1796]

→ **SAPForm Class**

→ AddCbControl (Method) [Seite 1813]
→ Execute (Method) [Seite 2210]
→ GetData (Method) [Seite 1876]
→ Load (Method) [Seite 2213]
→ LogIt (Method) [Seite 1931]
→ Options (Method) [Seite 2217]
→ PutData (Method) [Seite 1884]
→ SAPData (Property) [Seite 1885]
→ SAPXmit (Property) [Seite 2093]
→ Save (Method) [Seite 2219]
→ Start (Method) [Seite 2223]
→ Submit (Method) [Seite 1894]
→ UpdateInstance (Method) [Seite 2102]
AddCbControl (Method)

Definition
Adds an SAP “Send” button to the command line of an Outlook item.

Remarks

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>CbType</td>
<td>Type of command line</td>
</tr>
<tr>
<td>Cb</td>
<td>Command line to which the “Send” button is added</td>
</tr>
<tr>
<td>OnAction</td>
<td>Subroutine that is executed on Click</td>
</tr>
<tr>
<td>Caption</td>
<td>Name of button</td>
</tr>
<tr>
<td>Pos1</td>
<td>Position in main menu</td>
</tr>
<tr>
<td>Pos2</td>
<td>Position in submenu</td>
</tr>
<tr>
<td>Picture</td>
<td>Button image</td>
</tr>
<tr>
<td>Return</td>
<td>The created “button” object</td>
</tr>
</tbody>
</table>

Classification
AddCbControl

← SAPForm class [Seite 1812]
← SAPforms Form Plug-In for MS Outlook [Seite 1796]
Execute (Method)

Definition
Method that calls the Execute function in the R/3 System. The semantics of this function depend on the SAPData object.

Remarks
The method can be called directly via the SAPData object or indirectly via the Start method of the SAPForm object, whose SAPData property is a reference to the SAPData object.

Classification 1
Execute

Properties
Terminates a work item in the R/3 System and supplies its container elements with the data values in the fields list.

Classification 2
Execute

Properties
Does not execute an action in the R/3 System but rather displays the data of the object in a separate window.

Classification 3
Execute
**Properties**
Invokes the Execute method in the associated Xmit or SAPData object. The semantics of this method depend on this object.

**Classification 4**
**Execute**

- SAPXmit class [Seite 2126]
- SAPforms Transmit Plug-In for RFC [Seite 2125]

**Properties**
This method executes a work item synchronously via a remote function call (RFC). Supports SAPData object of type *Container*.
This method causes a certain function module to be executed in the R/3 System. This module receives a table with a complete R3F structure containing the work item information and the container data that are to be supplied from the form.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Data</td>
<td>Object representing the container data.</td>
</tr>
</tbody>
</table>

**Classification 5**
**Execute**

- SAPXmit class [Seite 2168]
- SAPforms Transmit Plug-In for MAPI [Seite 2146]
- SAPXmit class [Seite 2189]
- SAPforms Transmit Plug-In for WinSock [Seite 2188]

**Properties**
Sends a mail to the R/3 System that contains the information required to terminate a work item (including container data) as an attachment.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Data</td>
<td>Object representing the container data.</td>
</tr>
</tbody>
</table>

**Classification 6**
**Execute**

- SAPXmit class [Seite 2147]
- SAPforms Transmit Plug-In for SAPMAPI [Seite 2167]

**Properties**
Overwrites the R3F attachment, thereby updating it.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Pdata</td>
<td>Object that represents the mobjData container data</td>
</tr>
</tbody>
</table>

**Classification 7**
**Execute**

- SAPXmit class [Seite 2209]
- SAPforms Transmit Plug-In for Demo [Seite 2208]
Properties
Displays the instance data of the associated SAPData object in a window instead of sending it to the R/3 System.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Data</td>
<td>object of class SAPData</td>
</tr>
</tbody>
</table>
GetData (Method)

Definition
Transfers the contents of the SAPData object to the form.

Remarks
This method evaluates the link chain associated with each control in order to make the transfer. The method is called from the code contained in the Load method of a form.

Classification
GetData

← SAPForm class [Seite 1772]
← SAPforms Form Plug-In for Visual Basic [Seite 1756]
← SAPForm class [Seite 1812]
← SAPforms Form Plug-In for MS Outlook [Seite 1796]
← SAPForm class [Seite 1872]
← SAPforms Form Plug-In for MS Office [Seite 1856]
Load (Method)

Definition
Loads structure information to SAPforms objects.

Remarks
Classification 1
Load

<table>
<thead>
<tr>
<th>From Character string containing the structure information</th>
</tr>
</thead>
</table>

Classification 2
Load

<table>
<thead>
<tr>
<th>From Structured string.</th>
</tr>
</thead>
</table>

Properties
Loads the property values of an SAPData object from a structured string. As a result, the status of the SAPData object that was previously saved permanently with the Save method is restored. This saves the additional overheads that would result if the structure information had to be loaded each time from the original source. This method is usually called within the Load method of the SAPForm class.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>From</td>
<td>Character string containing the structure information</td>
</tr>
</tbody>
</table>

Classification 3
Load

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>From</td>
<td>Structured string.</td>
</tr>
<tr>
<td>Pos</td>
<td>Current position within the string. Should point to the number of remaining properties to read.</td>
</tr>
</tbody>
</table>
Properties

Loads the property values of a fields object from a structured string. As a result, the status of the fields object that was previously saved permanently with the Save method is restored.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>From</td>
<td>Character string that provides the structure information.</td>
</tr>
<tr>
<td>Pos</td>
<td>Current search position within the character string.</td>
</tr>
</tbody>
</table>

Classification 4

Load

Properties

Reconstructs the segment object from a structured string. Counterpart of the Save method.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>From</td>
<td>Character string containing structure information.</td>
</tr>
<tr>
<td>Pos</td>
<td>Current search position within the character string.</td>
</tr>
</tbody>
</table>

Classification 5

Load

Properties

This method reconstructs the segment list from a structured string. Counterpart of the Save method.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>From</td>
<td>Character string containing structure information.</td>
</tr>
<tr>
<td>Pos</td>
<td>Current search position within the character string.</td>
</tr>
</tbody>
</table>

Classification 6

Load

Properties

This method reconstructs an object of the Link class from the serialized representation. It assumes that the next token is located at the Pos position in the serialized representation of the “Fld” keyword.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>From</td>
<td>Serialization string.</td>
</tr>
</tbody>
</table>
Load (Method)

| Pos | Current position within the character string. |

**Classification 7**

**Load**

- SAPForm class [Seite 1772]
- SAPForms Form Plug-In for Visual Basic [Seite 1756]
- SAPForm class [Seite 1812]
- SAPForms Form Plug-In for MS Outlook [Seite 1796]
- SAPForm class [Seite 1872]
- SAPForms Form Plug-In for MS Office [Seite 1856]

**Properties**

Method for loading form controls into Controls Collection from a form or form template. Note that the form information is not stored in the Tag property of this object, but reconstructed each time the form is loaded. This method is called at design time and at runtime. It has to populate the Controls collection for both cases. At runtime, the method is called by the user code in the form's Load method. When the form is designed, the method is called by the SAPForms Designer or from outside SAPForms. If you are using the SAPForms Designer as a stand-alone application, the Load method is called with Mode=2. This instructs the Load method to load the form from a file. If you are using the SAPForms Designer as a plug-in, the Load method is invoked from the hosting environment and receives the proprietary form description as a parameter. The populated SAPForm object should be passed to the SAPForms Designer by the PlugIn method of the Designer class.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Frm</td>
<td>form or form template containing controls.</td>
</tr>
<tr>
<td>Mode</td>
<td>0=design time (add-in), 1=design time (file), 2=runtime.</td>
</tr>
</tbody>
</table>

**Classification 8**

**Load**

- SAPXmit class [Seite 2106]
- SAPForms Data Plug-In for Demo [Seite 2077]
- SAPXmit class [Seite 1837]
- SAPForms Form Plug-In for Outlook [Seite 1796]
- SAPXmit class [Seite 2126]
- SAPForms Transmit Plug-In for RFC [Seite 2125]
- SAPXmit class [Seite 2168]
- SAPForms Transmit Plug-In for MAPI [Seite 2146]
- SAPXmit class [Seite 2147]
- SAPForms Transmit Plug-In for SAPMAPI [Seite 2167]
- SAPXmit class [Seite 2189]
- SAPForms Transmit Plug-In for WinSock [Seite 2188]
- SAPXmit class [Seite 2209]
- SAPForms Transmit Plug-In for Demo [Seite 2208]
Load (Method)

Properties
Loads the property values of an SAPXmit object from a structured string. As a result, the status of the SAPXmit object that was previously saved permanently with the Save method is restored.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>From</td>
<td>Character string containing the structure information</td>
</tr>
</tbody>
</table>
LogIt (Method)

Definition
Allows form-specific data to be written to the Trace Log Viewer [Seite 1742].

Remarks

Classification

LogIt

← SAPData [Seite 1918]
← SAPforms Data Plug-In for Workflow [Seite 1897]
← SAPForm [Seite 1772]
← SAPforms Form Plug-In for Visual Basic [Seite 1756]
← SAPForm [Seite 1812]
← SAPforms Form Plug-In for MS Outlook [Seite 1796]
← SAPForm [Seite 1872]
← SAPforms Form Plug-In for MS Office [Seite 1856]
Options (Method)

Definition
Method that outputs the object-specific options dialog box.

Remarks
This dialog box displays either global settings or settings that are valid for the current form. These settings can be changed here. The global settings are read from the registry before they are displayed and are then saved again to the registry.
The method is called within the SAPforms Designer if the required plug-in is active under Properties on the Plug-Ins tab in the Options dialog. You can also use the sample program Options.exe to display the Options dialog of a certain plug-in.

Classification 1
Options

- SAPData class [Seite 1918]
- SAPforms Data Plug-In for Workflow [Seite 1897]
- SAPData class [Seite 1975]
- SAPForms Data Plug-In for Function Modules [Seite 1954]
- SAPData class [Seite 2027]
- SAPForms Data Plug-In for IDoc [Seite 2006]
- SAPData class [Seite 2078]
- SAPForms Data Plug-In for Demo [Seite 2077]

Classification 2
Options

- SAPForm class [Seite 1772]
- SAPforms Form Plug-In for Visual Basic [Seite 1756]
- SAPForm class [Seite 1812]
- SAPforms Form Plug-In for MS Outlook [Seite 1796]
- SAPForm class [Seite 1872]
- SAPforms Form Plug-In for MS Office [Seite 1856]

Classification 3
Options

- SAPXmit class [Seite 2106]
- SAPforms Data Plug-In for Demo [Seite 2077]
- SAPXmit class [Seite 1837]
- SAPforms Form Plug-In for MS Outlook [Seite 1796]
- SAPXmit class [Seite 2126]
SAAPforms Transmit Plug-In for RFC [Seite 2125]
SAAPXmit class [Seite 2168]
SAAPforms Transmit Plug-In for MAPI [Seite 2146]
SAAPXmit class [Seite 2147]
SAAPforms Transmit Plug-In for SAPMAPI [Seite 2167]
SAAPXmit class [Seite 2189]
SAAPforms Transmit Plug-In for SMTP [Seite 2188]
SAAPXmit class [Seite 2209]
SAAPforms Transmit Plug-In for Demo [Seite 2208]
PutData (Method)

Definition
Transfers the contents of the form to the SAPData object.

Remarks
This method evaluates the link chain associated with each control in order to make the transfer. After the method has been executed, the SAPData object is populated with instance data and can be transferred to the R/3 System.

This method is usually called by the Start or Execute method within an event procedure of a form.

Classification

PutData

➥ SAPForm class [Seite 1772]
➥ SAPforms Form Plug-In for Visual Basic [Seite 1756]
➥ SAPForm class [Seite 1812]
➥ SAPforms Form Plug-In for MS Outlook [Seite 1796]
➥ SAPForm class [Seite 1872]
➥ SAPforms Form Plug-In for MS Office [Seite 1856]
SAPData (Property)

Definition
Contains a reference to the SAPData object used in the form.

Remarks
The SAPData object is reconstructed at runtime by the form's Load method and then made available via this property.

Classification
SAPData

← SAPForm class [Seite 1772]
← SAPforms Form Plug-In for Visual Basic [Seite 1756]
← SAPForm class [Seite 1812]
← SAPforms Form Plug-In for MS Outlook [Seite 1796]
← SAPForm class [Seite 1872]
← SAPforms Form Plug-In for MS Office [Seite 1856]
SAPXmit (Property)

Definition
Reference to the SAPXmit object that is used to encapsulate the transport of SAPData to the R/3 System.

Remarks
Classification
SAPXmit

⇒ SAPData class [Seite 1918]
  ⇒ SAPforms Data Plug-In for Workflow [Seite 1897]
  ⇒ SAPforms Data Plug-In for Function Modules [Seite 1954]
  ⇒ SAPforms Data Plug-In for IDoc [Seite 2006]
  ⇒ SAPforms Data Plug-In for Demo [Seite 2077]
  ⇒ SAPforms Form Plug-In for Visual Basic [Seite 1756]
  ⇒ SAPforms Form Plug-In for MS Outlook [Seite 1796]
  ⇒ SAPforms Form Plug-In for MS Office [Seite 1856]
Save (Method)

Definition
Saves the updated form definition.

Remarks
This method is used by the SAPforms Designer when a form is designed to save the modified form definition after the binding has been defined. The modifications can affect the binding definition or possibly the source code entered in the form.

Classification 1
Save

Properties
Saves the current property values of an SAPData object in a structured string. Counterpart of the Load method.

When a form is created, the SAPforms Designer uses this method to save the structure description retrieved from the R/3 System permanently in the form.

The R/3 structure is restored from the structured string at runtime by the Load method.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Tgt</td>
<td>String in which the status is stored.</td>
</tr>
</tbody>
</table>

Classification 2
Save

Properties
Saves the current property values of an SAPData object in a structured string. Counterpart of the Load method. The string is saved with a form so that the structure information is available when the form is executed.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Tgt</td>
<td>Stores the string structure.</td>
</tr>
</tbody>
</table>
Classification 3

Save

Launcher Class [Seite 2018]
SAPforms Data Plug-In for Workflow [Seite 1897]
SAPforms Data Plug-In for Function Modules [Seite 1954]
SAPforms Data Plug-In for IDoc [Seite 2006]

Properties

Loads the property values of a fields object from a structured string. As a result, the status of the fields object that was previously saved permanently with the Save method is restored.
The string is saved with the form so that the structure information is available when the form is executed.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Tgt</td>
<td>Stores the string structure.</td>
</tr>
</tbody>
</table>

Classification 4

Save

Launcher class [Seite 2055]
SAPforms Data Plug-In for IDoc [Seite 2006]

Properties

This method saves the structure of a segment, including the field list, in a structured string.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Tgt</td>
<td>Stores the string structure.</td>
</tr>
</tbody>
</table>

Classification 5

Save

Launcher class [Seite 2067]
SAPforms Data Plug-In for IDoc [Seite 2006]

Properties

This method saves the segment list in a structured string. Counterpart of the Load method.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Tgt</td>
<td>Stores the string structure.</td>
</tr>
</tbody>
</table>

Classification 6

Save

Launcher class [Seite 1862]
SAPforms Form Plug-In for Visual Basic [Seite 1756]
Launcher class [Seite 1862]
SAPforms Form Plug-In for MS Office [Seite 1856]
Launcher class [Seite 1862]
SAPforms Form Plug-In for MS Outlook [Seite 1796]
Properties
Method that writes the content of all links to a file. Counterpart of the Load method.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Tgt</td>
<td>Stores the string structure.</td>
</tr>
</tbody>
</table>

Classification 7
Save

- SAPForm class [Seite 1772]
- SAPforms Form Plug-In for Visual Basic [Seite 1756]
- SAPForm class [Seite 1872]
- SAPforms Form Plug-In for MS Office [Seite 1856]
- SAPForm class [Seite 1812]
- SAPforms Form Plug-In for MS Outlook [Seite 1796]

Properties
Method that saves a form to a file if it was created from a file.
Reads the old file and writes updated lines to the new file.

Classification 8
Save

- SAPXmit class [Seite 2106]
- SAPforms Data Plug-In for Demo [Seite 2077]
- SAPXmit class [Seite 1837]
- SAPforms Form Plug-In for Outlook [Seite 1796]
- SAPXmit class [Seite 2126]
- SAPforms Transmit Plug-In for RFC [Seite 2125]
- SAPXmit class [Seite 2168]
- SAPforms Transmit Plug-In for MAPI [Seite 2146]
- SAPXmit class [Seite 2147]
- SAPforms Transmit Plug-In for SAPMAPI [Seite 2167]
- SAPXmit class [Seite 2189]
- SAPforms Transmit Plug-In for WinSock [Seite 2188]
- SAPXmit class [Seite 2209]
- SAPforms Transmit Plug-In for Demo [Seite 2208]

Properties
Saves the current property values of an SAPXmit object in a structured string. Counterpart of the Load method.
When a form is created, the SAPforms Designer uses this method to save the structure description retrieved from the R/3 System permanently in the form.
The R/3 structure is restored from the structured string at runtime by the Load method.
## Save (Method)

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Tgt</td>
<td>String in which the status is stored.</td>
</tr>
</tbody>
</table>
Start (Method)

Definition
Method that executes the Start function in R/3. The semantics of this function depend on the concrete SAPData object.

Remarks
In the case of the SAPData object for workflow containers, this method starts a workflow and populates its container elements with values from the form fields. This method can be called directly via the SAPData object or indirectly via the Start method of the SAPForm object, whose SAPData property is a reference to the SAPData object. If an SAPXmit object is assigned to the SAPData object, the Start method of the SAPXmit object is called.

Classification 1
Start

Properties
Starts a workflow in the R/3 System and supplies its container elements with the data values in the fields list.

Classification 2
Start

Properties
This method carries out the current R/3 function module call.

Classification 3
Start

Properties
Generates an IDoc file from the data in the segments list and transfers this IDoc to the R/3 System. The transfer method depends on the assigned SAPXmit object. A workflow can be started in the R/3 System when the inbound IDoc is processed.

Classification 4
Start
Properties
Does not execute an action in the R/3 System but rather displays the data of the object in a separate window.

Classification 5
Start

\[\text{SAPForm class [Seite 1772]}\]
\[\text{SAPforms Form Plug-In for Visual Basic [Seite 1756]}\]
\[\text{SAPForm class [Seite 1872]}\]
\[\text{SAPforms Form Plug-In for MS Office [Seite 1856]}\]
\[\text{SAPForm class [Seite 1812]}\]
\[\text{SAPforms Form Plug-In for MS Outlook [Seite 1796]}\]

Properties
Invokes the \textit{Start} method in the associated SAPXmit or SAPData object. The semantics of this method depend on this SAPData object. If this object is the \textit{SAPData object for SAP Business Workflow}, this method starts a workflow in an R/3 System.

Classification 6
Start

\[\text{SAPXmit class [Seite 2126]}\]
\[\text{SAPforms Transmit Plug-In for RFC [Seite 2125]}\]

Properties
This method starts a workflow synchronously via RFC. Supports SAPData objects of types "Idoc" and "Container".

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Data</td>
<td>Object of type SAPData.</td>
</tr>
</tbody>
</table>

Classification 7
Start

\[\text{SAPXmit class [Seite 2168]}\]
\[\text{SAPforms Transmit Plug-In for MAPI [Seite 2146]}\]
\[\text{SAPXmit class [Seite 2147]}\]
\[\text{SAPforms Transmit Plug-In for SAPMAPI [Seite 2167]}\]
\[\text{SAPXmit class [Seite 2189]}\]
\[\text{SAPforms Transmit Plug-In for SMTP [Seite 2188]}\]
\[\text{SAPXmit class [Seite 1837]}\]
\[\text{SAPforms Form Plug-In for MS Outlook [Seite 1796]}\]

Properties
Sends a mail to the R/3 System with an attachment containing the information required to start a workflow.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Data</td>
<td>Object of type SAPData.</td>
</tr>
</tbody>
</table>
Classification 8

Start

- SAPXmit class [Seite 2209]
- SAPforms Transmit Plug-In for Demo [Seite 2208]
- SAPXmit class [Seite 2106]
- SAPforms Data Plug-In for Demo [Seite 2077]

Properties

Displays the instance data of the associated SAPData object in a window instead of sending it to the R/3 System.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Data</td>
<td>Object of type SAPData.</td>
</tr>
</tbody>
</table>
Submit (Method)

Definition
Calls the Start() or Execute() method, depending on the internal status of the form.

Remarks
This status must indicate whether the form was called with or without input data. If it was called with input data, the Execute() method is called. This logic is specially designed for integration in SAP Business Workflow. The Execute() method here executes a work item in the R/3 System.

Classification
Submit

← SAPForm class [Seite 1772]
← SAPforms Form Plug-In for Visual Basic [Seite 1756]
← SAPForm class [Seite 1872]
← SAPforms Form Plug-In for MS Office [Seite 1856]
← SAPForm class [Seite 1812]
← SAPforms Form Plug-In for MS Outlook [Seite 1796]
UpdateInstance (Method)

Definition
Updates the fields in the R3F file.

Remarks

Classification 1

UpdateInstance

- SAPData class [Seite 1918]
- SAPforms Data Plug-In for Workflow [Seite 1897]
- SAPData class [Seite 2078]
- SAPforms Data Plug-In for Demo [Seite 2077]

Properties
Updates the R3F file that was read by the last LoadInstance call. Unlike the SaveInstance method, this method is designed to update R3F attachments in incoming messages.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Path</td>
<td>Fully qualified target path of storage procedure</td>
</tr>
</tbody>
</table>

Classification 2

UpdateInstance

- SAPForm class [Seite 1812]
- SAPforms Form Plug-In for MS Outlook [Seite 1796]

Properties
Saves the modified form data to the R3F attachment in a mail.
**SAPXmit Class**

**Definition**

Controls communication between MS Outlook forms and an R/3 System.

**Remarks**

This class implements the SAPXmit interface through which other SAPForm and SAPData objects are enabled to transmit data to R/3. Communication between the form and the R/3 System be either synchronous (via RFC) or asynchronous (via mail). The class supplies methods for communication via MAPI.

If you want to develop your own SAPXmit-type plug-ins, you should use the source code supplied for the *SAPforms Transmit Plug-Ins for Demo* as a reference (template). The template is stored in the directory `...\sapforms\sapxdemo`.

**Classification**

*SAPforms Form Plug-In for MS Outlook [Seite 1796]*

→ SAPXmit Class

→ Execute (Method) [Seite 2210]
→ Load (Method) [Seite 2213]
→ MailItem (Property) [Seite 2176]
→ Options (Method) [Seite 2217]
→ Save (Method) [Seite 2219]
→ Start (Method) [Seite 2223]
→ Subject (Property) [Seite 2207]
Execute (Method)

Definition
Method that calls the Execute function in the R/3 System. The semantics of this function depend on the SAPData object.

Remarks
The method can be called directly via the SAPData object or indirectly via the Start method of the SAPForm object, whose SAPData property is a reference to the SAPData object.

Classification 1
Execute

Properties
Terminates a work item in the R/3 System and supplies its container elements with the data values in the fields list.

Classification 2
Execute

Properties
Does not execute an action in the R/3 System but rather displays the data of the object in a separate window.

Classification 3
Execute


Properties
Invokes the Execute method in the associated Xmit or SAPData object. The semantics of this method depend on this object.

Classification 4
Execute

Properties
This method executes a work item synchronously via a remote function call (RFC). Supports SAPData object of type Container.
This method causes a certain function module to be executed in the R/3 System. This module receives a table with a complete R3F structure containing the work item information and the container data that are to be supplied from the form.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Data</td>
<td>Object representing the container data.</td>
</tr>
</tbody>
</table>

Classification 5
Execute

Properties
Sends a mail to the R/3 System that contains the information required to terminate a work item (including container data) as an attachment.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Data</td>
<td>Object representing the container data.</td>
</tr>
</tbody>
</table>

Classification 6
Execute

Properties
Overwrites the R3F attachment, thereby updating it.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Pdata</td>
<td>Object that represents the mobjData container data</td>
</tr>
</tbody>
</table>

Classification 7
Execute

Properties
Properties
Displays the instance data of the associated SAPData object in a window instead of sending it to the R/3 System.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Data</td>
<td>object of class SAPData</td>
</tr>
</tbody>
</table>
Load (Method)

Definition
Loads structure information to SAPforms objects.

Remarks

Classification 1

Load

Properties
Loads the property values of an SAPData object from a structured string. As a result, the status of the SAPData object that was previously saved permanently with the Save method is restored. This saves the additional overheads that would result if the structure information had to be loaded each time from the original source. This method is usually called within the Load method of the SAPForm class.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>From</td>
<td>Character string containing the structure information</td>
</tr>
</tbody>
</table>

Classification 2

Load

Properties
Loads the property values of a field object from a structured string. As a result, the status of the field object that was previously saved permanently with the Save method is restored. For this method to be called successfully, the next character at the Pos position must be keyword “Fld”.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>From</td>
<td>Structured string</td>
</tr>
<tr>
<td>Pos</td>
<td>Current position within the string. Should point to the number of remaining properties to read</td>
</tr>
</tbody>
</table>

Classification 3

Load
**Load (Method)**

**Properties**
Loads the property values of a fields object from a structured string. As a result, the status of the fields object that was previously saved permanently with the Save method is restored.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>From</td>
<td>Character string that provides the structure information.</td>
</tr>
<tr>
<td>Pos</td>
<td>Current search position within the character string.</td>
</tr>
</tbody>
</table>

**Classification 4**

**Load**

**Properties**
Reconstructs the segment object from a structured string. Counterpart of the Save method.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>From</td>
<td>Character string containing structure information.</td>
</tr>
<tr>
<td>Pos</td>
<td>Current search position within the character string.</td>
</tr>
</tbody>
</table>

**Classification 5**

**Load**

**Properties**
This method reconstructs the segment list from a structured string. Counterpart of the Save method.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>From</td>
<td>Character string containing structure information.</td>
</tr>
<tr>
<td>Pos</td>
<td>Current search position within the character string.</td>
</tr>
</tbody>
</table>

**Classification 6**

**Load**

**Properties**
This method reconstructs an object of the Link class from the serialized representation. It assumes that the next token is located at the Pos position in the serialized representation of the “Fld” keyword.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>From</td>
<td>Serialization string.</td>
</tr>
</tbody>
</table>
Load (Method)

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Frm</td>
<td>form or form template containing controls.</td>
</tr>
<tr>
<td>Mode</td>
<td>0=design time (add-in), 1=design time (file), 2=runtime.</td>
</tr>
</tbody>
</table>

Classification 7

Load

- SAPForm class [Seite 1772]
- SAPForm class [Seite 1812]
- SAPForm class [Seite 1872]
- SAPforms Form Plug-In for MS Outlook [Seite 1796]
- SAPforms Form Plug-In for MS Office [Seite 1856]

Properties

Method for loading form controls into Controls Collection from a form or form template. Note that the form information is not stored in the Tag property of this object, but reconstructed each time the form is loaded. This method is called at design time and at runtime. It has to populate the Controls collection for both cases. At runtime, the method is called by the user code in the form's Load method. When the form is designed, the method is called by the SAPForms Designer or from outside SAPForms. If you are using the SAPForms Designer as a stand-alone application, the Load method is called with Mode=2. This instructs the Load method to load the form from a file. If you are using the SAPForms Designer as a plug-in, the Load method is invoked from the hosting environment and receives the proprietary form description as a parameter. The populated SAPForm object should be passed to the SAPForms Designer by the PlugIn method of the Designer class.

Classification 8

Load

- SAPXmit class [Seite 2106]
- SAPXmit class [Seite 1837]
- SAPXmit class [Seite 2126]
- SAPxmit class [Seite 2168]
- SAPXmit class [Seite 2147]
- SAPXmit class [Seite 2189]
- SAPXmit class [Seite 2209]
- SAPforms Transmit Plug-In for Demo [Seite 2208]
Properties

Loads the property values of an SAPXmit object from a structured string. As a result, the status of the SAPXmit object that was previously saved permanently with the Save method is restored.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>From</td>
<td>Character string containing the structure information</td>
</tr>
</tbody>
</table>
MailItem (Property)

Definition
Reference to the mail that is used to transport the form data.

Remarks
The property must be populated “externally” before the Start or Execute method is executed.

Classification
MailItem

← SAPXmit class [Seite 1837]
   ← SAPforms Form Plug-In for MS Outlook [Seite 1796]
← SAPXmit class [Seite 2168]
   ← SAPforms Transmit Plug-In for MAPI [Seite 2146]
← SAPXmit class [Seite 2147]
   ← SAPforms Transmit Plug-In for SAPMAPI [Seite 2167]
Options (Method)

Definition
Method that outputs the object-specific options dialog box.

Remarks
This dialog box displays either global settings or settings that are valid for the current form. These settings can be changed here. The global settings are read from the registry before they are displayed and are then saved again to the registry.
The method is called within the SAPforms Designer if the required plug-in is active under Properties on the Plug-Ins tab in the Options dialog. You can also use the sample program Options.exe to display the Options dialog of a certain plug-in.

Classification 1
Options

↓ SAPData class [Seite 1918]
↓ SAPforms Data Plug-In for Workflow [Seite 1897]
↓ SAPData class [Seite 1975]
↓ SAPforms Data Plug-In for Function Modules [Seite 1954]
↓ SAPData class [Seite 2027]
↓ SAPforms Data Plug-In for IDoc [Seite 2006]
↓ SAPData class [Seite 2078]
↓ SAPforms Data Plug-In for Demo [Seite 2077]

Classification 2
Options

↓ SAPForm class [Seite 1772]
↓ SAPforms Form Plug-In for Visual Basic [Seite 1756]
↓ SAPForm class [Seite 1812]
↓ SAPforms Form Plug-In for MS Outlook [Seite 1796]
↓ SAPForm class [Seite 1872]
↓ SAPforms Form Plug-In for MS Office [Seite 1856]

Classification 3
Options

↓ SAPXmit class [Seite 2106]
↓ SAPforms Data Plug-In for Demo [Seite 2077]
↓ SAPXmit class [Seite 1837]
↓ SAPforms Form Plug-In for MS Outlook [Seite 1796]
↓ SAPXmit class [Seite 2126]
Options (Method)

- SAPforms Transmit Plug-In for RFC [Seite 2125]
- SAPXmit class [Seite 2168]
- SAPforms Transmit Plug-In for MAPI [Seite 2146]
- SAPXmit class [Seite 2147]
- SAPforms Transmit Plug-In for SAPMAPI [Seite 2167]
- SAPXmit class [Seite 2189]
- SAPforms Transmit Plug-In for SMTP [Seite 2188]
- SAPXmit class [Seite 2209]
- SAPforms Transmit Plug-In for Demo [Seite 2208]
Save (Method)

Definition
Saves the updated form definition.

Remarks
This method is used by the SAPforms Designer when a form is designed to save the modified form definition after the binding has been defined. The modifications can affect the binding definition or possibly the source code entered in the form.

Classification 1
Save

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Tgt</td>
<td>String in which the status is stored.</td>
</tr>
</tbody>
</table>

Properties
Saves the current property values of an SAPData object in a structured string. Counterpart of the Load method.

When a form is created, the SAPforms Designer uses this method to save the structure description retrieved from the R/3 System permanently in the form.

The R/3 structure is restored from the structured string at runtime by the Load method.

Classification 2
Save

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Tgt</td>
<td>Stores the string structure.</td>
</tr>
</tbody>
</table>

Properties
Saves the current property values of an SAPData object in a structured string. Counterpart of the Load method. The string is saved with a form so that the structure information is available when the form is executed.
Classification 3

Save

- Fields Class [Seite 2018]
- SAPforms Data Plug-In for Workflow [Seite 1897]
- SAPforms Data Plug-In for Function Modules [Seite 1954]
- SAPforms Data Plug-In for IDoc [Seite 2006]

Properties

Loads the property values of a fields object from a structured string. As a result, the status of the fields object that was previously saved permanently with the Save method is restored.

The string is saved with the form so that the structure information is available when the form is executed.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Tgt</td>
<td>Stores the string structure.</td>
</tr>
</tbody>
</table>

Classification 4

Save

- Segment class [Seite 2055]
- SAPforms Data Plug-In for IDoc [Seite 2006]

Properties

This method saves the structure of a segment, including the field list, in a structured string.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Tgt</td>
<td>Stores the string structure.</td>
</tr>
</tbody>
</table>

Classification 5

Save

- Segments class [Seite 2067]
- SAPforms Data Plug-In for IDoc [Seite 2006]

Properties

This method saves the segment list in a structured string. Counterpart of the Load method.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Tgt</td>
<td>Stores the string structure.</td>
</tr>
</tbody>
</table>

Classification 6

Save

- Links class [Seite 1862]
- SAPforms Form Plug-In for Visual Basic [Seite 1756]
- Links class [Seite 1862]
- SAPforms Form Plug-In for MS Office [Seite 1856]
- Links class [Seite 1862]
- SAPforms Form Plug-In for MS Outlook [Seite 1796]
Properties
Method that writes the content of all links to a file. Counterpart of the Load method.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Tgt</td>
<td>Stores the string structure.</td>
</tr>
</tbody>
</table>

Classification 7
Save

- SAPForm class [Seite 1772]
- SAPforms Form Plug-In for Visual Basic [Seite 1756]
- SAPForm class [Seite 1872]
- SAPforms Form Plug-In for MS Office [Seite 1856]
- SAPForm class [Seite 1812]
- SAPforms Form Plug-In for MS Outlook [Seite 1796]

Properties
Method that saves a form to a file if it was created from a file.
Reads the old file and writes updated lines to the new file.

Classification 8
Save

- SAPXmit class [Seite 2106]
- SAPforms Data Plug-In for Demo [Seite 2077]
- SAPXmit class [Seite 1837]
- SAPforms Form Plug-In for Outlook [Seite 1796]
- SAPXmit class [Seite 2126]
- SAPforms Transmit Plug-In for RFC [Seite 2125]
- SAPXmit class [Seite 2168]
- SAPforms Transmit Plug-In for MAPI [Seite 2146]
- SAPXmit class [Seite 2147]
- SAPforms Transmit Plug-In for SAPMAPI [Seite 2167]
- SAPXmit class [Seite 2189]
- SAPforms Transmit Plug-In for WinSock [Seite 2188]
- SAPXmit class [Seite 2209]
- SAPforms Transmit Plug-In for Demo [Seite 2208]

Properties
Saves the current property values of an SAPXmit object in a structured string. Counterpart of the Load method.
When a form is created, the SAPforms Designer uses this method to save the structure description retrieved from the R/3 System permanently in the form.
The R/3 structure is restored from the structured string at runtime by the Load method.
<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Tgt</td>
<td>String in which the status is stored.</td>
</tr>
</tbody>
</table>
Start (Method)

Definition
Method that executes the Start function in R/3. The semantics of this function depend on the concrete SAPData object.

Remarks
In the case of the SAPData object for workflow containers, this method starts a workflow and populates its container elements with values from the form fields. This method can be called directly via the SAPData object or indirectly via the Start method of the SAPForm object, whose SAPData property is a reference to the SAPData object. If an SAPXmit object is assigned to the SAPData object, the Start method of the SAPXmit object is called.

Classification 1
Start

Properties
Starts a workflow in the R/3 System and supplies its container elements with the data values in the fields list.

Classification 2
Start

Properties
This method carries out the current R/3 function module call.

Classification 3
Start

Properties
Generates an IDoc file from the data in the segments list and transfers this IDoc to the R/3 System. The transfer method depends on the assigned SAPXmit object. A workflow can be started in the R/3 System when the inbound IDoc is processed.

Classification 4
Start

Properties

Properties
Does not execute an action in the R/3 System but rather displays the data of the object in a separate window.

Classification 5
Start

Properties
Invokes the Start method in the associated SAPXmit or SAPData object. The semantics of this method depend on this SAPData object. If this object is the SAPData object for SAP Business Workflow, this method starts a workflow in an R/3 System.

Classification 6
Start

Properties
This method starts a workflow synchronously via RFC. Supports SAPData objects of types "Idoc" and "Container".

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Data</td>
<td>Object of type SAPData.</td>
</tr>
</tbody>
</table>

Classification 7
Start

Properties
Sends a mail to the R/3 System with an attachment containing the information required to start a workflow.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Data</td>
<td>Object of type SAPData.</td>
</tr>
</tbody>
</table>
Classification 8

Start

- SAPXmit class [Seite 2209]
- SAPforms Transmit Plug-In for Demo [Seite 2208]
- SAPXmit class [Seite 2106]
- SAPforms Data Plug-In for Demo [Seite 2077]

Properties

Displays the instance data of the associated SAPData object in a window instead of sending it to the R/3 System.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Data</td>
<td>Object of type SAPData.</td>
</tr>
</tbody>
</table>
Subject (Property)

Definition
“Subject” property of the message sent to the R/3 System by mail.

Remarks
If you do not specify a subject explicitly, the system chooses a subject that can be read by this property after the mail has been sent.

Classification

Subject

← SAPXmit class [Seite 2168]
← SAPforms Transmit Plug-In for MAPI [Seite 2146]
← SAPXmit class [Seite 2147]
← SAPforms Transmit Plug-In for SAPMAPI [Seite 2167]
← SAPXmit class [Seite 2189]
← SAPforms Transmit Plug-In for SMTP [Seite 2188]
← SAPXmit class [Seite 1837]
← SAPforms Form Plug-In for MS Outlook [Seite 1796]
SAPforms Form Plug-In for MS Office

Definition
Form plug-in that implements an interface between MS Word or MS Excel forms and SAPforms.

Use
You use this form whenever you want to process an electronic form designed with MS Word or MS Excel. You must set this plug-in if you want to use the stand-alone version of the SAPforms Designer [Seite 1735]. If you use the Submit control [Seite 1747] in MS Word or MS Excel, the plug-in is set automatically.
This plug-in implements the SAPforms interface that provides the SAPforms Designer and other SAPforms components with generic access to the properties and controls of a certain form type (Visual Basic, MS Outlook, MS Office). In other words, this plug-in can be regarded as being synonymous with the associated SAPforms class.

Classification
SAPforms Form Plug-In for MS Office
→ Link class [Seite 1857]
→ Links class [Seite 1862]
→ SAPCtrl class [Seite 1867]
→ SAPform class [Seite 1872]
Link Class

Definition
Represents a binding between a generic form control and an R/3 data field.

Remarks
A form control may be linked to several data fields in an SAPData object. All such bindings of a form control are contained in the Links list in the SAPCtrl class. The bindings are established when the form is designed by means of the SAPForms Designer and are stored in the Save method of the SAPForm class. Furthermore, they are reconstructed in the Load method of the SAPForm class every time the form is loaded and are used by the PutData and GetData methods to populate the SAP fields with the form fields and vice versa.

The form control referenced in the link is identified by means of its display name. The data field referenced in the link is identified by means of the field qualifier and optionally by a field selector plus selector value.

Links are m:n relationships, that is, a form field may be linked to several data fields and a certain data field may be linked to several form fields.

The Link class is only required for developing plug-ins.

Classification
SAPforms Form Plug-In for MS Office [Seite 1856]
SAPforms Form Plug-In for Visual Basic [Seite 1756]
SAPforms Form Plug-In for MS Outlook [Seite 1796]

→ Link Class
→ Load (Method) [Seite 2213]
Load (Method)

Definition
Loads structure information to SAPforms objects.

Remarks

Classification 1

Load

Properties
Loads the property values of an SAPData object from a structured string. As a result, the status of the SAPData object that was previously saved permanently with the Save method is restored. This saves the additional overheads that would result if the structure information had to be loaded each time from the original source. This method is usually called within the Load method of the SAPForm class.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>From</td>
<td>Character string containing the structure information</td>
</tr>
</tbody>
</table>

Classification 2

Load

Properties
Loads the property values of a field object from a structured string. As a result, the status of the field object that was previously saved permanently with the Save method is restored. For this method to be called successfully, the next character at the Pos position must be keyword “Fld”.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>From</td>
<td>Structured string</td>
</tr>
<tr>
<td>Pos</td>
<td>Current position within the string. Should point to the number of remaining properties to read</td>
</tr>
</tbody>
</table>

Classification 3

Load
Load (Method)

Properties
Loads the property values of a fields object from a structured string. As a result, the status of the fields object that was previously saved permanently with the Save method is restored.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>From</td>
<td>Character string that provides the structure information.</td>
</tr>
<tr>
<td>Pos</td>
<td>Current search position within the character string.</td>
</tr>
</tbody>
</table>

Classification 4
Load

Properties
Reconstructs the segment object from a structured string. Counterpart of the Save method.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>From</td>
<td>Character string containing structure information.</td>
</tr>
<tr>
<td>Pos</td>
<td>Current search position within the character string.</td>
</tr>
</tbody>
</table>

Classification 5
Load

Properties
This method reconstructs the segment list from a structured string. Counterpart of the Save method.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>From</td>
<td>Character string containing structure information.</td>
</tr>
<tr>
<td>Pos</td>
<td>Current search position within the character string.</td>
</tr>
</tbody>
</table>

Classification 6
Load

Properties
This method reconstructs an object of the Link class from the serialized representation. It assumes that the next token is located at the Pos position in the serialized representation of the “Fld” keyword.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>From</td>
<td>Serialization string.</td>
</tr>
</tbody>
</table>
Load (Method)

| Pos | Current position within the character string. |

### Classification 7

#### Load

- SAPForm class [Seite 1772]
- SAPforms Form Plug-In for Visual Basic [Seite 1756]
- SAPForm class [Seite 1812]
- SAPforms Form Plug-In for MS Outlook [Seite 1796]
- SAPForm class [Seite 1872]
- SAPforms Form Plug-In for MS Office [Seite 1856]

### Properties

Method for loading form controls into Controls Collection from a form or form template.

Note that the form information is not stored in the Tag property of this object, but reconstructed each time the form is loaded. This method is called at design time and at runtime. It has to populate the Controls collection for both cases. At runtime, the method is called by the user code in the form's Load method. When the form is designed, the method is called by the SAPForms Designer or from outside SAPForms. If you are using the SAPForms Designer as a stand-alone application, the Load method is called with Mode=2. This instructs the Load method to load the form from a file. If you are using the SAPForms Designer as a plug-in, the Load method is invoked from the hosting environment and receives the proprietary form description as a parameter. The populated SAPForm object should be passed to the SAPForms Designer by the PlugIn method of the Designer class.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Frm</td>
<td>form or form template containing controls.</td>
</tr>
<tr>
<td>Mode</td>
<td>0=design time (add-in), 1=design time (file), 2=runtime.</td>
</tr>
</tbody>
</table>

### Classification 8

#### Load

- SAPXmit class [Seite 2106]
- SAPforms Data Plug-In for Demo [Seite 2077]
- SAPXmit class [Seite 1837]
- SAPforms Form Plug-In for Outlook [Seite 1796]
- SAPXmit class [Seite 2126]
- SAPforms Transmit Plug-In for RFC [Seite 2125]
- SAPXmit class [Seite 2168]
- SAPforms Transmit Plug-In for MAPI [Seite 2146]
- SAPXmit class [Seite 2147]
- SAPforms Transmit Plug-In for SAPMAPI [Seite 2167]
- SAPXmit class [Seite 2189]
- SAPforms Transmit Plug-In for WinSock [Seite 2188]
- SAPXmit class [Seite 2209]
- SAPforms Transmit Plug-In for Demo [Seite 2208]
Properties

Loads the property values of an SAPXmit object from a structured string. As a result, the status of the SAPXmit object that was previously saved permanently with the Save method is restored.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>From</td>
<td>Character string containing the structure information</td>
</tr>
</tbody>
</table>
Links Class

Definition
Collection of all the bindings in a form containing an object of the Link class.

Remarks
The content of this list is stored in serialized form in the Tag property of the form. The list is reconstructed from this property in the Load method of the SAPforms object.

Classification
SAPforms Form Plug-In for MS Office [Seite 1856]
SAPforms Form Plug-In for Visual Basic [Seite 1756]
SAPforms Form Plug-In for MS Outlook [Seite 1796]

→Links Class
→Save (Method) [Seite 2219]
Save (Method)

Definition
Saves the updated form definition.

Remarks
This method is used by the SAPforms Designer when a form is designed to save the modified form definition after the binding has been defined. The modifications can affect the binding definition or possibly the source code entered in the form.

Classification 1
Save

Properties
Saves the current property values of an SAPData object in a structured string. Counterpart of the Load method.
When a form is created, the SAPforms Designer uses this method to save the structure description retrieved from the R/3 System permanently in the form.
The R/3 structure is restored from the structured string at runtime by the Load method.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Tgt</td>
<td>String in which the status is stored.</td>
</tr>
</tbody>
</table>

Classification 2
Save

Properties
Saves the current property values of an SAPData object in a structured string. Counterpart of the Load method. The string is saved with a form so that the structure information is available when the form is executed.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Tgt</td>
<td>Stores the string structure.</td>
</tr>
</tbody>
</table>
## Classification 3

**Save**

- Fields Class [Seite 2018]
- SAPforms Data Plug-In for Workflow [Seite 1897]
- SAPforms Data Plug-In for Function Modules [Seite 1954]
- SAPforms Data Plug-In for IDoc [Seite 2006]

**Properties**

Loads the property values of a fields object from a structured string. As a result, the status of the fields object that was previously saved permanently with the Save method is restored.

The string is saved with the form so that the structure information is available when the form is executed.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Tgt</td>
<td>Stores the string structure.</td>
</tr>
</tbody>
</table>

## Classification 4

**Save**

- Segment class [Seite 2055]
- SAPforms Data Plug-In for IDoc [Seite 2006]

**Properties**

This method saves the structure of a segment, including the field list, in a structured string.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Tgt</td>
<td>Stores the string structure.</td>
</tr>
</tbody>
</table>

## Classification 5

**Save**

- Segments class [Seite 2067]
- SAPforms Data Plug-In for IDoc [Seite 2006]

**Properties**

This method saves the segment list in a structured string. Counterpart of the Load method.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Tgt</td>
<td>Stores the string structure.</td>
</tr>
</tbody>
</table>

## Classification 6

**Save**

- Links class [Seite 1862]
- SAPforms Form Plug-In for Visual Basic [Seite 1756]
- Links class [Seite 1862]
- SAPforms Form Plug-In for MS Office [Seite 1856]
- Links class [Seite 1862]
- SAPforms Form Plug-In for MS Outlook [Seite 1796]
Properties
Method that writes the content of all links to a file. Counterpart of the Load method.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Tgt</td>
<td>Stores the string structure.</td>
</tr>
</tbody>
</table>

Classification 7
Save

↓ SAPForm class [Seite 1772]
  ↓ SAPforms Form Plug-In for Visual Basic [Seite 1756]
  ↓ SAPForm class [Seite 1872]
  ↓ SAPforms Form Plug-In for MS Office [Seite 1856]
  ↓ SAPForm class [Seite 1812]
  ↓ SAPforms Form Plug-In for MS Outlook [Seite 1796]

Properties
Method that saves a form to a file if it was created from a file. Reads the old file and writes updated lines to the new file.

Classification 8
Save

↓ SAPXmit class [Seite 2106]
  ↓ SAPforms Data Plug-In for Demo [Seite 2077]
  ↓ SAPXmit class [Seite 1837]
  ↓ SAPforms Form Plug-In for Outlook [Seite 1796]
  ↓ SAPXmit class [Seite 2126]
  ↓ SAPforms Transmit Plug-In for RFC [Seite 2125]
  ↓ SAPXmit class [Seite 2168]
  ↓ SAPforms Transmit Plug-In for MAPI [Seite 2146]
  ↓ SAPXmit class [Seite 2147]
  ↓ SAPforms Transmit Plug-In for SAPMAPI [Seite 2167]
  ↓ SAPXmit class [Seite 2189]
  ↓ SAPforms Transmit Plug-In for WinSock [Seite 2188]
  ↓ SAPXmit class [Seite 2209]
  ↓ SAPforms Transmit Plug-In for Demo [Seite 2208]

Properties
Saves the current property values of an SAPXmit object in a structured string. Counterpart of the Load method.
When a form is created, the SAPforms Designer uses this method to save the structure description retrieved from the R/3 System permanently in the form.
The R/3 structure is restored from the structured string at runtime by the Load method.
### Save (Method)

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Tgt</td>
<td>String in which the status is stored.</td>
</tr>
</tbody>
</table>
SAPCtrl Class

Definition
This class implements a generic form control that is contained in the Controls list of the SAPForm class.

Remarks
Represents a dependent object of the SAPform interface for Visual Basic forms. The Controls list of the SAPForm class consists of objects of this type. An object in the SAPCtrl class usually represents a form control. The table controls are an exception here. With table controls, an object of the SAPCtrl class is added to the Controls list for each column. In the case of control fields, each element of a field is added to the Controls list as if it were a separate control.
This interface provides the SAPForms Designer and other SAPForms components with generic access to form properties and controls. The concept requires that each object that is a possible binding target be interpreted as a separate generic control. With VB, this means that a separate control is to be created for every element of a control array and for every grid column.

Classification
SAPforms Form Plug-In for MS Office [Seite 1856]

→ SAPCtrl Class
  → Name (Property) [Seite 2088]
  → Value (Property) [Seite 2103]
**Name (Property)**

**Definition**
Name of the object, field, segment, or control.

**Remarks**
This name, for example, must be used as the root in the SAP structure tree that is built by the `BuildTree` method.

**Classification 1**

**Name**

- SAPData class [Seite 1918]
- SAPforms Data Plug-In for Workflow [Seite 1897]

**Properties**
Returns the name of the workflow to whose task the container description belongs.

**Classification 2**

**Name**

- SAPData class [Seite 1975]
- SAPforms Data Plug-In for Function Modules [Seite 1954]

**Properties**
Name of the R/3 function module that is to be executed.

**Classification 3**

**Name**

- SAPData class [Seite 2027]
- SAPforms Data Plug-In for IDoc [Seite 2006]

**Properties**
Name of the IDoc as it appears in the control record.

**Classification 4**

**Name**

- SAPData class [Seite 2078]
- SAPforms Data Plug-In for Demo [Seite 2077]

**Classification 5**

**Name**

- Field Class [Seite 2007]
- SAPforms Data Plug-In for Workflow [Seite 1897]
- SAPforms Data Plug-In for Function Modules [Seite 1954]
- SAPforms Data Plug-In for IDoc [Seite 2006]
Properties
Name of the field.

Classification 6

Name

- Segment class [Seite 2055]
- SAPforms Data Plug-In for IDoc [Seite 2006]

Properties
Name of the segment. Used as a key in the Templates list of the IDoc object. Only the first attribute with this name is added to the list of segments with a key. All other segment attributes with the same name are added without a key.

Classification 7

Name

- SAPForm class [Seite 1772]
- SAPforms Form Plug-In for Visual Basic [Seite 1756]
- SAPForm Class [Seite 1812]
- SAPforms Form Plug-In for MS Outlook [Seite 1796]
- SAPForm class [Seite 1872]
- SAPforms Form Plug-In for MS Office [Seite 1856]

Properties
Original name of the control in the real form (e.g. Visual Basic form).
Value (Property)

Definition
Value of a specific field type.

Remarks

Classification 1

Value

Properties
The value of an SAP data field, defined in greater detail by the Qual parameter.
The structure and syntax of the Qual parameter depends on the concrete SAP data structure from which this field originates. In the simplest case, the Qual parameter designates the name of a field whose value is to be set or read. If the field is part of a structure (for example, an IDoc segment), Qual must identify the field uniquely within the overall structure. If the structure is repeated within an instance - as is the case with some IDocs - Qual must also contain a selector that identifies the IDoc segment uniquely.

```
obj.Value("Field1") = "abc".
obj.Value("Seg1.Field1") = "abc".
X = obj.Value("Seg1.Seg2.Field1").
X = obj.Value("Seg1.Field1 Seg2.Field2 = '123'" )  ' selector.
```

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Qual</td>
<td>Qualifier and selector that uniquely identifies a particular field.</td>
</tr>
</tbody>
</table>

Classification 2

Value

Properties

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Qual</td>
<td>Qualifier and selector that uniquely identifies a particular field.</td>
</tr>
</tbody>
</table>
Properties

Returns the value of control.
This property is an abstraction of the different properties used by the original controls to present their content, for example, as Text, Caption, Value, and so on. The implementation of the SAPCtrl must map the original control property in the Value property for each control that is to be supported by SAPforms.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>X</td>
<td>Index (e.g. for grids).</td>
</tr>
<tr>
<td>Return</td>
<td>Value</td>
</tr>
</tbody>
</table>
**SAPForm Class**

**Definition**
Original class of this plug-in.

**Remarks**
This class implements the SAPForm interface for MS Office forms and makes these forms available to the SAPForms Designer and other components of SAPForms. The Label, TextBox, ComboBox, OptionButton, CheckBox, ListBox, SAPTableView, Grid, as well as MSFlexGrid controls and even control arrays are supported.

The classes of the SAPForm plug-in type implement the SAPforms interface that provides the SAPforms Designer and other SAPforms components with generic access to the properties and controls of a certain form type (Visual Basic, MS Outlook, MS Office). In other words, these classes can be regarded as being synonymous with various plug-ins of this type.

The SAPForm object is used as an anchor to all other SAPforms objects. The SAPData property, for example, contains a reference to the SAPData object used in the form, while the SAPXmit property references the SAPXmit object (see below) used.

If you want to develop your own SAPForm-type plug-ins, you should use the source code supplied for plug-ins for Visual Basic as a reference (template). The template is stored in the directory ...\sapforms\sapfvb4.

**Classification**
SAPforms Form Plug-In for MS Office [Seite 1856]

- SAPForm Class
  - Execute (Method) [Seite 2210]
  - GetData (Method) [Seite 1876]
  - Load (Method) [Seite 2213]
  - LogIt (Method) [Seite 1931]
  - Options (Method) [Seite 2217]
  - PutData (Method) [Seite 1884]
  - SAPData (Property) [Seite 1885]
  - SAPXmit (Property) [Seite 2093]
  - Save (Method) [Seite 2219]
  - Start (Method) [Seite 2223]
  - Submit (Method) [Seite 1894]
Execute (Method)

Definition
Method that calls the Execute function in the R/3 System. The semantics of this function depend on the SAPData object.

Remarks
The method can be called directly via the SAPData object or indirectly via the Start method of the SAPForm object, whose SAPData property is a reference to the SAPData object.

Classification 1
Execute

Properties
Terminates a work item in the R/3 System and supplies its container elements with the data values in the fields list.

Classification 2
Execute

Properties
Does not execute an action in the R/3 System but rather displays the data of the object in a separate window.

Classification 3
Execute
Properties
Invokes the Execute method in the associated Xmit or SAPData object. The semantics of this method depend on this object.

Classification 4
Execute
← SAPXmit class [Seite 2126]
← SAPforms Transmit Plug-In for RFC [Seite 2125]

Properties
This method executes a work item synchronously via a remote function call (RFC). Supports SAPData object of type Container.
This method causes a certain function module to be executed in the R/3 System. This module receives a table with a complete R3F structure containing the work item information and the container data that are to be supplied from the form.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Data</td>
<td>Object representing the container data.</td>
</tr>
</tbody>
</table>

Classification 5
Execute
← SAPXmit class [Seite 2168]
← SAPforms Transmit Plug-In for MAPI [Seite 2146]
← SAPXmit class [Seite 2189]
← SAPforms Transmit Plug-In for WinSock [Seite 2188]

Properties
Sends a mail to the R/3 System that contains the information required to terminate a work item (including container data) as an attachment.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Data</td>
<td>Object representing the container data.</td>
</tr>
</tbody>
</table>

Classification 6
Execute
← SAPXmit class [Seite 2147]
← SAPforms Transmit Plug-In for SAPMAPI [Seite 2167]

Properties
Overwrites the R3F attachment, thereby updating it.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Pdata</td>
<td>Object that represents the mobiData container data</td>
</tr>
</tbody>
</table>

Classification 7
Execute
← SAPXmit class [Seite 2209]
← SAPforms Transmit Plug-In for Demo [Seite 2208]
Properties
Displays the instance data of the associated SAPData object in a window instead of sending it to the R/3 System.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Data</td>
<td>object of class SAPData</td>
</tr>
</tbody>
</table>
**GetData (Method)**

**Definition**
Transfers the contents of the SAPData object to the form.

**Remarks**
This method evaluates the link chain associated with each control in order to make the transfer. The method is called from the code contained in the Load method of a form.

**Classification**

- SAPForm class [Seite 1772]
- SAPforms Form Plug-In for Visual Basic [Seite 1756]
- SAPForm class [Seite 1812]
- SAPforms Form Plug-In for MS Outlook [Seite 1796]
- SAPForm class [Seite 1872]
- SAPforms Form Plug-In for MS Office [Seite 1856]
Load (Method)

Definition
Loads structure information to SAPforms objects.

Remarks

Classification 1
Load

Properties
Loads the property values of an SAPData object from a structured string. As a result, the status of the SAPData object that was previously saved permanently with the Save method is restored. This saves the additional overheads that would result if the structure information had to be loaded each time from the original source. This method is usually called within the Load method of the SAPForm class.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>From</td>
<td>Character string containing the structure information</td>
</tr>
</tbody>
</table>

Classification 2
Load

Properties
Loads the property values of a field object from a structured string. As a result, the status of the field object that was previously saved permanently with the Save method is restored. For this method to be called successfully, the next character at the Pos position must be keyword “Fld”.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>From</td>
<td>Structured string</td>
</tr>
<tr>
<td>Pos</td>
<td>Current position within the string. Should point to the number of remaining properties to read</td>
</tr>
</tbody>
</table>

Classification 3
Load
Load (Method)

Properties
Loads the property values of a fields object from a structured string. As a result, the status of the fields object that was previously saved permanently with the Save method is restored.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>From</td>
<td>Character string that provides the structure information.</td>
</tr>
<tr>
<td>Pos</td>
<td>Current search position within the character string.</td>
</tr>
</tbody>
</table>

Classification 4
Load

Properties
Reconstructs the segment object from a structured string. Counterpart of the Save method.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>From</td>
<td>Character string containing structure information.</td>
</tr>
<tr>
<td>Pos</td>
<td>Current search position within the character string.</td>
</tr>
</tbody>
</table>

Classification 5
Load

Properties
This method reconstructs the segment list from a structured string. Counterpart of the Save method.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>From</td>
<td>Character string containing structure information.</td>
</tr>
<tr>
<td>Pos</td>
<td>Current search position within the character string.</td>
</tr>
</tbody>
</table>

Classification 6
Load

Properties
This method reconstructs an object of the Link class from the serialized representation. It assumes that the next token is located at the Pos position in the serialized representation of the “Fld” keyword.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>From</td>
<td>Serialization string.</td>
</tr>
</tbody>
</table>
Classification 7

Load

- SAPForm class [Seite 1772]
- SAPforms Form Plug-In for Visual Basic [Seite 1756]
- SAPForm class [Seite 1812]
- SAPforms Form Plug-In for MS Outlook [Seite 1796]
- SAPForm class [Seite 1872]
- SAPforms Form Plug-In for MS Office [Seite 1856]

Properties

Method for loading form controls into Controls Collection from a form or form template. Note that the form information is not stored in the Tag property of this object, but reconstructed each time the form is loaded. This method is called at design time and at runtime. It has to populate the Controls collection for both cases. At runtime, the method is called by the user code in the form's Load method. When the form is designed, the method is called by the SAPForms Designer or from outside SAPForms. If you are using the SAPForms Designer as a stand-alone application, the Load method is called with Mode=2. This instructs the Load method to load the form from a file. If you are using the SAPForms Designer as a plug-in, the Load method is invoked from the hosting environment and receives the proprietary form description as a parameter. The populated SAPForm object should be passed to the SAPForms Designer by the PlugIn method of the Designer class.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Frm</td>
<td>form or form template containing controls.</td>
</tr>
<tr>
<td>Mode</td>
<td>0=design time (add-in), 1=design time (file), 2=runtime.</td>
</tr>
</tbody>
</table>

Classification 8

Load

- SAPXmit class [Seite 2106]
- SAPforms Data Plug-In for Demo [Seite 2077]
- SAPXmit class [Seite 1837]
- SAPforms Form Plug-In for Outlook [Seite 1796]
- SAPXmit class [Seite 2126]
- SAPforms Transmit Plug-In for RFC [Seite 2125]
- SAPXmit class [Seite 2168]
- SAPforms Transmit Plug-In for MAPI [Seite 2146]
- SAPXmit class [Seite 2147]
- SAPforms Transmit Plug-In for SAPMAPI [Seite 2167]
- SAPXmit class [Seite 2189]
- SAPforms Transmit Plug-In for WinSock [Seite 2188]
- SAPXmit class [Seite 2209]
- SAPforms Transmit Plug-In for Demo [Seite 2208]
Load (Method)

**Properties**

Loads the property values of an SAPXmit object from a structured string. As a result, the status of the SAPXmit object that was previously saved permanently with the Save method is restored.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>From</td>
<td>Character string containing the structure information</td>
</tr>
</tbody>
</table>
LogIt (Method)

Definition
Allows form-specific data to be written to the Trace Log Viewer [Seite 1742].

Remarks

Classification

LogIt

← SAPData [Seite 1918]
← SAPforms Data Plug-In for Workflow [Seite 1897]
← SAPForm [Seite 1772]
← SAPforms Form Plug-In for Visual Basic [Seite 1756]
← SAPForm [Seite 1812]
← SAPforms Form Plug-In for MS Outlook [Seite 1796]
← SAPForm [Seite 1872]
← SAPforms Form Plug-In for MS Office [Seite 1856]
Options (Method)

Definition
Method that outputs the object-specific options dialog box.

Remarks
This dialog box displays either global settings or settings that are valid for the current form. These settings can be changed here. The global settings are read from the registry before they are displayed and are then saved again to the registry.

The method is called within the SAPforms Designer if the required plug-in is active under Properties on the Plug-Ins tab in the Options dialog. You can also use the sample program Options.exe to display the Options dialog of a certain plug-in.

Classification 1
Options

- SAPData class [Seite 1918]
- SAPforms Data Plug-In for Workflow [Seite 1897]
- SAPData class [Seite 1975]
- SAPforms Data Plug-In for Function Modules [Seite 1954]
- SAPData class [Seite 2027]
- SAPforms Data Plug-In for IDoc [Seite 2006]
- SAPData class [Seite 2078]
- SAPforms Data Plug-In for Demo [Seite 2077]

Classification 2
Options

- SAPForm class [Seite 1772]
- SAPforms Form Plug-In for Visual Basic [Seite 1756]
- SAPForm class [Seite 1812]
- SAPforms Form Plug-In for MS Outlook [Seite 1796]
- SAPForm class [Seite 1872]
- SAPforms Form Plug-In for MS Office [Seite 1856]

Classification 3
Options

- SAPXmit class [Seite 2106]
- SAPforms Data Plug-In for Demo [Seite 2077]
- SAPXmit class [Seite 1837]
- SAPforms Form Plug-In for MS Outlook [Seite 1796]
- SAPXmit class [Seite 2126]
- SAPforms Transmit Plug-In for RFC [Seite 2125]
- SAPXmit class [Seite 2168]
  - SAPforms Transmit Plug-In for MAPI [Seite 2146]
- SAPXmit class [Seite 2147]
  - SAPforms Transmit Plug-In for SAPMAPI [Seite 2167]
- SAPXmit class [Seite 2189]
  - SAPforms Transmit Plug-In for SMTP [Seite 2188]
- SAPXmit class [Seite 2209]
  - SAPforms Transmit Plug-In for Demo [Seite 2208]
PutData (Method)

**Definition**
Transfers the contents of the form to the SAPData object.

**Remarks**
This method evaluates the link chain associated with each control in order to make the transfer. After the method has been executed, the SAPData object is populated with instance data and can be transferred to the R/3 System.
This method is usually called by the Start or Execute method within an event procedure of a form.

**Classification**

PutData

[←SAPForm class [Seite 1772]]

[←SAPforms Form Plug-In for Visual Basic [Seite 1756]]

[←SAPForm class [Seite 1812]]

[←SAPforms Form Plug-In for MS Outlook [Seite 1796]]

[←SAPForm class [Seite 1872]]

[←SAPforms Form Plug-In for MS Office [Seite 1856]]
SAPData (Property)

Definition
Contains a reference to the SAPData object used in the form.

Remarks
The SAPData object is reconstructed at runtime by the form's Load method and then made available via this property.

Classification
SAPData

← SAPForm class [Seite 1772]
← SAPforms Form Plug-In for Visual Basic [Seite 1756]
← SAPForm class [Seite 1812]
← SAPforms Form Plug-In for MS Outlook [Seite 1796]
← SAPForm class [Seite 1872]
← SAPforms Form Plug-In for MS Office [Seite 1856]
SAPXmit (Property)

Definition
Reference to the SAPXmit object that is used to encapsulate the transport of SAPData to the R/3 System.

Remarks
Classification
SAPXmit

- SAPData class [Seite 1918]
- SAPforms Data Plug-In for Workflow [Seite 1897]
- SAPData class [Seite 1975]
- SAPforms Data Plug-In for Function Modules [Seite 1954]
- SAPData class [Seite 2027]
- SAPforms Data Plug-In for IDoc [Seite 2006]
- SAPData class [Seite 2078]
- SAPforms Data Plug-In for Demo [Seite 2077]
- SAPForm class [Seite 1772]
- SAPforms Form Plug-In for Visual Basic [Seite 1756]
- SAPForm class [Seite 1812]
- SAPforms Form Plug-In for MS Outlook [Seite 1796]
- SAPForm class [Seite 1872]
- SAPforms Form Plug-In for MS Office [Seite 1856]
Save (Method)

Definition
Saves the updated form definition.

Remarks
This method is used by the SAPforms Designer when a form is designed to save the modified form definition after the binding has been defined. The modifications can affect the binding definition or possibly the source code entered in the form.

Classification 1
Save

Properties
Saves the current property values of an SAPData object in a structured string. Counterpart of the Load method.

Parameter Description
Tgt String in which the status is stored.

Classification 2
Save

Properties
Saves the current property values of an SAPData object in a structured string. Counterpart of the Load method. The string is saved with a form so that the structure information is available when the form is executed.

Parameter Description
Tgt Stores the string structure.
**Classification 3**

**Save**

- Fields Class [Seite 2018]
- SAPforms Data Plug-In for Workflow [Seite 1897]
- SAPforms Data Plug-In for Function Modules [Seite 1954]
- SAPforms Data Plug-In for IDoc [Seite 2006]

**Properties**

Loads the property values of a fields object from a structured string. As a result, the status of the fields object that was previously saved permanently with the Save method is restored.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Tgt</td>
<td>Stores the string structure.</td>
</tr>
</tbody>
</table>

**Classification 4**

**Save**

- Segment class [Seite 2055]
- SAPforms Data Plug-In for IDoc [Seite 2006]

**Properties**

This method saves the structure of a segment, including the field list, in a structured string.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Tgt</td>
<td>Stores the string structure.</td>
</tr>
</tbody>
</table>

**Classification 5**

**Save**

- Segments class [Seite 2067]
- SAPforms Data Plug-In for IDoc [Seite 2006]

**Properties**

This method saves the segment list in a structured string. Counterpart of the Load method.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Tgt</td>
<td>Stores the string structure.</td>
</tr>
</tbody>
</table>

**Classification 6**

**Save**

- Links class [Seite 1862]
- SAPforms Form Plug-In for Visual Basic [Seite 1756]
- Links class [Seite 1862]
- SAPforms Form Plug-In for MS Office [Seite 1856]
- Links class [Seite 1862]
- SAPforms Form Plug-In for MS Outlook [Seite 1796]
Properties
Method that writes the content of all links to a file. Counterpart of the Load method.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Tgt</td>
<td>Stores the string structure.</td>
</tr>
</tbody>
</table>

Classification 7

Save

- SAPForm class [Seite 1772]
- SAPforms Form Plug-In for Visual Basic [Seite 1756]
- SAPForm class [Seite 1872]
- SAPforms Form Plug-In for MS Office [Seite 1856]
- SAPForm class [Seite 1812]
- SAPforms Form Plug-In for MS Outlook [Seite 1796]

Properties
Method that saves a form to a file if it was created from a file. Reads the old file and writes updated lines to the new file.

Classification 8

Save

- SAPXmit class [Seite 2106]
- SAPforms Data Plug-In for Demo [Seite 2077]
- SAPXmit class [Seite 1837]
- SAPforms Form Plug-In for Outlook [Seite 1796]
- SAPXmit class [Seite 2126]
- SAPforms Transmit Plug-In for RFC [Seite 2125]
- SAPXmit class [Seite 2168]
- SAPforms Transmit Plug-In for MAPI [Seite 2146]
- SAPXmit class [Seite 2147]
- SAPforms Transmit Plug-In for SAPMAPI [Seite 2167]
- SAPXmit class [Seite 2189]
- SAPforms Transmit Plug-In for WinSock [Seite 2188]
- SAPXmit class [Seite 2209]
- SAPforms Transmit Plug-In for Demo [Seite 2208]

Properties
Saves the current property values of an SAPXmit object in a structured string. Counterpart of the Load method.
When a form is created, the SAPforms Designer uses this method to save the structure description retrieved from the R/3 System permanently in the form.
The R/3 structure is restored from the structured string at runtime by the Load method.
### Save (Method)

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Tgt</td>
<td>String in which the status is stored.</td>
</tr>
</tbody>
</table>
Start (Method)

Definition
Method that executes the \textit{Start} function in R/3. The semantics of this function depend on the concrete SAPData object.

Remarks
In the case of the SAPData object for workflow containers, this method starts a workflow and populates its container elements with values from the form fields. This method can be called directly via the SAPData object or indirectly via the \textit{Start} method of the SAPForm object, whose SAPData property is a reference to the SAPData object. If an SAPXmit object is assigned to the SAPData object, the \textit{Start} method of the SAPXmit object is called.

Classification 1
Start
\begin{itemize}
  \item \textit{SAPData class} \cite{Seite 1918}
  \item \textit{SAPforms Data Plug-In for Workflow} \cite{Seite 1897}
\end{itemize}

Properties
Starts a workflow in the R/3 System and supplies its container elements with the data values in the fields list.

Classification 2
Start
\begin{itemize}
  \item \textit{SAPData class} \cite{Seite 1975}
  \item \textit{SAPforms Data Plug-In for Function Modules} \cite{Seite 1954}
\end{itemize}

Properties
This method carries out the current R/3 function module call.

Classification 3
Start
\begin{itemize}
  \item \textit{SAPData class} \cite{Seite 2027}
  \item \textit{SAPforms Data Plug-In for IDoc} \cite{Seite 2006}
\end{itemize}

Properties
Generates an IDoc file from the data in the segments list and transfers this IDoc to the R/3 System. The transfer method depends on the assigned SAPXmit object. A workflow can be started in the R/3 System when the inbound IDoc is processed.

Classification 4
Start
\begin{itemize}
  \item \textit{SAPData class} \cite{Seite 2078}
  \item \textit{SAPforms Data Plug-In for Demo} \cite{Seite 2077}
\end{itemize}
Properties
Does not execute an action in the R/3 System but rather displays the data of the object in a separate window.

Classification 5

Start

- SAPForm class [Seite 1772]
- SAPforms Form Plug-In for Visual Basic [Seite 1756]
- SAPForm class [Seite 1872]
- SAPforms Form Plug-In for MS Office [Seite 1856]
- SAPForm class [Seite 1812]
- SAPforms Form Plug-In for MS Outlook [Seite 1796]

Properties
Invokes the Start method in the associated SAPXmit or SAPData object. The semantics of this method depend on this SAPData object. If this object is the SAPData object for SAP Business Workflow, this method starts a workflow in an R/3 System.

Classification 6

Start

- SAPXmit class [Seite 2126]
- SAPforms Transmit Plug-In for RFC [Seite 2125]

Properties
This method starts a workflow synchronously via RFC. Supports SAPData objects of types "Idoc" and "Container".

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Data</td>
<td>Object of type SAPData.</td>
</tr>
</tbody>
</table>

Classification 7

Start

- SAPXmit class [Seite 2168]
- SAPforms Transmit Plug-In for MAPI [Seite 2146]
- SAPXmit class [Seite 2147]
- SAPforms Transmit Plug-In for SAPMAPI [Seite 2167]
- SAPXmit class [Seite 2189]
- SAPforms Transmit Plug-In for SMTP [Seite 2188]
- SAPXmit class [Seite 1837]
- SAPforms Form Plug-In for MS Outlook [Seite 1796]

Properties
Sends a mail to the R/3 System with an attachment containing the information required to start a workflow.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Data</td>
<td>Object of type SAPData.</td>
</tr>
</tbody>
</table>

April 2001
**Classification 8**

**Start**

- SAPXmit class [Seite 2209]
- SAPforms Transmit Plug-In for Demo [Seite 2208]
- SAPXmit class [Seite 2106]
- SAPforms Data Plug-In for Demo [Seite 2077]

**Properties**

Displays the instance data of the associated SAPData object in a window instead of sending it to the R/3 System.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Data</td>
<td>Object of type SAPData.</td>
</tr>
</tbody>
</table>
Submit (Method)

Definition
Calls the Start() or Execute() method, depending on the internal status of the form.

Remarks
This status must indicate whether the form was called with or without input data. If it was called with input data, the Execute() method is called. This logic is specially designed for integration in SAP Business Workflow. The Execute() method here executes a work item in the R/3 System.

Classification
Submit

← SAPForm class [Seite 1772]
← SAPforms Form Plug-In for Visual Basic [Seite 1756]
← SAPForm class [Seite 1872]
← SAPforms Form Plug-In for MS Office [Seite 1856]
← SAPForm class [Seite 1812]
← SAPforms Form Plug-In for MS Outlook [Seite 1796]
SAPData Plug-In Type

Definition
Plug-in type that is provided for various interfaces that are to be accessed in the R/3 System. The interfaces currently supported are task and workflow containers, function modules, and IDocs. The main purpose of this plug-in type is to make the interface available to form programmers and other SAPforms components in a standard manner.

SAPforms Object Model

The plug-in type SAPData in relation to the various plug-in types [Seite 1746].
Every form is used with a certain plug-in of the type *SAPData*. You must enter this plug-in in the *SAPforms-Designer [Seite 1735]*. The SAPforms Designer is shipped with the following plug-ins of this type:

<table>
<thead>
<tr>
<th>Plug-In</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>SAPforms Data Plug-In for Workflow [Seite 1897]</strong></td>
<td>Supports data exchange with the task and workflow containers in SAP Business Workflow. Use this plug-in to create a form that starts a task or executes a work item.</td>
</tr>
<tr>
<td><strong>SAPforms Data Plug-In for Function Modules [Seite 1954]</strong></td>
<td>Calls SAP function modules. Use this plug-in if you want to fill the parameter interface of a function module via the form and start this function module in the R/3 System.</td>
</tr>
<tr>
<td><strong>SAPforms Data Plug-In for IDoc [Seite 2006]</strong></td>
<td>Supports SAP Intermediate Documents (IDocs). Use this plug-in if you want to fill the segment fields of an IDoc via the form and transfer this IDoc to the R/3 System.</td>
</tr>
<tr>
<td><strong>SAPforms Data Plug-In Demo [Seite 2077]</strong></td>
<td>Plug-in for demonstration purposes (no R/3 access needed) The source code is supplied with this plug-in.</td>
</tr>
</tbody>
</table>
SAPforms Data Plug-In for Workflow

Definition
SAPData type plug-in for using workflow functions.

Use
This plug-in allows you to start a workflow in an R/3 System and supply it with data. You can also edit form-based work items from an R/3 System in a Windows application. This plug-in can be used with all SAPXmit plug-ins.

This plug-in implements the SAPData interface that provides the SAPforms Designer [Seite 1735] and other SAPforms components with access to certain types of R/3 data structures and interfaces. In other words, this plug-in can be regarded as being synonymous with the associated SAPData class.

Classification

SAPforms Data Plug-In for Workflow

→ Field Class [Seite 2007]
→ Fields Class [Seite 2018]
→ SAPData Class [Seite 1918]
Field Class

Definition
Class that describes a single element in an R/3 data structure.

Remark
This class describes
a single element in the workflow container
a single field in an IDoc segment
a single function module parameter. Appropriate properties are defined for differentiating between import, export, and tables parameters.

Classification
SAPforms Data Plug-In for Workflow [Seite 1897]
SAPforms Data Plug-In for IDoc [Seite 2006]
SAPforms Data Plug-In for Function Modules [Seite 1954]

Field Class
→Load (Method) [Seite 2213]
→Name (Property) [Seite 2088]
→Save (Method) [Seite 2219]
Load (Method)

Definition
Loads structure information to SAPforms objects.

Remarks

Classification 1

Load

Properties
Loads the property values of a SAPData object from a structured string. As a result, the status of the SAPData object that was previously saved permanently with the Save method is restored. This saves the additional overheads that would result if the structure information had to be loaded each time from the original source. This method is usually called within the Load method of the SAPForm class.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>From</td>
<td>Character string containing the structure information</td>
</tr>
</tbody>
</table>

Classification 2

Load

Properties
Loads the property values of a field object from a structured string. As a result, the status of the field object that was previously saved permanently with the Save method is restored. For this method to be called successfully, the next character at the Pos position must be keyword “Fld”.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>From</td>
<td>Structured string</td>
</tr>
<tr>
<td>Pos</td>
<td>Current position within the string. Should point to the number of remaining properties to read.</td>
</tr>
</tbody>
</table>

Classification 3

Load
Load (Method)

Properties
Loads the property values of a fields object from a structured string. As a result, the status of the fields object that was previously saved permanently with the Save method is restored.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>From</td>
<td>Character string that provides the structure information.</td>
</tr>
<tr>
<td>Pos</td>
<td>Current search position within the character string.</td>
</tr>
</tbody>
</table>

Classification 4
Load

Properties
Reconstructs the segment object from a structured string. Counterpart of the Save method.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>From</td>
<td>Character string containing structure information.</td>
</tr>
<tr>
<td>Pos</td>
<td>Current search position within the character string.</td>
</tr>
</tbody>
</table>

Classification 5
Load

Properties
This method reconstructs the segment list from a structured string. Counterpart of the Save method.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>From</td>
<td>Character string containing structure information.</td>
</tr>
<tr>
<td>Pos</td>
<td>Current search position within the character string.</td>
</tr>
</tbody>
</table>

Classification 6
Load

Properties
This method reconstructs an object of the Link class from the serialized representation. It assumes that the next token is located at the Pos position in the serialized representation of the “Fld” keyword.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>From</td>
<td>Serialization string.</td>
</tr>
</tbody>
</table>
Load (Method)

<table>
<thead>
<tr>
<th>Pos</th>
<th>Current position within the character string.</th>
</tr>
</thead>
</table>

### Classification 7

**Load**

- SAPForm class [Seite 1772]
- SAPforms Form Plug-In for Visual Basic [Seite 1756]
- SAPForm class [Seite 1812]
- SAPforms Form Plug-In for MS Outlook [Seite 1796]
- SAPForm class [Seite 1872]
- SAPforms Form Plug-In for MS Office [Seite 1856]

### Properties

Method for loading form controls into Controls Collection from a form or form template. Note that the form information is not stored in the Tag property of this object, but reconstructed each time the form is loaded. This method is called at design time and at runtime. It has to populate the Controls collection for both cases. At runtime, the method is called by the user code in the form's Load method. When the form is designed, the method is called by the SAPForms Designer or from outside SAPForms. If you are using the SAPForms Designer as a stand-alone application, the Load method is called with Mode=2. This instructs the Load method to load the form from a file. If you are using the SAPForms Designer as a plug-in, the Load method is invoked from the hosting environment and receives the proprietary form description as a parameter. The populated SAPForm object should be passed to the SAPForms Designer by the PlugIn method of the Designer class.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Frm</td>
<td>form or form template containing controls.</td>
</tr>
<tr>
<td>Mode</td>
<td>0=design time (add-in), 1=design time (file), 2=runtime.</td>
</tr>
</tbody>
</table>

### Classification 8

**Load**

- SAPXmit class [Seite 2106]
- SAPforms Data Plug-In for Demo [Seite 2077]
- SAPXmit class [Seite 1837]
- SAPforms Form Plug-In for Outlook [Seite 1796]
- SAPXmit class [Seite 2126]
- SAPforms Transmit Plug-In for RFC [Seite 2125]
- SAPXmit class [Seite 2168]
- SAPforms Transmit Plug-In for MAPI [Seite 2146]
- SAPXmit class [Seite 2147]
- SAPforms Transmit Plug-In for SAPMAPI [Seite 2167]
- SAPXmit class [Seite 2189]
- SAPforms Transmit Plug-In for WinSock [Seite 2188]
- SAPXmit class [Seite 2209]
- SAPforms Transmit Plug-In for Demo [Seite 2208]
Load (Method)

Properties
Loads the property values of an SAPXmit object from a structured string. As a result, the status of the SAPXmit object that was previously saved permanently with the Save method is restored.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>From</td>
<td>Character string containing the structure information</td>
</tr>
</tbody>
</table>
Name (Property)

Definition
Name of the object, field, segment, or control.

Remarks
This name, for example, must be used as the root in the SAP structure tree that is built by the BuildTree method.

Classification 1
Name
← SAPData class [Seite 1918]
← SAPforms Data Plug-In for Workflow [Seite 1897]
Properties
Returns the name of the workflow to whose task the container description belongs.

Classification 2
Name
← SAPData class [Seite 1975]
← SAPforms Data Plug-In for Function Modules [Seite 1954]
Properties
Name of the R/3 function module that is to be executed.

Classification 3
Name
← SAPData class [Seite 2027]
← SAPforms Data Plug-In for IDoc [Seite 2006]
Properties
Name of the IDoc as it appears in the control record.

Classification 4
Name
← SAPData class [Seite 2078]
← SAPforms Data Plug-In for Demo [Seite 2077]

Classification 5
Name
← Field Class [Seite 2007]
← SAPforms Data Plug-In for Workflow [Seite 1897]
← SAPforms Data Plug-In for Function Modules [Seite 1954]
← SAPforms Data Plug-In for IDoc [Seite 2006]
Properties
Name of the field.

Classification 6
Name

- Segment class [Seite 2055]
- SAPforms Data Plug-In for IDoc [Seite 2006]

Properties
Name of the segment.
Used as a key in the Templates list of the IDoc object. Only the first attribute with this name is added to the list of segments with a key. All other segment attributes with the same name are added without a key.

Classification 7
Name

- SAPForm class [Seite 1772]
- SAPforms Form Plug-In for Visual Basic [Seite 1756]
- SAPForm Class [Seite 1812]
- SAPforms Form Plug-In for MS Outlook [Seite 1796]
- SAPForm class [Seite 1872]
- SAPforms Form Plug-In for MS Office [Seite 1856]

Properties
Original name of the control in the real form (e.g. Visual Basic form).
Save (Method)

Definition
Saves the updated form definition.

Remarks
This method is used by the SAPforms Designer when a form is designed to save the modified form definition after the binding has been defined. The modifications can affect the binding definition or possibly the source code entered in the form.

Classification 1
Save

Properties
Saves the current property values of an SAPData object in a structured string. Counterpart of the Load method.
When a form is created, the SAPforms Designer uses this method to save the structure description retrieved from the R/3 System permanently in the form.
The R/3 structure is restored from the structured string at runtime by the Load method.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Tgt</td>
<td>String in which the status is stored.</td>
</tr>
</tbody>
</table>

Classification 2
Save

Properties
Saves the current property values of an SAPData object in a structured string. Counterpart of the Load method. The string is saved with a form so that the structure information is available when the form is executed.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Tgt</td>
<td>Stores the string structure.</td>
</tr>
</tbody>
</table>
Classification 3
Save

Loads the property values of a fields object from a structured string. As a result, the status of the fields object that was previously saved permanently with the Save method is restored.
The string is saved with the form so that the structure information is available when the form is executed.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Tgt</td>
<td>Stores the string structure.</td>
</tr>
</tbody>
</table>

Classification 4
Save

This method saves the structure of a segment, including the field list, in a structured string.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Tgt</td>
<td>Stores the string structure.</td>
</tr>
</tbody>
</table>

Classification 5
Save

This method saves the segment list in a structured string. Counterpart of the Load method.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Tgt</td>
<td>Stores the string structure.</td>
</tr>
</tbody>
</table>

Classification 6
Save

<table>
<thead>
<tr>
<th>Parameters</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Links class [Seite 1862]</td>
<td>Stores the string structure.</td>
</tr>
<tr>
<td>SAPforms Form Plug-In for Visual Basic [Seite 1756]</td>
<td></td>
</tr>
<tr>
<td>Links class [Seite 1862]</td>
<td></td>
</tr>
<tr>
<td>SAPforms Form Plug-In for MS Office [Seite 1856]</td>
<td></td>
</tr>
<tr>
<td>Links class [Seite 1862]</td>
<td></td>
</tr>
<tr>
<td>SAPforms Form Plug-In for MS Outlook [Seite 1796]</td>
<td></td>
</tr>
</tbody>
</table>
Properties
Method that writes the content of all links to a file. Counterpart of the Load method.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Tgt</td>
<td>Stores the string structure.</td>
</tr>
</tbody>
</table>

Classification 7

Save

- SAPForm class [Seite 1772]
- SAPforms Form Plug-In for Visual Basic [Seite 1756]
- SAPForm class [Seite 1872]
- SAPforms Form Plug-In for MS Office [Seite 1856]
- SAPForm class [Seite 1812]
- SAPforms Form Plug-In for MS Outlook [Seite 1796]

Properties
Method that saves a form to a file if it was created from a file.
Reads the old file and writes updated lines to the new file.

Classification 8

Save

- SAPXmit class [Seite 2106]
- SAPforms Data Plug-In for Demo [Seite 2077]
- SAPXmit class [Seite 1837]
- SAPforms Form Plug-In for Outlook [Seite 1796]
- SAPXmit class [Seite 2126]
- SAPforms Transmit Plug-In for RFC [Seite 2125]
- SAPXmit class [Seite 2168]
- SAPforms Transmit Plug-In for MAPI [Seite 2146]
- SAPXmit class [Seite 2147]
- SAPforms Transmit Plug-In for SAPMAPI [Seite 2167]
- SAPXmit class [Seite 2189]
- SAPforms Transmit Plug-In for WinSock [Seite 2188]
- SAPXmit class [Seite 2209]
- SAPforms Transmit Plug-In for Demo [Seite 2208]

Properties
Saves the current property values of an SAPXmit object in a structured string. Counterpart of the Load method.
When a form is created, the SAPforms Designer uses this method to save the structure description retrieved from the R/3 System permanently in the form.
The R/3 structure is restored from the structured string at runtime by the Load method.
<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Tgt</td>
<td>String in which the status is stored.</td>
</tr>
</tbody>
</table>
Fields Class

Definition
List class that describes the field list in an R/3 data structure.

Remark
This class can contain the following lists:
elements in a workflow container
segment fields in an IDoc
function module parameters.
The individual elements in the list can be accessed with the Item property or with a For ... Each loop. The R/3 data structures often represent hierarchies. To identify a field in one of these data structures, therefore, you must specify the fully name of this field.

```
... 
set objFld = SAPData.Fields.Item ("[StruCturX].[FieldY]") 
...
```

Classification
SAPforms Data Plug-In for Workflow [Seite 1897]
SAPforms Data Plug-In for IDoc [Seite 2006]
SAPforms Data Plug-In for Function Modules [Seite 1954]

→ Fields Class
  → Load (Method) [Seite 2213]
  → Save (Method) [Seite 2219]
Load (Method)

Definition
Loads structure information to SAPforms objects.

Remarks

Classification 1
Load

Properties
 Loads the property values of an SAPData object from a structured string. As a result, the status of the
 SAPData object that was previously saved permanently with the Save method is restored.
 This saves the additional overheads that would result if the structure information had to be loaded each
time from the original source. This method is usually called within the Load method of the SAPForm class.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>From</td>
<td>Character string containing the structure information</td>
</tr>
</tbody>
</table>

Classification 2
Load

Properties
 Loads the property values of a field object from a structured string. As a result, the status of the field object
that was previously saved permanently with the Save method is restored.
 For this method to be called successfully, the next character at the Pos position must be keyword “Fld”.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>From</td>
<td>Structured string</td>
</tr>
<tr>
<td>Pos</td>
<td>Current position within the string. Should point to the number of</td>
</tr>
<tr>
<td></td>
<td>remaining properties to read.</td>
</tr>
</tbody>
</table>

Classification 3
Load
Load (Method)

Properties
Loads the property values of a fields object from a structured string. As a result, the status of the fields object that was previously saved permanently with the Save method is restored.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>From</td>
<td>Character string that provides the structure information.</td>
</tr>
<tr>
<td>Pos</td>
<td>Current search position within the character string.</td>
</tr>
</tbody>
</table>

Classification 4
Load

Properties
Reconstructs the segment object from a structured string. Counterpart of the Save method.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>From</td>
<td>Character string containing structure information.</td>
</tr>
<tr>
<td>Pos</td>
<td>Current search position within the character string.</td>
</tr>
</tbody>
</table>

Classification 5
Load

Properties
This method reconstructs the segment list from a structured string. Counterpart of the Save method.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>From</td>
<td>Character string containing structure information.</td>
</tr>
<tr>
<td>Pos</td>
<td>Current search position within the character string.</td>
</tr>
</tbody>
</table>

Classification 6
Load

Properties
This method reconstructs an object of the Link class from the serialized representation. It assumes that the next token is located at the Pos position in the serialized representation of the “Fld” keyword.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>From</td>
<td>Serialization string.</td>
</tr>
</tbody>
</table>
Load (Method)

Classification 7

Load

กรีSAPForm class [Seite 1772](225,439),(391,455)

(225,455),(391,461)

(225,462),(391,468)

(225,469),(391,475)

(225,476),(391,482)

Properties

Method for loading form controls into Controls Collection from a form or form template. Note that the form information is not stored in the Tag property of this object, but reconstructed each time the form is loaded. This method is called at design time and at runtime. It has to populate the Controls collection for both cases. At runtime, the method is called by the user code in the form's Load method. When the form is designed, the method is called by the SAPForms Designer or from outside SAPForms. If you are using the SAPForms Designer as a stand-alone application, the Load method is called with Mode=2. This instructs the Load method to load the form from a file. If you are using the SAPForms Designer as a plug-in, the Load method is invoked from the hosting environment and receives the proprietary form description as a parameter. The populated SAPForm object should be passed to the SAPForms Designer by the PlugIn method of the Designer class.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Frm</td>
<td>form or form template containing controls.</td>
</tr>
<tr>
<td>Mode</td>
<td>0=design time (add-in), 1=design time (file), 2=runtime.</td>
</tr>
</tbody>
</table>

Classification 8

Load

(225,666),(391,682)

(225,683),(391,689)

(225,690),(391,696)

(225,697),(391,703)

(225,704),(391,710)

(225,711),(391,717)

(225,718),(391,724)

(225,725),(391,731)

(225,732),(391,738)

(225,739),(391,745)

(225,746),(391,752)

(225,753),(391,759)

(225,760),(391,766)

(225,767),(391,773)

(225,774),(391,780)

(225,781),(391,787)

(225,788),(391,794)

(225,795),(391,801)

(225,802),(391,808)

(225,809),(391,815)

(225,816),(391,822)

(225,823),(391,829)

(225,830),(391,836)

(225,837),(391,843)

(225,844),(391,850)

(225,851),(391,857)

(225,858),(391,864)

(225,865),(391,871)

(225,872),(391,878)

(225,879),(391,885)

(225,886),(391,892)

(225,893),(391,900)

(225,901),(391,907)

(225,908),(391,914)

(225,915),(391,921)

(225,922),(391,928)

(225,929),(391,935)

(225,936),(391,942)

(225,943),(391,949)

(225,950),(391,956)

(225,957),(391,963)

(225,964),(391,970)

(225,971),(391,977)

(225,978),(391,984)

(225,985),(391,991)

(225,992),(391,998)

(225,999),(391,1005)
Properties
Loads the property values of an SAPXmit object from a structured string. As a result, the status of the SAPXmit object that was previously saved permanently with the Save method is restored.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>From</td>
<td>Character string containing the structure information</td>
</tr>
</tbody>
</table>
Save (Method)

Definition
Saves the updated form definition.

Remarks
This method is used by the SAPforms Designer when a form is designed to save the modified form definition after the binding has been defined. The modifications can affect the binding definition or possibly the source code entered in the form.

Classification 1
Save

Properties
Saves the current property values of an SAPData object in a structured string. Counterpart of the Load method. When a form is created, the SAPforms Designer uses this method to save the structure description retrieved from the R/3 System permanently in the form. The R/3 structure is restored from the structured string at runtime by the Load method.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Tgt</td>
<td>String in which the status is stored.</td>
</tr>
</tbody>
</table>

Classification 2
Save

Properties
Saves the current property values of an SAPData object in a structured string. Counterpart of the Load method. The string is saved with a form so that the structure information is available when the form is executed.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Tgt</td>
<td>Stores the string structure.</td>
</tr>
</tbody>
</table>
Save (Method)

Classification 3
Save

🔗 Fields Class [Seite 2018]
🔗 SAPforms Data Plug-In for Workflow [Seite 1897]
🔗 SAPforms Data Plug-In for Function Modules [Seite 1954]
🔗 SAPforms Data Plug-In for IDoc [Seite 2006]

Properties
Loads the property values of a fields object from a structured string. As a result, the status of the fields object that was previously saved permanently with the Save method is restored.
The string is saved with the form so that the structure information is available when the form is executed.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Tgt</td>
<td>Stores the string structure.</td>
</tr>
</tbody>
</table>

Classification 4
Save

🔗 Segment class [Seite 2055]
🔗 SAPforms Data Plug-In for IDoc [Seite 2006]

Properties
This method saves the structure of a segment, including the field list, in a structured string.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Tgt</td>
<td>Stores the string structure.</td>
</tr>
</tbody>
</table>

Classification 5
Save

🔗 Segments class [Seite 2067]
🔗 SAPforms Data Plug-In for IDoc [Seite 2006]

Properties
This method saves the segment list in a structured string. Counterpart of the Load method.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Tgt</td>
<td>Stores the string structure.</td>
</tr>
</tbody>
</table>

Classification 6
Save

🔗 Links class [Seite 1862]
🔗 SAPforms Form Plug-In for Visual Basic [Seite 1756]
🔗 Links class [Seite 1862]
🔗 SAPforms Form Plug-In for MS Office [Seite 1856]
🔗 Links class [Seite 1862]
🔗 SAPforms Form Plug-In for MS Outlook [Seite 1796]
Properties

Method that writes the content of all links to a file. Counterpart of the Load method.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Tgt</td>
<td>Stores the string structure.</td>
</tr>
</tbody>
</table>

Classification 7

Save

- SAPForm class [Seite 1772]
- SAPforms Form Plug-In for Visual Basic [Seite 1756]
- SAPForm class [Seite 1872]
- SAPforms Form Plug-In for MS Office [Seite 1856]
- SAPForm class [Seite 1812]
- SAPforms Form Plug-In for MS Outlook [Seite 1796]

Properties

Method that saves a form to a file if it was created from a file.
Reads the old file and writes updated lines to the new file.

Classification 8

Save

- SAPXmit class [Seite 2106]
- SAPforms Data Plug-In for Demo [Seite 2077]
- SAPXmit class [Seite 1837]
- SAPforms Form Plug-In for Outlook [Seite 1796]
- SAPXmit class [Seite 2126]
- SAPforms Transmit Plug-In for RFC [Seite 2125]
- SAPXmit class [Seite 2168]
- SAPforms Transmit Plug-In for MAPI [Seite 2146]
- SAPXmit class [Seite 2147]
- SAPforms Transmit Plug-In for SAPMAPI [Seite 2167]
- SAPXmit class [Seite 2189]
- SAPforms Transmit Plug-In for WinSock [Seite 2188]
- SAPXmit class [Seite 2209]
- SAPforms Transmit Plug-In for Demo [Seite 2208]

Properties

Saves the current property values of an SAPXmit object in a structured string. Counterpart of the Load method.

When a form is created, the SAPforms Designer uses this method to save the structure description retrieved from the R/3 System permanently in the form.
The R/3 structure is restored from the structured string at runtime by the Load method.
<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Tgt</td>
<td>String in which the status is stored.</td>
</tr>
</tbody>
</table>
**SAPData Class**

**Definition**
Original class of this plug-in; representation of an SAP work item.

**Remarks**
You can use this class to edit the container data of a workflow or work item in a form. The *Fields* property refers to an object in the *Fields* list class that represents the container elements. The elements themselves can be accessed with the *Value* property. The important methods here are the *Start* method, which can be used to start a workflow, and the *Execute* method that executes work items.

SAPData plug-in type classes implement the SAPData interface that provides the SAPforms Designer and other SAPforms components with access to certain types of R/3 data structures and interfaces. In other words, these classes can be regarded as being synonymous with various plug-ins of this type.

A SAPData object is usually also instantiated with the SAPForm object. The SAPData object is then referenced by the SAPData property of the SAPForm object. An object of the SAPData class, however, cannot exist without an object of the SAPForms class.

If you want to develop your own SAPData-type plug-ins, you should use the source code supplied for the *SAPforms Data Plug-Ins for Demo* as a reference (template). The template is stored in the directory `...\sapforms\sapddemo`.

**Classification**
*SAPforms Data Plug-In for Workflow* [Seite 1897]

- **SAPData Class**
  - *Connection* (Property) [Seite 2028]
  - *Execute* (Method) [Seite 2210]
  - *Fields* (Property) [Seite 2056]
  - *Load* (Method) [Seite 2213]
  - *LoadInstance* (Method) [Seite 2086]
  - *LogIt* (Method) [Seite 1931]
  - *Name* (Property) [Seite 2088]
  - *Offline* (Property) [Seite 1934]
  - *Options* (Method) [Seite 2217]
  - *R3DoNotUse* (Property) [Seite 2137]
  - *Receiver* (Property) [Seite 2092]
  - *SAPXmit* (property) [Seite 2093]
  - *Save* (Method) [Seite 2219]
  - *SaveInstance* (Method) [Seite 2098]
  - *Start* (Method) [Seite 2223]
  - *SuppressSystemSelection* (Property) [Seite 1948]
  - *UpdateInstance* (Method) [Seite 2102]
  - *Value* (Property) [Seite 2103]
→ Version (Property) [Seite 2105)
→ WorkItemID (Property) [Seite 1953]
Connection (Property)

Definition
Sets or returns the object of the RFC connection.

Remarks

Classification
Connection

← SAPData class [Seite 1918]
← SAPforms Data Plug-In for Workflow [Seite 1897]
← SAPData class [Seite 1975]
← SAPforms Data Plug-In for Function Modules [Seite 1954]
← SAPData class [Seite 2027]
← SAPforms Data Plug-In for IDoc [Seite 2006]
Execute (Method)

Definition
Method that calls the *Execute* function in the R/3 System. The semantics of this function depend on the SAPData object.

Remarks
The method can be called directly via the SAPData object or indirectly via the Start method of the SAPForm object, whose SAPData property is a reference to the SAPData object.

In the case of the SAPData object for workflow containers, for example, this method executes a work item in an SAP Business Workflow and populates its container elements with values from the form fields.

Classification 1

Execute

- SAPData class [Seite 1918]
- SAPforms Data Plug-In for Workflow [Seite 1897]

Properties
Terminates a work item in the R/3 System and supplies its container elements with the data values in the fields list.

Classification 2

Execute

- SAPData class [Seite 2078]
- SAPXmit class [Seite 2106]
- SAPforms Data Plug-In for Demo [Seite 2077]

Properties
Does not execute an action in the R/3 System but rather displays the data of the object in a separate window.

Classification 3

Execute

- SAPForm class [Seite 1772]
- SAPforms Form Plug-In for Visual Basic [Seite 1756]
- SAPForm class [Seite 1872]
- SAPforms Form Plug-In for MS Office [Seite 1856]
- SAPForm class [Seite 1812]
- SAPforms Form Plug-In for MS Outlook [Seite 1796]
Properties
Invokes the Execute method in the associated Xmit or SAPData object. The semantics of this method depend on this object.

Classification 4
Execute

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Data</td>
<td>Object representing the container data.</td>
</tr>
</tbody>
</table>

Properties
This method executes a work item synchronously via a remote function call (RFC). Supports SAPData object of type Container.
This method causes a certain function module to be executed in the R/3 System. This module receives a table with a complete R3F structure containing the work item information and the container data that are to be supplied from the form.

Classification 5
Execute

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Data</td>
<td>Object representing the container data.</td>
</tr>
</tbody>
</table>

Properties
Sends a mail to the R/3 System that contains the information required to terminate a work item (including container data) as an attachment.

Classification 6
Execute

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Pdata</td>
<td>Object that represents the mobjData container data</td>
</tr>
</tbody>
</table>

Properties
Overwrites the R3F attachment, thereby updating it.

Classification 7
Execute

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Properties
Displays the instance data of the associated SAPData object in a window instead of sending it to the R/3 System.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Data</td>
<td>object of class SAPData</td>
</tr>
</tbody>
</table>
Fields (Property)

Definition
List of the elements in an R/3 Structure, i.e.
the container elements in the workflow or work item that is represented by the SAPData object
the parameters contained in the function module
the fields contained in the IDoc segment.

Remarks
Classification
Fields

← SAPData class [Seite 1918]
← SAPforms Data Plug-In for Workflow [Seite 1897]
← SAPData class [Seite 1975]
← SAPforms Data Plug-In for Function Modules [Seite 1954]
← SAPData class [Seite 2027]
← SAPforms Data Plug-In for IDoc [Seite 2006]
Load (Method)

Definition
Loads structure information to SAPforms objects.

Remarks
Classification 1

Load

Properties
Loads the property values of an SAPData object from a structured string. As a result, the status of the
SAPData object that was previously saved permanently with the Save method is restored.
This saves the additional overheads that would result if the structure information had to be loaded each
time from the original source. This method is usually called within the Load method of the SAPForm class.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>From</td>
<td>Character string containing the structure information</td>
</tr>
</tbody>
</table>

Classification 2

Load

Properties
Loads the property values of a field object from a structured string. As a result, the status of the field object
that was previously saved permanently with the Save method is restored.
For this method to be called successfully, the next character at the Pos position must be keyword “Fld”.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>From</td>
<td>Structured string</td>
</tr>
<tr>
<td>Pos</td>
<td>Current position within the string. Should point to the number of remaining properties to read</td>
</tr>
</tbody>
</table>

Classification 3

Load
Load (Method)

Properties
Loads the property values of a fields object from a structured string. As a result, the status of the fields object that was previously saved permanently with the Save method is restored.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>From</td>
<td>Character string that provides the structure information.</td>
</tr>
<tr>
<td>Pos</td>
<td>Current search position within the character string.</td>
</tr>
</tbody>
</table>

Classification 4

Load

Properties
Reconstructs the segment object from a structured string. Counterpart of the Save method.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>From</td>
<td>Character string containing structure information.</td>
</tr>
<tr>
<td>Pos</td>
<td>Current search position within the character string.</td>
</tr>
</tbody>
</table>

Classification 5

Load

Properties
This method reconstructs the segment list from a structured string. Counterpart of the Save method.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>From</td>
<td>Character string containing structure information.</td>
</tr>
<tr>
<td>Pos</td>
<td>Current search position within the character string.</td>
</tr>
</tbody>
</table>

Classification 6

Load

Properties
This method reconstructs an object of the Link class from the serialized representation. It assumes that the next token is located at the Pos position in the serialized representation of the “Fld” keyword.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>From</td>
<td>Serialization string.</td>
</tr>
</tbody>
</table>
**Classification 7**

**Load**

- SAPForm class [Seite 1772]
- SAPforms Form Plug-In for Visual Basic [Seite 1756]
- SAPForm class [Seite 1812]
- SAPforms Form Plug-In for MS Outlook [Seite 1796]
- SAPForm class [Seite 1872]
- SAPforms Form Plug-In for MS Office [Seite 1856]

**Properties**

Method for loading form controls into Controls Collection from a form or form template. Note that the form information is not stored in the Tag property of this object, but reconstructed each time the form is loaded. This method is called at design time and at runtime. It has to populate the Controls collection for both cases. At runtime, the method is called by the user code in the form's Load method. When the form is designed, the method is called by the SAPForms Designer or from outside SAPForms. If you are using the SAPForms Designer as a stand-alone application, the Load method is called with Mode=2. This instructs the Load method to load the form from a file. If you are using the SAPForms Designer as a plug-in, the Load method is invoked from the hosting environment and receives the proprietary form description as a parameter. The populated SAPForm object should be passed to the SAPForms Designer by the PlugIn method of the Designer class.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Frm</td>
<td>form or form template containing controls.</td>
</tr>
<tr>
<td>Mode</td>
<td>0=design time (add-in), 1=design time (file), 2=runtime.</td>
</tr>
</tbody>
</table>

**Classification 8**

**Load**

- SAPXmit class [Seite 2106]
- SAPforms Data Plug-In for Demo [Seite 2077]
- SAPXmit class [Seite 1837]
- SAPforms Form Plug-In for Outlook [Seite 1796]
- SAPXmit class [Seite 2126]
- SAPforms Transmit Plug-In for RFC [Seite 2125]
- SAPXmit class [Seite 2168]
- SAPforms Transmit Plug-In for MAPI [Seite 2146]
- SAPXmit class [Seite 2147]
- SAPforms Transmit Plug-In for SAPMAPI [Seite 2167]
- SAPXmit class [Seite 2189]
- SAPforms Transmit Plug-In for WinSock [Seite 2188]
- SAPXmit class [Seite 2209]
- SAPforms Transmit Plug-In for Demo [Seite 2208]
Properties
Loads the property values of an SAPXmit object from a structured string. As a result, the status of the
SAPXmit object that was previously saved permanently with the Save method is restored.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>From</td>
<td>Character string containing the structure information</td>
</tr>
</tbody>
</table>
LoadInstance (Method)

Definition
Method that loads the data content of a SAPData object from a file.

Remarks
The method is usually called when data flows from the R/3 System to the form and is to be displayed by the form. This is the case, for example, when a work item is executed. The form is then started by the SAPforms Helper [Seite 1741] and the R3F file is forwarded as a parameter. The existence of a parameter when the form is started ensures that the LoadInstance method is called in the Init() method of the SAPform object. The GetData method is usually called after the LoadInstance method.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Path</td>
<td>Complete path designation</td>
</tr>
</tbody>
</table>

Classification 1
LoadInstance

Properties
Loads the R/3 workflow information from an R3F file to the SAPData object. The current container element values can be interrogated and manipulated via the SAPData.Value and SAPData.Fields(s).Data properties in particular.

Classification 2
LoadInstance

Properties
Implementation of the SAPData interface for R/3 function modules (BAPIs).

Classification 3
LoadInstance

Properties
Loads an IDoc instance from a file.

Classification 4
LoadInstance

Builds the segments list from the records in the file. This method will only function correctly if the Templates list containing the entire IDoc meta structure exists. The structure of each segment in the Segments list is derived from the corresponding segment in the Templates list.
Properties
The method saves the contents of a field in the FieldValues array. This method requires that the elements be saved in a fixed order. Even though the field name is contained in the file, it is not evaluated.
LogIt (Method)

Definition
Allows form-specific data to be written to the Trace Log Viewer [Seite 1742].

Remarks

Classification

LogIt

← SAPData [Seite 1918]
← SAPforms Data Plug-In for Workflow [Seite 1897]
← SAPForm [Seite 1772]
← SAPforms Form Plug-In for Visual Basic [Seite 1756]
← SAPForm [Seite 1812]
← SAPforms Form Plug-In for MS Outlook [Seite 1796]
← SAPForm [Seite 1872]
← SAPforms Form Plug-In for MS Office [Seite 1856]
Name (Property)

Definition
Name of the object, field, segment, or control.

Remarks
This name, for example, must be used as the root in the SAP structure tree that is built by the `BuildTree` method.

Classification 1
Name
Name

Properties
Returns the name of the workflow to whose task the container description belongs.

Classification 2
Name

Properties
Name of the R/3 function module that is to be executed.

Classification 3
Name

Properties
Name of the IDoc as it appears in the control record.

Classification 4
Name

Classification 5
Name
**Properties**

Name of the field.

**Classification 6**

**Name**

- Segment class [Seite 2055]
- SAPforms Data Plug-In for IDoc [Seite 2006]

**Properties**

Name of the segment.

Used as a key in the Templates list of the IDoc object. Only the first attribute with this name is added to the list of segments with a key. All other segment attributes with the same name are added without a key.

**Classification 7**

**Name**

- SAPForm class [Seite 1772]
- SAPforms Form Plug-In for Visual Basic [Seite 1756]
- SAPForm Class [Seite 1812]
- SAPforms Form Plug-In for MS Outlook [Seite 1796]
- SAPForm class [Seite 1872]
- SAPforms Form Plug-In for MS Office [Seite 1856]

**Properties**

Original name of the control in the real form (e.g. Visual Basic form).
**Offline (Property)**

**Definition**
Property that is false when the R/3 System is accessed online.

**Remarks**

**Classification**
Offline

←SAPData class [Seite 1918]
←SAPforms Data Plug-In for Workflow [Seite 1897]
Options (Method)

Definition
Method that outputs the object-specific options dialog box.

Remarks
This dialog box displays either global settings or settings that are valid for the current form. These settings can be changed here. The global settings are read from the registry before they are displayed and are then saved again to the registry.
The method is called within the SAPforms Designer if the required plug-in is active under Properties on the Plug-Ins tab in the Options dialog. You can also use the sample program Options.exe to display the Options dialog of a certain plug-in.

Classification 1
Options

- SAPData class [Seite 1918]
- SAPforms Data Plug-In for Workflow [Seite 1897]
- SAPData class [Seite 1975]
- SAPforms Data Plug-In for Function Modules [Seite 1954]
- SAPData class [Seite 2027]
- SAPforms Data Plug-In for lDoc [Seite 2006]
- SAPData class [Seite 2078]
- SAPforms Data Plug-In for Demo [Seite 2077]

Classification 2
Options

- SAPForm class [Seite 1772]
- SAPforms Form Plug-In for Visual Basic [Seite 1756]
- SAPForm class [Seite 1812]
- SAPforms Form Plug-In for MS Outlook [Seite 1796]
- SAPForm class [Seite 1872]
- SAPforms Form Plug-In for MS Office [Seite 1856]

Classification 3
Options

- SAPXmit class [Seite 2106]
- SAPforms Data Plug-In for Demo [Seite 2077]
- SAPXmit class [Seite 1837]
- SAPforms Form Plug-In for MS Outlook [Seite 1796]
- SAPXmit class [Seite 2126]
SAPforms Transmit Plug-In for RFC [Seite 2125]
SAPXmit class [Seite 2168]
SAPforms Transmit Plug-In for MAPI [Seite 2146]
SAPXmit class [Seite 2147]
SAPforms Transmit Plug-In for SAPMAPI [Seite 2167]
SAPXmit class [Seite 2189]
SAPforms Transmit Plug-In for SMTP [Seite 2188]
SAPXmit class [Seite 2209]
SAPforms Transmit Plug-In for Demo [Seite 2208]
R3DoNotUse (Property)

Definition
Determines whether the R/3 system information, which was stored permanently in the form when it was designed, is to be used when the Start() and Execute() methods are executed.

Remarks

Classification
R3DoNotUse

Properties
The default value of this field is FALSE. When the methods are executed, logon takes place to the R/3 System that was used to design the form. This R/3 System selection is skipped during logon. If the same form is to be used with different R/3 Systems, you must set this property to TRUE. You can specify this either in the form's program code or by using the options dialog of the SAPData object.
**Receiver (Property)**

**Definition**
Recipient of the message from the perspective of the information creator.

**Remarks**

**Classification**

Receiver

- [SAPData class][Seite 1918]
- [SAPforms Data Plug-In for Workflow][Seite 1897]
- [SAPData class][Seite 2078]
- [SAPforms Data Plug-In for Demo][Seite 2077]

**Properties**

This property corresponds to the property with the same name in an R3F structure. When an R3F structure from the R/3 System is imported in the form, this property receives the mail address of the agent. This property is changed in the *SaveInstance* method and then receives the mail address of the receiver in the R/3 System.
**SAPXmit (Property)**

**Definition**
Reference to the SAPXmit object that is used to encapsulate the transport of SAPData to the R/3 System.

**Remarks**

**Classification**

**SAPXmit**

- SAPData class [Seite 1918]
- SAPforms Data Plug-In for Workflow [Seite 1897]
- SAPData class [Seite 1975]
- SAPforms Data Plug-In for Function Modules [Seite 1954]
- SAPData class [Seite 2027]
- SAPforms Data Plug-In for IDoc [Seite 2006]
- SAPData class [Seite 2078]
- SAPforms Data Plug-In for Demo [Seite 2077]
- SAPForm class [Seite 1772]
- SAPforms Form Plug-In for Visual Basic [Seite 1756]
- SAPForm class [Seite 1812]
- SAPforms Form Plug-In for MS Outlook [Seite 1796]
- SAPForm class [Seite 1872]
- SAPforms Form Plug-In for MS Office [Seite 1856]
Save (Method)

Definition
Saves the updated form definition.

Remarks
This method is used by the SAPforms Designer when a form is designed to save the modified form definition after the binding has been defined. The modifications can affect the binding definition or possibly the source code entered in the form.

Classification 1

Save

Properties
Saves the current property values of an SAPData object in a structured string. Counterpart of the Load method.

When a form is created, the SAPforms Designer uses this method to save the structure description retrieved from the R/3 System permanently in the form.

The R/3 structure is restored from the structured string at runtime by the Load method.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Tgt</td>
<td>String in which the status is stored.</td>
</tr>
</tbody>
</table>

Classification 2

Save

Properties
Saves the current property values of an SAPData object in a structured string. Counterpart of the Load method. The string is saved with a form so that the structure information is available when the form is executed.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Tgt</td>
<td>Stores the string structure.</td>
</tr>
</tbody>
</table>
Classification 3

Save

Properties

 Classification 3

 classification 3

 Fields Class [Seite 2018]

 SAPforms Data Plug-In for Workflow [Seite 1897]

 SAPforms Data Plug-In for Function Modules [Seite 1954]

 SAPforms Data Plug-In for IDoc [Seite 2006]

Load the property values of a fields object from a structured string. As a result, the status of the fields object that was previously saved permanently with the Save method is restored.

The string is saved with the form so that the structure information is available when the form is executed.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Tgt</td>
<td>Stores the string structure.</td>
</tr>
</tbody>
</table>

Classification 4

Save

Properties

 Classification 4

 segmentation class [Seite 2055]

 SAPforms Data Plug-In for IDoc [Seite 2006]

This method saves the structure of a segment, including the field list, in a structured string.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Tgt</td>
<td>Stores the string structure.</td>
</tr>
</tbody>
</table>

Classification 5

Save

Properties

 Classification 5

 segments class [Seite 2067]

 SAPforms Data Plug-In for IDoc [Seite 2006]

This method saves the segment list in a structured string. Counterpart of the Load method.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Tgt</td>
<td>Stores the string structure.</td>
</tr>
</tbody>
</table>

Classification 6

Save

Properties

 Classification 6

 link class [Seite 1862]

 SAPforms Form Plug-In for Visual Basic [Seite 1756]

 SAPforms Form Plug-In for MS Office [Seite 1856]

 SAPforms Form Plug-In for MS Outlook [Seite 1796]
Properties
Method that writes the content of all links to a file. Counterpart of the Load method.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Tgt</td>
<td>Stores the string structure.</td>
</tr>
</tbody>
</table>

Classification 7
Save

← SAPForm class [Seite 1772]
← SAPforms Form Plug-In for Visual Basic [Seite 1756]
← SAPForm class [Seite 1872]
← SAPforms Form Plug-In for MS Office [Seite 1856]
← SAPForm class [Seite 1812]
← SAPforms Form Plug-In for MS Outlook [Seite 1796]

Properties
Method that saves a form to a file if it was created from a file.
Reads the old file and writes updated lines to the new file.

Classification 8
Save

← SAPXmit class [Seite 2106]
← SAPforms Data Plug-In for Demo [Seite 2077]
← SAPXmit class [Seite 1837]
← SAPforms Form Plug-In for Outlook [Seite 1796]
← SAPXmit class [Seite 2126]
← SAPforms Transmit Plug-In for RFC [Seite 2125]
← SAPXmit class [Seite 2168]
← SAPforms Transmit Plug-In for MAPI [Seite 2146]
← SAPXmit class [Seite 2147]
← SAPforms Transmit Plug-In for SAPMAPI [Seite 2167]
← SAPXmit class [Seite 2189]
← SAPforms Transmit Plug-In for WinSock [Seite 2188]
← SAPXmit class [Seite 2209]
← SAPforms Transmit Plug-In for Demo [Seite 2208]

Properties
Saves the current property values of an SAPXmit object in a structured string. Counterpart of the Load method.

When a form is created, the SAPforms Designer uses this method to save the structure description retrieved from the R/3 System permanently in the form.
The R/3 structure is restored from the structured string at runtime by the Load method.
<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Tgt</td>
<td>String in which the status is stored.</td>
</tr>
</tbody>
</table>
SaveInstance (Method)

Definition
This method saves the current data content of the SAPData object to the specified file.

Remarks
This data content of the SAPData object is the result of the PutData method of the SAPForm object or the result of assignments to the SAPData method in the SAPData object. The method is usually called from the SAPXmit object to save the data content in a file before it is sent by email, for example.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Path</td>
<td>Complete path specifying where the data content is to be saved.</td>
</tr>
</tbody>
</table>

Classification 1
SaveInstance

Properties
Method that saves the current contents of the function module to the specified file in R3F format.

Classification 2
SaveInstance

Properties
Saves the current IDoc instance segment-by-segment to the specified file. Uses the Record property of the segment object to generate a record for a segment.

Classification 3
SaveInstance

Properties
Generates a file containing a separate line for each data field.
Start (Method)

Definition
Method that executes the Start function in R/3. The semantics of this function depend on the concrete SAPData object.

Remarks
In the case of the SAPData object for workflow containers, this method starts a workflow and populates its container elements with values from the form fields. This method can be called directly via the SAPData object or indirectly via the Start method of the SAPForm object, whose SAPData property is a reference to the SAPData object. If an SAPXmit object is assigned to the SAPData object, the Start method of the SAPXmit object is called.

Classification 1
Start

Properties
Starts a workflow in the R/3 System and supplies its container elements with the data values in the fields list.

Classification 2
Start

Properties
This method carries out the current R/3 function module call.

Classification 3
Start

Properties
Generates an IDoc file from the data in the segments list and transfers this IDoc to the R/3 System. The transfer method depends on the assigned SAPXmit object. A workflow can be started in the R/3 System when the inbound IDoc is processed.

Classification 4
Start
**Properties**
Does not execute an action in the R/3 System but rather displays the data of the object in a separate window.

**Classification 5**

**Start**

- SAPForm class [Seite 1772]
- SAPforms Form Plug-In for Visual Basic [Seite 1756]
- SAPForm class [Seite 1872]
- SAPforms Form Plug-In for MS Office [Seite 1856]
- SAPForm class [Seite 1812]
- SAPforms Form Plug-In for MS Outlook [Seite 1796]

**Properties**
Invokes the *Start* method in the associated SAPXmit or SAPData object. The semantics of this method depend on this SAPData object. If this object is the *SAPData object for SAP Business Workflow*, this method starts a workflow in an R/3 System.

**Classification 6**

**Start**

- SAPXmit class [Seite 2126]
- SAPforms Transmit Plug-In for RFC [Seite 2125]

**Properties**
This method starts a workflow synchronously via RFC. Supports SAPData objects of types "Idoc" and "Container".

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Data</td>
<td>Object of type SAPData.</td>
</tr>
</tbody>
</table>

**Classification 7**

**Start**

- SAPXmit class [Seite 2168]
- SAPforms Transmit Plug-In for MAPI [Seite 2146]
- SAPXmit class [Seite 2147]
- SAPforms Transmit Plug-In for SAPMAPI [Seite 2167]
- SAPXmit class [Seite 2189]
- SAPforms Transmit Plug-In for SMTP [Seite 2188]
- SAPXmit class [Seite 1837]
- SAPforms Form Plug-In for MS Outlook [Seite 1796]

**Properties**
Sends a mail to the R/3 System with an attachment containing the information required to start a workflow.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Data</td>
<td>Object of type SAPData.</td>
</tr>
</tbody>
</table>
Classification 8

Start

- SAPXmit class [Seite 2209]
- SAPforms Transmit Plug-In for Demo [Seite 2208]
- SAPXmit class [Seite 2106]
- SAPforms Data Plug-In for Demo [Seite 2077]

Properties

Displays the instance data of the associated SAPData object in a window instead of sending it to the R/3 System.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Data</td>
<td>Object of type SAPData.</td>
</tr>
</tbody>
</table>
SuppressSystemSelection (Property)

Definition
Property that is “true” if a system selection is to be suppressed during the R/3 logon procedure.

Remarks

Classification
SuppressSystemSelection

← SAPData class [Seite 1918]
← SAPforms Data Plug-In for Workflow [Seite 1897]
UpdateInstance (Method)

Definition
Updates the fields in the R3F file.

Remarks

Classification 1

UpdateInstance

Properties
Updates the R3F file that was read by the last LoadInstance call.
Unlike the SaveInstance method, this method is designed to update R3F attachments in incoming messages.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Path</td>
<td>Fully qualified target path of storage procedure</td>
</tr>
</tbody>
</table>

Classification 2

UpdateInstance

Properties
Saves the modified form data to the R3F attachment in a mail.
Value (Property)

Definition
Value of a specific field type.

Remarks

Classification 1

Value

Properties
The value of an SAP data field, defined in greater detail by the Qual parameter.
The structure and syntax of the Qual parameter depends on the concrete SAP data structure from which this field originates. In the simplest case, the Qual parameter designates the name of a field whose value is to be set or read. If the field is part of a structure (for example, an IDoc segment), Qual must identify the field uniquely within the overall structure. If the structure is repeated within an instance - as is the case with some IDocs - Qual must also contain a selector that identifies the IDoc segment uniquely.

```
obj.Value("Field1") = "abc".
Obj.Value("Seg1.Field1") = "abc".
X = obj.Value("Seg1.Seg2.Field1").
X = obj.Value("Seg1.Field1 Seg2.Field2 = '123'")  ' selector.
```

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Qual</td>
<td>Qualifier and selector that uniquely identifies a particular field.</td>
</tr>
</tbody>
</table>

Classification 2

Value

Properties

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Qual</td>
<td>Qualifier and selector that uniquely identifies a particular field.</td>
</tr>
</tbody>
</table>
Properties

Returns the value of control. This property is an abstraction of the different properties used by the original controls to present their content, for example, as Text, Caption, Value, and so on. The implementation of the SAPCtrl must map the original control property in the Value property for each control that is to be supported by SAPforms.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>X</td>
<td>Index (e.g. for grids).</td>
</tr>
<tr>
<td>Return</td>
<td>Value</td>
</tr>
</tbody>
</table>
Version (Property)

Definition
Version of the structure information (workflow) interface (function module)
IDoc class used for compatibility checks.

Remarks
Classification
Version

- SAPData class [Seite 1918]
- SAPforms Data Plug-In for Workflow [Seite 1897]
- SAPData class [Seite 1975]
- SAPforms Data Plug-In for Function Modules [Seite 1954]
- SAPData class [Seite 2027]
- SAPforms Data Plug-In for IDoc [Seite 2006]
- SAPData class [Seite 2078]
- SAPforms Data Plug-In for Demo [Seite 2077]
WorkitemID (Property)

Definition
Returns the work item ID once a workflow has been successfully started.

Remarks

Classification

← SAPData class [Seite 1918]
← SAPforms Data Plug-In for Workflow [Seite 1897]
SAPforms Data Plug-In for Function Modules

Definition
SAPData type plug-in for calling function modules and BAPIs with RFC capability.

Use
You can use this plug-in to design forms that use a function module interface. When the Start() method is executed, the import parameters of the function module are first supplied from the form controls. Following this, the function module in R/3 is called via RFC and the form control is supplied with the results from the export and tables parameters. If you use this plug-in, you do not need an SAPXmit plug-in. The Data plug-in is also responsible for communication with the R/3 System. This plug-in implements the SAPData interface that provides the SAPforms Designer [Seite 1735] and other SAPforms components with access to certain types of R/3 data structures and interfaces. In other words, this plug-in can be regarded as being synonymous with the associated SAPData class.

Classification
SAPforms Data Plug-In for Function Modules

→ Field [Seite 2007]
→ Fields [Seite 2018]
→ SAPData [Seite 1975]

Properties
You can maintain the following properties for this plug-in:

Always bring up R/3 system selection:

If you have selected this indicator, you must chose the R/3 System, in which the function module is to be called, every time you execute the form. If you have not selected this indicator, the RFC is executed in the system you specified when you defined the form.
Field Class

Definition
Class that describes a single element in an R/3 data structure.

Remark
This class describes
a single element in the workflow container
a single field in an IDoc segment
a single function module parameter. Appropriate properties are defined for differentiating between import, export, and tables parameters.

Classification
SAPforms Data Plug-In for Workflow [Seite 1897]
SAPforms Data Plug-In for IDoc [Seite 2006]
SAPforms Data Plug-In for Function Modules [Seite 1954]
**Load (Method)**

**Definition**
Loads structure information to SAPforms objects.

**Remarks**

**Classification 1**

**Load**

- SAPData class [Seite 1918]
- SAPforms Data Plug-In for Workflow [Seite 1897]
- SAPData class [Seite 1975]
- SAPforms Data Plug-In for Function Modules [Seite 1954]
- SAPData class [Seite 2027]
- SAPforms Data Plug-In for IDoc [Seite 2006]
- SAPData class [Seite 2078]
- SAPform Data Plug-In for Demo [Seite 2077]

**Properties**
Loads the property values of an SAPData object from a structured string. As a result, the status of the SAPData object that was previously saved permanently with the Save method is restored. This saves the additional overheads that would result if the structure information had to be loaded each time from the original source. This method is usually called within the Load method of the SAPForm class.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>From</td>
<td>Character string containing the structure information</td>
</tr>
</tbody>
</table>

**Classification 2**

**Load**

- Field Class [Seite 2007]
- SAPforms Data Plug-In for Workflow [Seite 1897]
- SAPforms Data Plug-In for Function Modules [Seite 1954]
- SAPforms Data Plug-In for IDoc [Seite 2006]

**Properties**
Loads the property values of a field object from a structured string. As a result, the status of the field object that was previously saved permanently with the Save method is restored. For this method to be called successfully, the next character at the Pos position must be keyword “Fld”.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>From</td>
<td>Structured string</td>
</tr>
<tr>
<td>Pos</td>
<td>Current position within the string. Should point to the number of remaining properties to read</td>
</tr>
</tbody>
</table>

**Classification 3**

**Load**
Properties

Load (Method)

Loads the property values of a fields object from a structured string. As a result, the status of the fields object that was previously saved permanently with the Save method is restored.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>From</td>
<td>Character string that provides the structure information.</td>
</tr>
<tr>
<td>Pos</td>
<td>Current search position within the character string.</td>
</tr>
</tbody>
</table>

Classification 4

Load

Reconstructs the segment object from a structured string. Counterpart of the Save method.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>From</td>
<td>Character string containing structure information.</td>
</tr>
<tr>
<td>Pos</td>
<td>Current search position within the character string.</td>
</tr>
</tbody>
</table>

Classification 5

Load

This method reconstructs the segment list from a structured string. Counterpart of the Save method.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>From</td>
<td>Character string containing structure information.</td>
</tr>
<tr>
<td>Pos</td>
<td>Current search position within the character string.</td>
</tr>
</tbody>
</table>

Classification 6

Load

This method reconstructs an object of the Link class from the serialized representation. It assumes that the next token is located at the Pos position in the serialized representation of the “Fld” keyword.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>From</td>
<td>Serialization string.</td>
</tr>
</tbody>
</table>
Load (Method)

Current position within the character string.

Classification 7

Load

- SAPForm class [Seite 1772]
  - SAPForms Form Plug-In for Visual Basic [Seite 1756]
- SAPForm class [Seite 1812]
  - SAPForms Form Plug-In for MS Outlook [Seite 1796]
- SAPForm class [Seite 1872]
  - SAPForms Form Plug-In for MS Office [Seite 1856]

Properties

Method for loading form controls into Controls Collection from a form or form template.
Note that the form information is not stored in the Tag property of this object, but reconstructed each time the form is loaded. This method is called at design time and at runtime. It has to populate the Controls collection for both cases. At runtime, the method is called by the user code in the form's Load method. When the form is designed, the method is called by the SAPForms Designer or from outside SAPForms. If you are using the SAPForms Designer as a stand-alone application, the Load method is called with Mode=2. This instructs the Load method to load the form from a file. If you are using the SAPForms Designer as a plug-in, the Load method is invoked from the hosting environment and receives the proprietary form description as a parameter. The populated SAPForm object should be passed to the SAPForms Designer by the PlugIn method of the Designer class.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Frm</td>
<td>form or form template containing controls.</td>
</tr>
<tr>
<td>Mode</td>
<td>0=design time (add-in), 1=design time (file), 2=runtime.</td>
</tr>
</tbody>
</table>

Classification 8

Load

- SAPXmit class [Seite 2106]
  - SAPForms Data Plug-In for Demo [Seite 2077]
- SAPXmit class [Seite 1837]
  - SAPForms Form Plug-In for Outlook [Seite 1796]
- SAPXmit class [Seite 2126]
  - SAPForms Transmit Plug-In for RFC [Seite 2125]
- SAPXmit class [Seite 2168]
  - SAPForms Transmit Plug-In for MAPI [Seite 2146]
- SAPXmit class [Seite 2147]
  - SAPForms Transmit Plug-In for SAPMAPI [Seite 2167]
- SAPXmit class [Seite 2189]
  - SAPForms Transmit Plug-In for WinSock [Seite 2188]
- SAPXmit class [Seite 2209]
  - SAPForms Transmit Plug-In for Demo [Seite 2208]
Properties

Loads the property values of an SAPXmit object from a structured string. As a result, the status of the SAPXmit object that was previously saved permanently with the Save method is restored.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>From</td>
<td>Character string containing the structure information</td>
</tr>
</tbody>
</table>
Name (Property)

Definition
Name of the object, field, segment, or control.

Remarks
This name, for example, must be used as the root in the SAP structure tree that is built by the BuildTree method.

Classification 1
Name

- SAPData class [Seite 1918]
- SAPforms Data Plug-In for Workflow [Seite 1897]

Properties
Returns the name of the workflow to whose task the container description belongs.

Classification 2
Name

- SAPData class [Seite 1975]
- SAPforms Data Plug-In for Function Modules [Seite 1954]

Properties
Name of the R/3 function module that is to be executed.

Classification 3
Name

- SAPData class [Seite 2027]
- SAPforms Data Plug-In for IDoc [Seite 2006]

Properties
Name of the IDoc as it appears in the control record.

Classification 4
Name

- SAPData class [Seite 2078]
- SAPforms Data Plug-In for Demo [Seite 2077]

Classification 5
Name

- Field Class [Seite 2007]
- SAPforms Data Plug-In for Workflow [Seite 1897]
- SAPforms Data Plug-In for Function Modules [Seite 1954]
- SAPforms Data Plug-In for IDoc [Seite 2006]
Properties
Name of the field.

Classification 6
Name
Segment class [Seite 2055]
SAPforms Data Plug-In for IDoc [Seite 2006]

Properties
Name of the segment.
Used as a key in the Templates list of the IDoc object. Only the first attribute with this name is added to the list of segments with a key. All other segment attributes with the same name are added without a key.

Classification 7
Name
SAPForm class [Seite 1772]
SAPforms Form Plug-In for Visual Basic [Seite 1756]
SAPForm Class [Seite 1812]
SAPforms Form Plug-In for MS Outlook [Seite 1796]
SAPForm class [Seite 1872]
SAPforms Form Plug-In for MS Office [Seite 1856]

Properties
Original name of the control in the real form (e.g. Visual Basic form).
Save (Method)

Definition
Saves the updated form definition.

Remarks
This method is used by the SAPforms Designer when a form is designed to save the modified form definition after the binding has been defined. The modifications can affect the binding definition or possibly the source code entered in the form.

Classification 1

Save

 Properties
Saves the current property values of an SAPData object in a structured string. Counterpart of the Load method.

When a form is created, the SAPforms Designer uses this method to save the structure description retrieved from the R/3 System permanently in the form.

The R/3 structure is restored from the structured string at runtime by the Load method.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Tgt</td>
<td>String in which the status is stored.</td>
</tr>
</tbody>
</table>

Classification 2

Save

 Properties
Saves the current property values of an SAPData object in a structured string. Counterpart of the Load method. The string is saved with a form so that the structure information is available when the form is executed.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Tgt</td>
<td>Stores the string structure.</td>
</tr>
</tbody>
</table>
**Classification 3**

**Save**

- Fields Class [Seite 2018]
- SAPforms Data Plug-In for Workflow [Seite 1897]
- SAPforms Data Plug-In for Function Modules [Seite 1954]
- SAPforms Data Plug-In for IDoc [Seite 2006]

**Properties**

Loads the property values of a fields object from a structured string. As a result, the status of the fields object that was previously saved permanently with the Save method is restored. The string is saved with the form so that the structure information is available when the form is executed.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Tgt</td>
<td>Stores the string structure.</td>
</tr>
</tbody>
</table>

**Classification 4**

**Save**

- Segment class [Seite 2055]
- SAPforms Data Plug-In for IDoc [Seite 2006]

**Properties**

This method saves the structure of a segment, including the field list, in a structured string.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Tgt</td>
<td>Stores the string structure.</td>
</tr>
</tbody>
</table>

**Classification 5**

**Save**

- Segments class [Seite 2067]
- SAPforms Data Plug-In for IDoc [Seite 2006]

**Properties**

This method saves the segment list in a structured string. Counterpart of the Load method.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Tgt</td>
<td>Stores the string structure.</td>
</tr>
</tbody>
</table>

**Classification 6**

**Save**

- Links class [Seite 1862]
- SAPforms Form Plug-In for Visual Basic [Seite 1756]
- Links class [Seite 1862]
- SAPforms Form Plug-In for MS Office [Seite 1856]
- Links class [Seite 1862]
- SAPforms Form Plug-In for MS Outlook [Seite 1796]
Properties
Method that writes the content of all links to a file. Counterpart of the Load method.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Tgt</td>
<td>Stores the string structure.</td>
</tr>
</tbody>
</table>

Classification 7
Save

-save sapform class [seite 1772]
save sapforms form plug-in for visual basic [seite 1756]
save sapform class [seite 1872]
save sapforms form plug-in for ms office [seite 1856]
save sapform class [seite 1812]
save sapforms form plug-in for ms outlook [seite 1796]

Properties
Method that saves a form to a file if it was created from a file.
Reads the old file and writes updated lines to the new file.

Classification 8
Save

-save sapxmit class [seite 2106]
save sapforms data plug-in for demo [seite 2077]
save sapxmit class [seite 1837]
save sapforms form plug-in for outlook [seite 1796]
save sapxmit class [seite 2126]
save sapforms transmit plug-in for rfc [seite 2125]
save sapxmit class [seite 2168]
save sapforms transmit plug-in for mapi [seite 2146]
save sapxmit class [seite 2147]
save sapforms transmit plug-in for sapmapi [seite 2167]
save sapxmit class [seite 2189]
save sapforms transmit plug-in for winsock [seite 2188]
save sapxmit class [seite 2209]
save sapforms transmit plug-in for demo [seite 2208]

Properties
Saves the current property values of an SAPXmit object in a structured string. Counterpart of the Load method.
When a form is created, the SAPforms Designer uses this method to save the structure description retrieved from the R/3 System permanently in the form.
The R/3 structure is restored from the structured string at runtime by the Load method.
### Save (Method)

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Tgt</td>
<td>String in which the status is stored.</td>
</tr>
</tbody>
</table>
Fields Class

Definition
List class that describes the field list in an R/3 data structure.

Remark
This class can contain the following lists:
- elements in a workflow container
- segment fields in an IDoc
- function module parameters.

The individual elements in the list can be accessed with the Item property or with a For...Each loop. The R/3 data structures often represent hierarchies. To identify a field in one of these data structures, therefore, you must specify the fully name of this field.

```python
... 
set objFld = SAPData.Fields.Item ("[StruCturX].[FieldY]")
... 
```

Classification
SAPforms Data Plug-In for Workflow [Seite 1897]
SAPforms Data Plug-In for IDoc [Seite 2006]
SAPforms Data Plug-In for Function Modules [Seite 1954]

"Fields Class"
- Load (Method) [Seite 2213]
- Save (Method) [Seite 2219]
Load (Method)

Definition
Loads structure information to SAPforms objects.

Remarks

Classification 1

Load

Properties
Loads the property values of an SAPData object from a structured string. As a result, the status of the SAPData object that was previously saved permanently with the Save method is restored. This saves the additional overheads that would result if the structure information had to be loaded each time from the original source. This method is usually called within the Load method of the SAPForm class.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>From</td>
<td>Character string containing the structure information</td>
</tr>
</tbody>
</table>

Classification 2

Load

Properties
Loads the property values of a field object from a structured string. As a result, the status of the field object that was previously saved permanently with the Save method is restored. For this method to be called successfully, the next character at the Pos position must be keyword “Fld”.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>From</td>
<td>Structured string</td>
</tr>
<tr>
<td>Pos</td>
<td>Current position within the string. Should point to the number of remaining properties to read</td>
</tr>
</tbody>
</table>

Classification 3

Load
Load (Method)

Properties
Loads the property values of a fields object from a structured string. As a result, the status of the fields object that was previously saved permanently with the Save method is restored.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>From</td>
<td>Character string that provides the structure information.</td>
</tr>
<tr>
<td>Pos</td>
<td>Current search position within the character string.</td>
</tr>
</tbody>
</table>

Classification 4
Load

Properties
Reconstructs the segment object from a structured string. Counterpart of the Save method.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>From</td>
<td>Character string containing structure information.</td>
</tr>
<tr>
<td>Pos</td>
<td>Current search position within the character string.</td>
</tr>
</tbody>
</table>

Classification 5
Load

Properties
This method reconstructs the segment list from a structured string. Counterpart of the Save method.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>From</td>
<td>Character string containing structure information.</td>
</tr>
<tr>
<td>Pos</td>
<td>Current search position within the character string.</td>
</tr>
</tbody>
</table>

Classification 6
Load

Properties
This method reconstructs an object of the Link class from the serialized representation. It assumes that the next token is located at the Pos position in the serialized representation of the “Fld” keyword.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>From</td>
<td>Serialization string.</td>
</tr>
</tbody>
</table>
Load (Method)

Current position within the character string.

Classification 7

Load

SAPForm class [Seite 1772]
SAPforms Form Plug-In for Visual Basic [Seite 1756]
SAPForm class [Seite 1812]
SAPforms Form Plug-In for MS Outlook [Seite 1796]
SAPForm class [Seite 1872]
SAPforms Form Plug-In for MS Office [Seite 1856]

Properties

Method for loading form controls into Controls Collection from a form or form template. Note that the form information is not stored in the Tag property of this object, but reconstructed each time the form is loaded. This method is called at design time and at runtime. It has to populate the Controls collection for both cases. At runtime, the method is called by the user code in the form's Load method. When the form is designed, the method is called by the SAPForms Designer or from outside SAPForms. If you are using the SAPForms Designer as a stand-alone application, the Load method is called with Mode=2. This instructs the Load method to load the form from a file. If you are using the SAPForms Designer as a plug-in, the Load method is invoked from the hosting environment and receives the proprietary form description as a parameter. The populated SAPForm object should be passed to the SAPForms Designer by the PlugIn method of the Designer class.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Frm</td>
<td>form or form template containing controls.</td>
</tr>
<tr>
<td>Mode</td>
<td>0=design time (add-in), 1=design time (file), 2=runtime.</td>
</tr>
</tbody>
</table>

Classification 8

Load

SAPXmit class [Seite 2106]
SAPforms Data Plug-In for Demo [Seite 2077]
SAPXmit class [Seite 1837]
SAPforms Form Plug-In for Outlook [Seite 1796]
SAPXmit class [Seite 2126]
SAPforms Transmit Plug-In for RFC [Seite 2125]
SAPXmit class [Seite 2168]
SAPforms Transmit Plug-In for MAPI [Seite 2146]
SAPXmit class [Seite 2147]
SAPforms Transmit Plug-In for SAPMAPI [Seite 2167]
SAPXmit class [Seite 2189]
SAPforms Transmit Plug-In for WinSock [Seite 2188]
SAPXmit class [Seite 2209]
SAPforms Transmit Plug-In for Demo [Seite 2208]
**Load (Method)**

**Properties**
Loads the property values of an SAPXmit object from a structured string. As a result, the status of the SAPXmit object that was previously saved permanently with the Save method is restored.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>From</td>
<td>Character string containing the structure information</td>
</tr>
</tbody>
</table>
Save (Method)

Definition
Saves the updated form definition.

Remarks
This method is used by the SAPforms Designer when a form is designed to save the modified form definition after the binding has been defined. The modifications can affect the binding definition or possibly the source code entered in the form.

Classification 1
Save

Properties
Saves the current property values of an SAPData object in a structured string. Counterpart of the Load method.
When a form is created, the SAPforms Designer uses this method to save the structure description retrieved from the R/3 System permanently in the form.
The R/3 structure is restored from the structured string at runtime by the Load method.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Tgt</td>
<td>String in which the status is stored.</td>
</tr>
</tbody>
</table>

Classification 2
Save

Properties
Saves the current property values of an SAPData object in a structured string. Counterpart of the Load method. The string is saved with a form so that the structure information is available when the form is executed.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Tgt</td>
<td>Stores the string structure.</td>
</tr>
</tbody>
</table>
Classification 3

Save

辽宁省 Class [Page 2018]
- SAPforms Data Plug-In for Workflow [Page 1897]
- SAPforms Data Plug-In for Function Modules [Page 1954]
- SAPforms Data Plug-In for IDoc [Page 2006]

Properties

Load the property values of a fields object from a structured string. As a result, the status of the fields object that was previously saved permanently with the Save method is restored.

The string is saved with the form so that the structure information is available when the form is executed.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Tgt</td>
<td>Stores the string structure.</td>
</tr>
</tbody>
</table>

Classification 4

Save

辽宁省 class [Page 2055]
- SAPforms Data Plug-In for IDoc [Page 2006]

Properties

This method saves the structure of a segment, including the field list, in a structured string.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Tgt</td>
<td>Stores the string structure.</td>
</tr>
</tbody>
</table>

Classification 5

Save

辽宁省 class [Page 2067]
- SAPforms Data Plug-In for IDoc [Page 2006]

Properties

This method saves the segment list in a structured string. Counterpart of the Load method.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Tgt</td>
<td>Stores the string structure.</td>
</tr>
</tbody>
</table>

Classification 6

Save

辽宁省 class [Page 1862]
- SAPforms Form Plug-In for Visual Basic [Page 1756]
- SAPforms Form Plug-In for MS Office [Page 1856]
- SAPforms Form Plug-In for MS Outlook [Page 1796]
Properties
Method that writes the content of all links to a file. Counterpart of the Load method.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Tgt</td>
<td>Stores the string structure.</td>
</tr>
</tbody>
</table>

Classification 7

Save

- SAPForm class [Seite 1772]
- SAPforms Form Plug-In for Visual Basic [Seite 1756]
- SAPForm class [Seite 1872]
- SAPforms Form Plug-In for MS Office [Seite 1856]
- SAPForm class [Seite 1812]
- SAPforms Form Plug-In for MS Outlook [Seite 1796]

Properties
Method that saves a form to a file if it was created from a file.
Reads the old file and writes updated lines to the new file.

Classification 8

Save

- SAPXmit class [Seite 2106]
- SAPforms Data Plug-In for Demo [Seite 2077]
- SAPXmit class [Seite 1837]
- SAPforms Form Plug-In for Outlook [Seite 1796]
- SAPXmit class [Seite 2126]
- SAPforms Transmit Plug-In for RFC [Seite 2125]
- SAPXmit class [Seite 2168]
- SAPforms Transmit Plug-In for MAPI [Seite 2146]
- SAPXmit class [Seite 2147]
- SAPforms Transmit Plug-In for SAPMAPI [Seite 2167]
- SAPXmit class [Seite 2189]
- SAPforms Transmit Plug-In for WinSock [Seite 2188]
- SAPXmit class [Seite 2209]
- SAPforms Transmit Plug-In for Demo [Seite 2208]

Properties
Saves the current property values of an SAPXmit object in a structured string. Counterpart of the Load method.
When a form is created, the SAPforms Designer uses this method to save the structure description retrieved from the R/3 System permanently in the form.
The R/3 structure is restored from the structured string at runtime by the Load method.
<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Tgt</td>
<td>String in which the status is stored.</td>
</tr>
</tbody>
</table>
SAPData Class

Definition
Original class of this plug-in; representation of an R/3 function module.

Remarks
The class represents function modules with their import, export, and tables parameters. The Fields property refers to an object in the Fields list class that represents the parameters. The parameters themselves can be accessed with the Value property.

SAPData plug-in type classes implement the SAPData interface that provides the SAPforms Designer and other SAPforms components with access to certain types of R/3 data structures and interfaces. In other words, these classes can be regarded as being synonymous with various plug-ins of this type.

A SAPData object is usually also instantiated with the SAPForm object. The SAPData object is then referenced by the SAPData property of the SAPForm object. An object of the SAPData class, however, cannot exist without an object of the SAPForms class.

If you want to develop your own SAPData-type plug-ins, you should use the source code supplied for the SAPforms Data Plug-Ins for Demo as a reference (template). The template is stored in the directory ...\sapforms\sapddemo.

Classification
SAPforms Data Plug-In for Function Modules [Seite 1954]

→ SAPData Class
  → Connection (Property) [Seite 2028]
  → Execute (Method) [Seite 2210]
  → Fields (Property) [Seite 2056]
  → Load (Method) [Seite 2213]
  → LoadInstance (Method) [Seite 2086]
  → Name (Property) [Seite 2088]
  → Options (Method) [Seite 2217]
  → R3DoNotUse (Property) [Seite 2137]
  → RfcHandle (Property) [Seite 2138]
  → SAPXmit (Property) [Seite 2093]
  → Save (Method) [Seite 2219]
  → SaveInstance (Method) [Seite 2098]
  → Start (Method) [Seite 2223]
  → Value (Property) [Seite 2103]
  → Version (Property) [Seite 2105]
  → WorkitemID (Property) [Seite 1953]
Connection (Property)

Definition
Sets or returns the object of the RFC connection.

Remarks

Classification

Connection

← SAPData class [Seite 1918]
← SAPforms Data Plug-In for Workflow [Seite 1897]
← SAPData class [Seite 1975]
← SAPforms Data Plug-In for Function Modules [Seite 1954]
← SAPData class [Seite 2027]
← SAPforms Data Plug-In for IDoc [Seite 2006]
Execute (Method)

Definition
Method that calls the Execute function in the R/3 System. The semantics of this function depend on the SAPData object.

Remarks
The method can be called directly via the SAPData object or indirectly via the Start method of the SAPForm object, whose SAPData property is a reference to the SAPData object.

In the case of the SAPData object for workflow containers, for example, this method executes a work item in an SAP Business Workflow and populates its container elements with values from the form fields.

Classification 1

Execute

- SAPData class [Seite 1918]
- SAPforms Data Plug-In for Workflow [Seite 1897]

Properties
Terminates a work item in the R/3 System and supplies its container elements with the data values in the fields list.

Classification 2

Execute

- SAPData class [Seite 2078]
- SAPXmit class [Seite 2106]
- SAPforms Data Plug-In for Demo [Seite 2077]

Properties
Does not execute an action in the R/3 System but rather displays the data of the object in a separate window.

Classification 3

Execute

- SAPForm class [Seite 1772]
- SAPforms Form Plug-In for Visual Basic [Seite 1756]
- SAPForm class [Seite 1872]
- SAPforms Form Plug-In for MS Office [Seite 1856]
- SAPForm class [Seite 1812]
- SAPforms Form Plug-In for MS Outlook [Seite 1796]
Properties
Invokes the Execute method in the associated Xmit or SAPData object. The semantics of this method depend on this object.

Classification 4
Execute

Properties
This method executes a work item synchronously via a remote function call (RFC). Supports SAPData object of type Container.
This method causes a certain function module to be executed in the R/3 System. This module receives a table with a complete R3F structure containing the work item information and the container data that are to be supplied from the form.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Data</td>
<td>Object representing the container data.</td>
</tr>
</tbody>
</table>

Classification 5
Execute

Properties
Sends a mail to the R/3 System that contains the information required to terminate a work item (including container data) as an attachment.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Data</td>
<td>Object representing the container data.</td>
</tr>
</tbody>
</table>

Classification 6
Execute

Properties
Overwrites the R3F attachment, thereby updating it.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Pdata</td>
<td>Object that represents the mobjData container data</td>
</tr>
</tbody>
</table>

Classification 7
Execute

Properties
Properties
Displays the instance data of the associated SAPData object in a window instead of sending it to the R/3 System.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Data</td>
<td>object of class SAPData</td>
</tr>
</tbody>
</table>
Fields (Property)

**Definition**
List of the elements in an R/3 Structure, i.e.

- the container elements in the workflow or work item that is represented by the SAPData object
- the parameters contained in the function module
- the fields contained in the IDoc segment.

**Remarks**

**Classification**

**Fields**

- SAPData class [Seite 1918]
- SAPforms Data Plug-In for Workflow [Seite 1897]
- SAPData class [Seite 1975]
- SAPforms Data Plug-In for Function Modules [Seite 1954]
- SAPData class [Seite 2027]
- SAPforms Data Plug-In for IDoc [Seite 2006]
Load (Method)

Definition
Loads structure information to SAPforms objects.

Remarks
Classification 1

Load

Properties
Loads the property values of an SAPData object from a structured string. As a result, the status of the SAPData object that was previously saved permanently with the Save method is restored. This saves the additional overheads that would result if the structure information had to be loaded each time from the original source. This method is usually called within the Load method of the SAPForm class.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>From</td>
<td>Character string containing the structure information</td>
</tr>
</tbody>
</table>

Classification 2

Load

Properties
Loads the property values of a field object from a structured string. As a result, the status of the field object that was previously saved permanently with the Save method is restored. For this method to be called successfully, the next character at the Pos position must be keyword “Fld”.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>From</td>
<td>Structured string</td>
</tr>
<tr>
<td>Pos</td>
<td>Current position within the string. Should point to the number of remaining properties to read</td>
</tr>
</tbody>
</table>

Classification 3

Load
Load (Method)

Properties
Loads the property values of a fields object from a structured string. As a result, the status of the fields object that was previously saved permanently with the Save method is restored.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>From</td>
<td>Character string that provides the structure information.</td>
</tr>
<tr>
<td>Pos</td>
<td>Current search position within the character string.</td>
</tr>
</tbody>
</table>

Classification 4
Load

Properties
Reconstructs the segment object from a structured string. Counterpart of the Save method.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>From</td>
<td>Character string containing structure information.</td>
</tr>
<tr>
<td>Pos</td>
<td>Current search position within the character string.</td>
</tr>
</tbody>
</table>

Classification 5
Load

Properties
This method reconstructs the segment list from a structured string. Counterpart of the Save method.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>From</td>
<td>Character string containing structure information.</td>
</tr>
<tr>
<td>Pos</td>
<td>Current search position within the character string.</td>
</tr>
</tbody>
</table>

Classification 6
Load

Properties
This method reconstructs an object of the Link class from the serialized representation.
It assumes that the next token is located at the Pos position in the serialized representation of the “Fld” keyword.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>From</td>
<td>Serialization string.</td>
</tr>
</tbody>
</table>
Load (Method)

Current position within the character string:

<table>
<thead>
<tr>
<th>Pos</th>
</tr>
</thead>
</table>

Classification 7

Load

- SAPForm class [Seite 1772]
- SAPforms Form Plug-In for Visual Basic [Seite 1756]
- SAPForm class [Seite 1812]
- SAPforms Form Plug-In for MS Outlook [Seite 1796]
- SAPForm class [Seite 1872]
- SAPforms Form Plug-In for MS Office [Seite 1856]

Properties

Method for loading form controls into Controls Collection from a form or form template.

Note that the form information is not stored in the Tag property of this object, but reconstructed each time the form is loaded. This method is called at design time and at runtime. It has to populate the Controls collection for both cases. At runtime, the method is called by the user code in the form's Load method.

When the form is designed, the method is called by the SAPForms Designer or from outside SAPForms. If you are using the SAPForms Designer as a stand-alone application, the Load method is called with Mode=2. This instructs the Load method to load the form from a file. If you are using the SAPForms Designer as a plug-in, the Load method is invoked from the hosting environment and receives the proprietary form description as a parameter. The populated SAPForm object should be passed to the SAPForms Designer by the Plugin method of the Designer class.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Frm</td>
<td>form or form template containing controls.</td>
</tr>
<tr>
<td>Mode</td>
<td>0=design time (add-in), 1=design time (file), 2=runtime.</td>
</tr>
</tbody>
</table>

Classification 8

Load

- SAPXmit class [Seite 2106]
- SAPforms Data Plug-In for Demo [Seite 2077]
- SAPXmit class [Seite 1837]
- SAPforms Form Plug-In for Outlook [Seite 1796]
- SAPXmit class [Seite 2126]
- SAPforms Transmit Plug-In for RFC [Seite 2125]
- SAPXmit class [Seite 2168]
- SAPforms Transmit Plug-In for MAPI [Seite 2146]
- SAPXmit class [Seite 2147]
- SAPforms Transmit Plug-In for SAPMAPI [Seite 2167]
- SAPXmit class [Seite 2189]
- SAPforms Transmit Plug-In for WinSock [Seite 2188]
- SAPXmit class [Seite 2209]
- SAPforms Transmit Plug-In for Demo [Seite 2208]
Load (Method)

Properties
Loads the property values of an SAPXmit object from a structured string. As a result, the status of the SAPXmit object that was previously saved permanently with the Save method is restored.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>From</td>
<td>Character string containing the structure information</td>
</tr>
</tbody>
</table>
LoadInstance (Method)

Definition
Method that loads the data content of a SAPData object from a file.

Remarks
The method is usually called when data flows from the R/3 System to the form and is to be displayed by the form. This is the case, for example, when a work item is executed. The form is then started by the SAPforms Helper [Seite 1741] and the R3F file is forwarded as a parameter. The existence of a parameter when the form is started ensures that the LoadInstance method is called in the Init() method of the SAPform object. The GetData method is usually called after the LoadInstance method.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Path</td>
<td>Complete path designation</td>
</tr>
</tbody>
</table>

Classification 1
LoadInstance

Properties
Loads the R/3 workflow information from an R3F file to the SAPData object. The current container element values can be interrogated and manipulated via the SAPData.Value and SAPData.Fields(s).Data properties in particular.

Classification 2
LoadInstance

Properties
Implementation of the SAPData interface for R/3 function modules (BAPIs).

Classification 3
LoadInstance

Properties
Loads an IDoc instance from a file.

Builds the segments list from the records in the file. This method will only function correctly if the Templates list containing the entire IDoc meta structure exists. The structure of each segment in the Segments list is derived from the corresponding segment in the Templates list.

Classification 4
LoadInstance
LoadInstance (Method)

Properties
The method saves the contents of a field in the FieldValues array. This method requires that the elements be saved in a fixed order. Even though the field name is contained in the file, it is not evaluated.
Name (Property)

Definition
Name of the object, field, segment, or control.

Remarks
This name, for example, must be used as the root in the SAP structure tree that is built by the `BuildTree` method.

Classification 1
Name
− SAPData class [Seite 1918]
− SAPforms Data Plug-In for Workflow [Seite 1897]

Properties
Returns the name of the workflow to whose task the container description belongs.

Classification 2
Name
− SAPData class [Seite 1975]
− SAPforms Data Plug-In for Function Modules [Seite 1954]

Properties
Name of the R/3 function module that is to be executed.

Classification 3
Name
− SAPData class [Seite 2027]
− SAPforms Data Plug-In for IDoc [Seite 2006]

Properties
Name of the IDoc as it appears in the control record.

Classification 4
Name
− SAPData class [Seite 2078]
− SAPforms Data Plug-In for Demo [Seite 2077]

Classification 5
Name
− Field Class [Seite 2007]
− SAPforms Data Plug-In for Workflow [Seite 1897]
− SAPforms Data Plug-In for Function Modules [Seite 1954]
− SAPforms Data Plug-In for IDoc [Seite 2006]
Properties
Name of the field.

Classification 6
Name

← Segment class [Seite 2055]
← SAPforms Data Plug-In for IDoc [Seite 2006]

Properties
Name of the segment.
Used as a key in the Templates list of the IDoc object. Only the first attribute with this name is added to the list of segments with a key. All other segment attributes with the same name are added without a key.

Classification 7
Name

← SAPForm class [Seite 1772]
← SAPforms Form Plug-In for Visual Basic [Seite 1756]
← SAPForm Class [Seite 1812]
← SAPforms Form Plug-In for MS Outlook [Seite 1796]
← SAPForm class [Seite 1872]
← SAPforms Form Plug-In for MS Office [Seite 1856]

Properties
Original name of the control in the real form (e.g. Visual Basic form).
Options (Method)

Definition
Method that outputs the object-specific options dialog box.

Remarks
This dialog box displays either global settings or settings that are valid for the current form. These settings can be changed here. The global settings are read from the registry before they are displayed and are then saved again to the registry.
The method is called within the SAPforms Designer if the required plug-in is active under Properties on the Plug-Ins tab in the Options dialog. You can also use the sample program Options.exe to display the Options dialog of a certain plug-in.

Classification 1
Options

- SAPData class [Seite 1918]
- SAPforms Data Plug-In for Workflow [Seite 1897]
- SAPData class [Seite 1975]
- SAPforms Data Plug-In for Function Modules [Seite 1954]
- SAPData class [Seite 2027]
- SAPforms Data Plug-In for IDoc [Seite 2006]
- SAPData class [Seite 2078]
- SAPforms Data Plug-In for Demo [Seite 2077]

Classification 2
Options

- SAPForm class [Seite 1772]
- SAPforms Form Plug-In for Visual Basic [Seite 1756]
- SAPForm class [Seite 1812]
- SAPforms Form Plug-In for MS Outlook [Seite 1796]
- SAPForm class [Seite 1872]
- SAPforms Form Plug-In for MS Office [Seite 1856]

Classification 3
Options

- SAPXmit class [Seite 2106]
- SAPforms Data Plug-In for Demo [Seite 2077]
- SAPXmit class [Seite 1837]
- SAPforms Form Plug-In for MS Outlook [Seite 1796]
- SAPXmit class [Seite 2126]
Options (Method)

- SAPforms Transmit Plug-In for RFC [Seite 2125]
- SAPXmit class [Seite 2168]
- SAPforms Transmit Plug-In for MAPI [Seite 2146]
- SAPXmit class [Seite 2147]
- SAPforms Transmit Plug-In for SAPMAPI [Seite 2167]
- SAPXmit class [Seite 2189]
- SAPforms Transmit Plug-In for SMTP [Seite 2188]
- SAPXmit class [Seite 2209]
- SAPforms Transmit Plug-In for Demo [Seite 2208]
**R3DoNotUse (Property)**

**Definition**
Determines whether the R/3 system information, which was stored permanently in the form when it was designed, is to be used when the Start() and Execute() methods are executed.

**Remarks**

**Classification**

**R3DoNotUse**

← SAPData class [Seite 1918]
← SAPforms Data Plug-In for Workflow [Seite 1897]
← SAPData class [Seite 1975]
← SAPforms Data Plug-In for Function Modules [Seite 1954]
← SAPData class [Seite 2027]
← SAPforms Data Plug-In for IDoc [Seite 2006]
← SAPXmit class [Seite 2126]
← SAPforms Transmit Plug-In for RFC [Seite 2125]

**Properties**
The default value of this field is FALSE. When the methods are executed, logon takes place to the R/3 System that was used to design the form. This R/3 System selection is skipped during logon. If the same form is to be used with different R/3 Systems, you must set this property to TRUE. You can specify this either in the form's program code or by using the options dialog of the SAPData object.
RfcHandle (Property)

Definition
Represents the reference to a connection to an R/3 System.

Remarks

Classification

RfcHandle

← SAPData class [Seite 1975]
← SAPforms Data Plug-In for Function Modules [Seite 1954]
← SAPXmit class [Seite 2126]
← SAPforms Transmit Plug-In for RFC [Seite 2125]

Properties
This reference is used if it exists. Otherwise, an R/3 Logon is carried out.
SAPXmit (Property)

Definition
Reference to the SAPXmit object that is used to encapsulate the transport of SAPData to the R/3 System.

Remarks

Classification

SAPXmit

- SAPData class [Seite 1918]
- SAPforms Data Plug-In for Workflow [Seite 1897]
- SAPData class [Seite 1975]
- SAPforms Data Plug-In for Function Modules [Seite 1954]
- SAPData class [Seite 2027]
- SAPforms Data Plug-In for IDoc [Seite 2006]
- SAPData class [Seite 2078]
- SAPforms Data Plug-In for Demo [Seite 2077]
- SAPForm class [Seite 1772]
- SAPforms Form Plug-In for Visual Basic [Seite 1756]
- SAPForm class [Seite 1812]
- SAPforms Form Plug-In for MS Outlook [Seite 1796]
- SAPForm class [Seite 1872]
- SAPforms Form Plug-In for MS Office [Seite 1856]
Save (Method)

Definition
Saves the updated form definition.

Remarks
This method is used by the SAPforms Designer when a form is designed to save the modified form definition after the binding has been defined. The modifications can affect the binding definition or possibly the source code entered in the form.

Classification 1
Save

Properties
Saves the current property values of an SAPData object in a structured string. Counterpart of the Load method.

When a form is created, the SAPforms Designer uses this method to save the structure description retrieved from the R/3 System permanently in the form.

The R/3 structure is restored from the structured string at runtime by the Load method.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Tgt</td>
<td>String in which the status is stored.</td>
</tr>
</tbody>
</table>

Classification 2
Save

Properties
Saves the current property values of an SAPData object in a structured string. Counterpart of the Load method. The string is saved with a form so that the structure information is available when the form is executed.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Tgt</td>
<td>Stores the string structure.</td>
</tr>
</tbody>
</table>
Classification 3
Save

- Fields Class [Seite 2018]
- SAPforms Data Plug-In for Workflow [Seite 1897]
- SAPforms Data Plug-In for Function Modules [Seite 1954]
- SAPforms Data Plug-In for IDoc [Seite 2006]

Properties
Loads the property values of a fields object from a structured string. As a result, the status of the fields object that was previously saved permanently with the Save method is restored. The string is saved with the form so that the structure information is available when the form is executed.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Tgt</td>
<td>Stores the string structure.</td>
</tr>
</tbody>
</table>

Classification 4
Save

- Segment class [Seite 2055]
- SAPforms Data Plug-In for IDoc [Seite 2006]

Properties
This method saves the structure of a segment, including the field list, in a structured string.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Tgt</td>
<td>Stores the string structure.</td>
</tr>
</tbody>
</table>

Classification 5
Save

- Segments class [Seite 2067]
- SAPforms Data Plug-In for IDoc [Seite 2006]

Properties
This method saves the segment list in a structured string. Counterpart of the Load method.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Tgt</td>
<td>Stores the string structure.</td>
</tr>
</tbody>
</table>

Classification 6
Save

- Links class [Seite 1862]
- SAPforms Form Plug-In for Visual Basic [Seite 1756]
- Links class [Seite 1862]
- SAPforms Form Plug-In for MS Office [Seite 1856]
- Links class [Seite 1862]
- SAPforms Form Plug-In for MS Outlook [Seite 1796]
Properties
Method that writes the content of all links to a file. Counterpart of the Load method.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Tgt</td>
<td>Stores the string structure.</td>
</tr>
</tbody>
</table>

Classification 7

Save

- SAPForm class [Seite 1772]
- SAPforms Form Plug-In for Visual Basic [Seite 1756]
- SAPForm class [Seite 1872]
- SAPforms Form Plug-In for MS Office [Seite 1856]
- SAPForm class [Seite 1812]
- SAPforms Form Plug-In for MS Outlook [Seite 1796]

Properties
Method that saves a form to a file if it was created from a file.
Reads the old file and writes updated lines to the new file.

Classification 8

Save

- SAPXmit class [Seite 2106]
- SAPforms Data Plug-In for Demo [Seite 2077]
- SAPXmit class [Seite 1837]
- SAPforms Form Plug-In for Outlook [Seite 1796]
- SAPXmit class [Seite 2126]
- SAPforms Transmit Plug-In for RFC [Seite 2125]
- SAPXmit class [Seite 2168]
- SAPforms Transmit Plug-In for MAPI [Seite 2146]
- SAPXmit class [Seite 2147]
- SAPforms Transmit Plug-In for SAPMAPI [Seite 2167]
- SAPXmit class [Seite 2189]
- SAPforms Transmit Plug-In for WinSock [Seite 2188]
- SAPXmit class [Seite 2209]
- SAPforms Transmit Plug-In for Demo [Seite 2208]

Properties
Saves the current property values of an SAPXmit object in a structured string. Counterpart of the Load method.
When a form is created, the SAPforms Designer uses this method to save the structure description retrieved from the R/3 System permanently in the form.
The R/3 structure is restored from the structured string at runtime by the Load method.
<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Tgt</td>
<td>String in which the status is stored.</td>
</tr>
</tbody>
</table>
SaveInstance (Method)

Definition
This method saves the current data content of the SAPData object to the specified file.

Remarks
This data content of the SAPData object is the result of the PutData method of the SAPForm object or the result of assignments to the SAPData method in the SAPData object. The method is usually called from the SAPXmit object to save the data content in a file before it is sent by email, for example.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Path</td>
<td>Complete path specifying where the data content is to be saved.</td>
</tr>
</tbody>
</table>

Classification 1
SaveInstance

Properties
Method that saves the current contents of the function module to the specified file in R3F format.

Classification 2
SaveInstance

Properties
Saves the current IDoc instance segment-by-segment to the specified file. Uses the Record property of the segment object to generate a record for a segment.

Classification 3
SaveInstance

Properties
Generates a file containing a separate line for each data field.
Start (Method)

Definition
Method that executes the *Start* function in R/3. The semantics of this function depend on the concrete SAPData object.

Remarks
In the case of the SAPData object for workflow containers, this method starts a workflow and populates its container elements with values from the form fields. This method can be called directly via the SAPData object or indirectly via the *Start* method of the SAPForm object, whose SAPData property is a reference to the SAPData object. If an SAPXmit object is assigned to the SAPData object, the *Start* method of the SAPXmit object is called.

Classification 1

Start

- **Properties**
  Starts a workflow in the R/3 System and supplies its container elements with the data values in the fields list.

Classification 2

Start

- **Properties**
  This method carries out the current R/3 function module call.

Classification 3

Start

- **Properties**
  Generates an IDoc file from the data in the segments list and transfers this IDoc to the R/3 System. The transfer method depends on the assigned SAPXmit object. A workflow can be started in the R/3 System when the inbound IDoc is processed.

Classification 4

Start

- **Properties**
  - SAPData class [Seite 2077]
Properties
Does not execute an action in the R/3 System but rather displays the data of the object in a separate window.

Classification 5

Start

- SAPForm class [Seite 1772]
- SAPforms Form Plug-In for Visual Basic [Seite 1756]
- SAPForm class [Seite 1872]
- SAPforms Form Plug-In for MS Office [Seite 1856]
- SAPForm class [Seite 1812]
- SAPforms Form Plug-In for MS Outlook [Seite 1796]

Properties
Invokes the Start method in the associated SAPXmit or SAPData object. The semantics of this method depend on this SAPData object. If this object is the SAPData object for SAP Business Workflow, this method starts a workflow in an R/3 System.

Classification 6

Start

- SAPXmit class [Seite 2126]
- SAPforms Transmit Plug-In for RFC [Seite 2125]

Properties
This method starts a workflow synchronously via RFC. Supports SAPData objects of types "Idoc" and "Container".

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Data</td>
<td>Object of type SAPData.</td>
</tr>
</tbody>
</table>

Classification 7

Start

- SAPXmit class [Seite 2168]
- SAPforms Transmit Plug-In for MAPI [Seite 2146]
- SAPXmit class [Seite 2147]
- SAPforms Transmit Plug-In for SAPMAPI [Seite 2167]
- SAPXmit class [Seite 2189]
- SAPforms Transmit Plug-In for SMTP [Seite 2188]
- SAPXmit class [Seite 1837]
- SAPforms Form Plug-In for MS Outlook [Seite 1796]

Properties
Sends a mail to the R/3 System with an attachment containing the information required to start a workflow.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Data</td>
<td>Object of type SAPData.</td>
</tr>
</tbody>
</table>
Classification 8

Start

← SAPXmit class [Seite 2209]
← SAPforms Transmit Plug-In for Demo [Seite 2208]
← SAPXmit class [Seite 2106]
← SAPforms Data Plug-In for Demo [Seite 2077]

Properties
Displays the instance data of the associated SAPData object in a window instead of sending it to the R/3 System.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Data</td>
<td>Object of type SAPData.</td>
</tr>
</tbody>
</table>
**Value (Property)**

**Definition**
Value of a specific field type.

**Remarks**

**Classification 1**
**Value**
- SAPData class [Seite 1918]
- SAPforms Data Plug-In for Workflow [Seite 1997]
- SAPData class [Seite 1975]
- SAPforms Data Plug-In for Function Modules [Seite 1954]
- SAPData class [Seite 2027]
- SAPforms Data Plug-In for IDoc [Seite 2006]
- SAPData class [Seite 2078]
- SAPforms Data Plug-In for Demo [Seite 2077]

**Properties**
The value of an SAP data field, defined in greater detail by the *Qual* parameter.
The structure and syntax of the *Qual* parameter depends on the concrete SAP data structure from which this field originates. In the simplest case, the *Qual* parameter designates the name of a field whose value is to be set or read. If the field is part of a structure (for example, an IDoc segment), *Qual* must identify the field uniquely within the overall structure. If the structure is repeated within an instance - as is the case with some IDocs - *Qual* must also contain a selector that identifies the IDoc segment uniquely.

```
   obj.Value("Field1") = "abc".
   Obj.Value("Seg1.Field1") = "abc".
   X = obj.Value("Seg1.Seg2.Field1").
   X = obj.Value("Seg1.Field1 Seg2.Field2 = '123'") ' selector.
```

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Qual</td>
<td>Qualifier and selector that uniquely identifies a particular field.</td>
</tr>
</tbody>
</table>

**Classification 2**
**Value**
- SAPCtrl class [Seite 1767]
- SAPforms Form Plug-In for Visual Basic [Seite 1756]
- SAPCtrl class [Seite 1807]
- SAPforms Form Plug-In for MS Outlook [Seite 1796]
- SAPCtrl class [Seite 1867]
- SAPforms Form Plug-In for MS Office [Seite 1856]
Properties

Returns the value of control.

This property is an abstraction of the different properties used by the original controls to present their content, for example, as Text, Caption, Value, and so on. The implementation of the SAPCtrl must map the original control property in the Value property for each control that is to be supported by SAPforms.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>X</td>
<td>Index (e.g. for grids).</td>
</tr>
<tr>
<td>Return</td>
<td>Value</td>
</tr>
</tbody>
</table>
Version (Property)

Definition
Version of the structure information (workflow) interface (function module)
IDoc class used for compatibility checks.

Remarks

Classification

Version

← SAPData class [Seite 1918]
← SAPforms Data Plug-In for Workflow [Seite 1897]
← SAPData class [Seite 1975]
← SAPforms Data Plug-In for Function Modules [Seite 1954]
← SAPData class [Seite 2027]
← SAPforms Data Plug-In for IDoc [Seite 2006]
← SAPData class [Seite 2078]
← SAPforms Data Plug-In for Demo [Seite 2077]
SAPforms Data Plug-In for IDoc

Definition
SAPData type plug-in for using an IDoc.

Use
This plug-in can be used to populate an IDoc using an electronic form. This plug-in can be used with the SAPXmit plug-in for RFC.
This plug-in implements the SAPData interface that provides the SAPforms Designer [Seite 1735] and other SAPforms components with access to certain types of R/3 data structures and interfaces. In other words, this plug-in can be regarded as being synonymous with the associated SAPData class.

Classification

Properties
Import Structure:
If you choose the From File option, the IDoc structure will be exported from a local file.
If you choose the From R/3 option, the IDoc structure will be retrieved from the R/3 System directly.
If you choose the Ask option, the system will ask you when you open the IDoc structure whether it is to be retrieved from a file or from R/3 directly.
Field Class

Definition
Class that describes a single element in an R/3 data structure.

Remark
This class describes:
- a single element in the workflow container
- a single field in an IDoc segment
- a single function module parameter. Appropriate properties are defined for differentiating between import, export, and tables parameters.

Classification
SAPforms Data Plug-In for Workflow [Seite 1897]
SAPforms Data Plug-In for IDoc [Seite 2006]
SAPforms Data Plug-In for Function Modules [Seite 1954]

➡ Field Class
➡ Load (Method) [Seite 2213]
➡ Name (Property) [Seite 2088]
➡ Save (Method) [Seite 2219]
Load (Method)

Definition
Loads structure information to SAPforms objects.

Remarks

Classification 1
Load

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>From</td>
<td>Character string containing the structure information</td>
</tr>
</tbody>
</table>

Properties
Loads the property values of an SAPData object from a structured string. As a result, the status of the SAPData object that was previously saved permanently with the Save method is restored. This saves the additional overheads that would result if the structure information had to be loaded each time from the original source. This method is usually called within the Load method of the SAPForm class.

Classification 2
Load

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>From</td>
<td>Structured string</td>
</tr>
<tr>
<td>Pos</td>
<td>Current position within the string. Should point to the number of remaining properties to read</td>
</tr>
</tbody>
</table>

Properties
Loads the property values of a field object from a structured string. As a result, the status of the field object that was previously saved permanently with the Save method is restored. For this method to be called successfully, the next character at the Pos position must be keyword “Fld”.

Classification 3
Load
Load (Method)

Properties
Loads the property values of a fields object from a structured string. As a result, the status of the fields object that was previously saved permanently with the Save method is restored.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>From</td>
<td>Character string that provides the structure information.</td>
</tr>
<tr>
<td>Pos</td>
<td>Current search position within the character string.</td>
</tr>
</tbody>
</table>

Classification 4
Load

Properties
Reconstructs the segment object from a structured string. Counterpart of the Save method.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>From</td>
<td>Character string containing structure information.</td>
</tr>
<tr>
<td>Pos</td>
<td>Current search position within the character string.</td>
</tr>
</tbody>
</table>

Classification 5
Load

Properties
This method reconstructs the segment list from a structured string. Counterpart of the Save method.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>From</td>
<td>Character string containing structure information.</td>
</tr>
<tr>
<td>Pos</td>
<td>Current search position within the character string.</td>
</tr>
</tbody>
</table>

Classification 6
Load

Properties
This method reconstructs an object of the Link class from the serialized representation. It assumes that the next token is located at the Pos position in the serialized representation of the “Fld” keyword.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>From</td>
<td>Serialization string.</td>
</tr>
</tbody>
</table>
Load (Method)

Classification 7

Load

- SAPForm class [Seite 1772]
  - SAPforms Form Plug-In for Visual Basic [Seite 1756]
- SAPForm class [Seite 1812]
  - SAPforms Form Plug-In for MS Outlook [Seite 1796]
- SAPForm class [Seite 1872]
  - SAPforms Form Plug-In for MS Office [Seite 1856]

Properties
Method for loading form controls into Controls Collection from a form or form template. Note that the form information is not stored in the Tag property of this object, but reconstructed each time the form is loaded. This method is called at design time and at runtime. It has to populate the Controls collection for both cases. At runtime, the method is called by the user code in the form's Load method. When the form is designed, the method is called by the SAPForms Designer or from outside SAPForms. If you are using the SAPForms Designer as a stand-alone application, the Load method is called with Mode=2. This instructs the Load method to load the form from a file. If you are using the SAPForms Designer as a plug-in, the Load method is invoked from the hosting environment and receives the proprietary form description as a parameter. The populated SAPForm object should be passed to the SAPForms Designer by the PlugIn method of the Designer class.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Frm</td>
<td>form or form template containing controls.</td>
</tr>
<tr>
<td>Mode</td>
<td>0=design time (add-in), 1=design time (file), 2=runtime.</td>
</tr>
</tbody>
</table>

Classification 8

Load

- SAPXmit class [Seite 2106]
  - SAPforms Data Plug-In for Demo [Seite 2077]
- SAPXmit class [Seite 1837]
  - SAPforms Form Plug-In for Outlook [Seite 1796]
- SAPXmit class [Seite 2126]
  - SAPforms Transmit Plug-In for RFC [Seite 2125]
- SAPXmit class [Seite 2168]
  - SAPforms Transmit Plug-In for MAPI [Seite 2146]
- SAPXmit class [Seite 2147]
  - SAPforms Transmit Plug-In for SAPMAPI [Seite 2167]
- SAPXmit class [Seite 2189]
  - SAPforms Transmit Plug-In for WinSock [Seite 2188]
- SAPXmit class [Seite 2209]
  - SAPforms Transmit Plug-In for Demo [Seite 2208]
Load (Method)

Properties
Loads the property values of an SAPXmit object from a structured string. As a result, the status of the SAPXmit object that was previously saved permanently with the Save method is restored.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>From</td>
<td>Character string containing the structure information</td>
</tr>
</tbody>
</table>
Name (Property)

Definition
Name of the object, field, segment, or control.

Remarks
This name, for example, must be used as the root in the SAP structure tree that is built by the BuildTree method.

Classification 1
Name

Returns the name of the workflow to whose task the container description belongs.

Classification 2
Name

Name of the R/3 function module that is to be executed.

Classification 3
Name

Name of the IDoc as it appears in the control record.

Classification 4
Name

Classification 5
Name
Properties
Name of the field.

Classification 6
Name

𝘂ếSegment class [Seite 2055]
	*SAPforms Data Plug-In for IDoc [Seite 2006]*

Properties
Name of the segment.
Used as a key in the *Templates* list of the IDoc object. Only the first attribute with this name is added to the list of segments with a key. All other segment attributes with the same name are added without a key.

Classification 7
Name

𝐮ếSAPForm class [Seite 1772]
	*SAPforms Form Plug-In for Visual Basic [Seite 1756]*
	*SAPForm Class [Seite 1812]*
	*SAPforms Form Plug-In for MS Outlook [Seite 1796]*
	*SAPForm class [Seite 1872]*
	*SAPforms Form Plug-In for MS Office [Seite 1856]*

Properties
Original name of the control in the real form (e.g. Visual Basic form).
Save (Method)

**Definition**
Saves the updated form definition.

**Remarks**
This method is used by the SAPforms Designer when a form is designed to save the modified form definition after the binding has been defined. The modifications can affect the binding definition or possibly the source code entered in the form.

**Classification 1**

**Save**

<table>
<thead>
<tr>
<th>Classification 1</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td><code>SAPData class [Seite 1918]</code></td>
<td></td>
</tr>
<tr>
<td><code>SAPforms Data Plug-In for Workflow [Seite 1897]</code></td>
<td></td>
</tr>
<tr>
<td><code>SAPData class [Seite 1975]</code></td>
<td></td>
</tr>
<tr>
<td><code>SAPforms Data Plug-In for Function Modules [Seite 1954]</code></td>
<td></td>
</tr>
<tr>
<td><code>SAPData class [Seite 2027]</code></td>
<td></td>
</tr>
<tr>
<td><code>SAPform Data Plug-In for IDoc [Seite 2006]</code></td>
<td></td>
</tr>
<tr>
<td><code>SAPData class [Seite 2078]</code></td>
<td></td>
</tr>
<tr>
<td><code>SAPform Data Plug-In for Demo [Seite 2077]</code></td>
<td></td>
</tr>
</tbody>
</table>

**Properties**
Saves the current property values of an SAPData object in a structured string. Counterpart of the Load method.

When a form is created, the SAPforms Designer uses this method to save the structure description retrieved from the R/3 System permanently in the form.

The R/3 structure is restored from the structured string at runtime by the `Load` method.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Tgt</td>
<td>String in which the status is stored.</td>
</tr>
</tbody>
</table>

**Classification 2**

**Save**

<table>
<thead>
<tr>
<th>Classification 2</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td><code>Field Class [Seite 2007]</code></td>
<td></td>
</tr>
<tr>
<td><code>SAPforms Data Plug-In for Workflow [Seite 1897]</code></td>
<td></td>
</tr>
<tr>
<td><code>SAPforms Data Plug-In for Function Modules [Seite 1954]</code></td>
<td></td>
</tr>
<tr>
<td><code>SAPforms Data Plug-In for IDoc [Seite 2006]</code></td>
<td></td>
</tr>
</tbody>
</table>

**Properties**
Saves the current property values of an SAPData object in a structured string. Counterpart of the Load method. The string is saved with a form so that the structure information is available when the form is executed.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Tgt</td>
<td>Stores the string structure.</td>
</tr>
</tbody>
</table>
Save (Method)

**Classification 3**

**Save**

*Fields Class [Seite 2018]*

*SAPforms Data Plug-In for Workflow [Seite 1897]*

*SAPforms Data Plug-In for Function Modules [Seite 1954]*

*SAPforms Data Plug-In for IDoc [Seite 2006]*

**Properties**

Loads the property values of a fields object from a structured string. As a result, the status of the fields object that was previously saved permanently with the Save method is restored. The string is saved with the form so that the structure information is available when the form is executed.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Tgt</td>
<td>Stores the string structure.</td>
</tr>
</tbody>
</table>

**Classification 4**

**Save**

*Segment class [Seite 2055]*

*SAPforms Data Plug-In for IDoc [Seite 2006]*

**Properties**

This method saves the structure of a segment, including the field list, in a structured string.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Tgt</td>
<td>Stores the string structure.</td>
</tr>
</tbody>
</table>

**Classification 5**

**Save**

*Segments class [Seite 2067]*

*SAPforms Data Plug-In for IDoc [Seite 2006]*

**Properties**

This method saves the segment list in a structured string. Counterpart of the Load method.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Tgt</td>
<td>Stores the string structure.</td>
</tr>
</tbody>
</table>

**Classification 6**

**Save**

*Links class [Seite 1862]*

*SAPforms Form Plug-In for Visual Basic [Seite 1756]*

*Links class [Seite 1862]*

*SAPforms Form Plug-In for MS Office [Seite 1856]*

*Links class [Seite 1862]*

*SAPforms Form Plug-In for MS Outlook [Seite 1796]*
Properties

Method that writes the content of all links to a file. Counterpart of the Load method.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Tgt</td>
<td>Stores the string structure.</td>
</tr>
</tbody>
</table>

Classification 7

Save

SAPForm class [Seite 1772]
SAPforms Form Plug-In for Visual Basic [Seite 1756]
SAPForm class [Seite 1872]
SAPforms Form Plug-In for MS Office [Seite 1856]
SAPForm class [Seite 1812]
SAPforms Form Plug-In for MS Outlook [Seite 1796]

Properties

Method that saves a form to a file if it was created from a file.
Reads the old file and writes updated lines to the new file.

Classification 8

Save

SAPXmit class [Seite 2106]
SAPforms Data Plug-In for Demo [Seite 2077]
SAPXmit class [Seite 1837]
SAPforms Form Plug-In for Outlook [Seite 1796]
SAPXmit class [Seite 2126]
SAPforms Transmit Plug-In for RFC [Seite 2125]
SAPXmit class [Seite 2168]
SAPforms Transmit Plug-In for MAPI [Seite 2146]
SAPXmit class [Seite 2147]
SAPforms Transmit Plug-In for SAPMAPI [Seite 2167]
SAPXmit class [Seite 2189]
SAPforms Transmit Plug-In for WinSock [Seite 2188]
SAPXmit class [Seite 2209]
SAPforms Transmit Plug-In for Demo [Seite 2208]

Properties

Saves the current property values of an SAPXmit object in a structured string. Counterpart of the Load method.

When a form is created, the SAPforms Designer uses this method to save the structure description retrieved from the R/3 System permanently in the form.
The R/3 structure is restored from the structured string at runtime by the \textit{Load} method.
<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Tgt</td>
<td>String in which the status is stored.</td>
</tr>
</tbody>
</table>
**Fields Class**

**Definition**
List class that describes the field list in an R/3 data structure.

**Remark**
This class can contain the following lists:
elements in a workflow container
segment fields in an IDoc
function module parameters.
The individual elements in the list can be accessed with the *Item* property or with a *For ... Each* loop.
The R/3 data structures often represent hierarchies. To identify a field in one of these data structures, therefore, you must specify the fully name of this field.

```plaintext
... set objFld = SAPData.Fields.Item("[StruCturX].[FieldY]") ... 
```

**Classification**
- SAPforms Data Plug-In for Workflow [Seite 1897]
- SAPforms Data Plug-In for IDoc [Seite 2006]
- SAPforms Data Plug-In for Function Modules [Seite 1954]

→ *Fields Class*
  → *Load (Method) [Seite 2213]*
  → *Save (Method) [Seite 2219]*
Load (Method)

Definition
Loads structure information to SAPforms objects.

Remarks

Classification 1

Load

 Properties
Loads the property values of an SAPData object from a structured string. As a result, the status of the SAPData object that was previously saved permanently with the Save method is restored. This saves the additional overheads that would result if the structure information had to be loaded each time from the original source. This method is usually called within the Load method of the SAPForm class.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>From</td>
<td>Character string containing the structure information</td>
</tr>
</tbody>
</table>

Classification 2

Load

 Properties
Loads the property values of a field object from a structured string. As a result, the status of the field object that was previously saved permanently with the Save method is restored. For this method to be called successfully, the next character at the Pos position must be keyword “Fld”.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>From</td>
<td>Structured string</td>
</tr>
<tr>
<td>Pos</td>
<td>Current position within the string. Should point to the number of remaining properties to read</td>
</tr>
</tbody>
</table>

Classification 3

Load
Load (Method)

Properties
Loads the property values of a fields object from a structured string. As a result, the status of the fields object that was previously saved permanently with the Save method is restored.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>From</td>
<td>Character string that provides the structure information.</td>
</tr>
<tr>
<td>Pos</td>
<td>Current search position within the character string.</td>
</tr>
</tbody>
</table>

Classification 4
Load

Properties
Reconstructs the segment object from a structured string. Counterpart of the Save method.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>From</td>
<td>Character string containing structure information.</td>
</tr>
<tr>
<td>Pos</td>
<td>Current search position within the character string.</td>
</tr>
</tbody>
</table>

Classification 5
Load

Properties
This method reconstructs the segment list from a structured string. Counterpart of the Save method.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>From</td>
<td>Character string containing structure information.</td>
</tr>
<tr>
<td>Pos</td>
<td>Current search position within the character string.</td>
</tr>
</tbody>
</table>

Classification 6
Load

Properties
This method reconstructs an object of the Link class from the serialized representation. It assumes that the next token is located at the Pos position in the serialized representation of the “Fld” keyword.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>From</td>
<td>Serialization string.</td>
</tr>
</tbody>
</table>
Load (Method)

Pos | Current position within the character string.

Classification 7

Load

←SAPForm class [Seite 1772]
←SAPForm class [Seite 1812]
←SAPForm class [Seite 1872]
←SAPforms Form Plug-In for Visual Basic [Seite 1756]
←SAPforms Form Plug-In for MS Outlook [Seite 1796]
←SAPforms Form Plug-In for MS Office [Seite 1856]

Properties

Method for loading form controls into Controls Collection from a form or form template. Note that the form information is not stored in the Tag property of this object, but reconstructed each time the form is loaded. This method is called at design time and at runtime. It has to populate the Controls collection for both cases. At runtime, the method is called by the user code in the form's Load method. When the form is designed, the method is called by the SAPForms Designer or from outside SAPForms. If you are using the SAPForms Designer as a stand-alone application, the Load method is called with Mode=2. This instructs the Load method to load the form from a file. If you are using the SAPForms Designer as a plug-in, the Load method is invoked from the hosting environment and receives the proprietary form description as a parameter. The populated SAPForm object should be passed to the SAPForms Designer by the PlugIn method of the Designer class.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Frm</td>
<td>form or form template containing controls.</td>
</tr>
<tr>
<td>Mode</td>
<td>0=design time (add-in), 1=design time (file), 2=runtime.</td>
</tr>
</tbody>
</table>

Classification 8

Load

←SAPXmit class [Seite 2106]
←SAPXmit class [Seite 1837]
←SAPXmit class [Seite 2126]
←SAPXmit class [Seite 2168]
←SAPXmit class [Seite 2147]
←SAPXmit class [Seite 2168]
←SAPforms Transmit Plug-In for SAPMAPI [Seite 2146]
←SAPforms Transmit Plug-In for WinSock [Seite 2188]
←SAPforms Transmit Plug-In for Demo [Seite 2208]
Properties
Loads the property values of an SAPXmit object from a structured string. As a result, the status of the SAPXmit object that was previously saved permanently with the Save method is restored.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>From</td>
<td>Character string containing the structure information</td>
</tr>
</tbody>
</table>
Save (Method)

Definition
Saves the updated form definition.

Remarks
This method is used by the SAPforms Designer when a form is designed to save the modified form definition after the binding has been defined. The modifications can affect the binding definition or possibly the source code entered in the form.

Classification 1
Save

Properties
Saves the current property values of an SAPData object in a structured string. Counterpart of the Load method.
When a form is created, the SAPforms Designer uses this method to save the structure description retrieved from the R/3 System permanently in the form.
The R/3 structure is restored from the structured string at runtime by the Load method.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Tgt</td>
<td>String in which the status is stored.</td>
</tr>
</tbody>
</table>

Classification 2
Save

Properties
Saves the current property values of an SAPData object in a structured string. Counterpart of the Load method. The string is saved with a form so that the structure information is available when the form is executed.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Tgt</td>
<td>Stores the string structure.</td>
</tr>
</tbody>
</table>
Classification 3

Save

Properties
Loads the property values of a fields object from a structured string. As a result, the status of the fields object that was previously saved permanently with the Save method is restored. The string is saved with the form so that the structure information is available when the form is executed.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Tgt</td>
<td>Stores the string structure.</td>
</tr>
</tbody>
</table>

Classification 4

Save

Properties
This method saves the structure of a segment, including the field list, in a structured string.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Tgt</td>
<td>Stores the string structure.</td>
</tr>
</tbody>
</table>

Classification 5

Save

Properties
This method saves the segment list in a structured string. Counterpart of the Load method.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Tgt</td>
<td>Stores the string structure.</td>
</tr>
</tbody>
</table>

Classification 6

Save

Properties

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Tgt</td>
<td>Stores the string structure.</td>
</tr>
</tbody>
</table>
Properties
Method that writes the content of all links to a file. Counterpart of the Load method.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Tgt</td>
<td>Stores the string structure.</td>
</tr>
</tbody>
</table>

Classification 7
Save

]<< SAPForm class [Seite 1772]
<< SAPforms Form Plug-In for Visual Basic [Seite 1756]
<< SAPForm class [Seite 1872]
<< SAPforms Form Plug-In for MS Office [Seite 1856]
<< SAPForm class [Seite 1812]
<< SAPforms Form Plug-In for MS Outlook [Seite 1796]

Properties
Method that saves a form to a file if it was created from a file.
Reads the old file and writes updated lines to the new file.

Classification 8
Save

<< SAPXmit class [Seite 2106]
<< SAPforms Data Plug-In for Demo [Seite 2077]
<< SAPXmit class [Seite 1837]
<< SAPforms Form Plug-In for Outlook [Seite 1796]
<< SAPXmit class [Seite 2126]
<< SAPforms Transmit Plug-In for RFC [Seite 2125]
<< SAPXmit class [Seite 2168]
<< SAPforms Transmit Plug-In for MAPI [Seite 2146]
<< SAPXmit class [Seite 2147]
<< SAPforms Transmit Plug-In for SAPMAPI [Seite 2167]
<< SAPXmit class [Seite 2189]
<< SAPforms Transmit Plug-In for WinSock [Seite 2188]
<< SAPXmit class [Seite 2209]
<< SAPforms Transmit Plug-In for Demo [Seite 2208]

Properties
Saves the current property values of an SAPXmit object in a structured string. Counterpart of the Load method.
When a form is created, the SAPforms Designer uses this method to save the structure description retrieved from the R/3 System permanently in the form.
The R/3 structure is restored from the structured string at runtime by the Load method.
<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Tgt</td>
<td>String in which the status is stored.</td>
</tr>
</tbody>
</table>
SAPData Class

Definition
Original class of this plug-in; representation of an IDoc.

Remarks
This class provides basic access to the segments of an IDoc and the fields in these segments. The `Segments` property refers to an object in the `Segments` list class that represents the IDoc segments. The list of corresponding segment fields is located in the `Fields` object of this plug-in. The fields and segments are accessed via the `Value` property. The `Start` method starts a workflow using an IDoc.

SAPData plug-in type classes implement the SAPData interface that provides the SAPforms Designer and other SAPforms components with access to certain types of R/3 data structures and interfaces. In other words, these classes can be regarded as being synonymous with various plug-ins of this type.

A SAPData object is usually also instantiated with the SAPForm object. The SAPData object is then referenced by the SAPData property of the SAPForm object. An object of the SAPData class, however, cannot exist without an object of the SAPForms class.

If you want to develop your own SAPData-type plug-ins, you should use the source code supplied for the `SAPforms Data Plug-Ins for Demo` as a reference (template). The template is stored in the directory `...\sapforms\sapddemo`.

Classification

**SAPforms Data Plug-In for IDoc [Site 2006]**
- SAPData Class
  - Connection (Property) [Site 2028]
  - Execute (Method) [Site 2210]
  - Load (Method) [Site 2213]
  - LoadInstance (Method) [Site 2086]
  - Name (Property) [Site 2088]
  - Options (Method) [Site 2217]
  - R3DoNotUse (Property) [Site 2137]
  - SAPXmit (Property) [Site 2093]
  - Save (Method) [Site 2219]
  - SaveInstance (Method) [Site 2098]
  - Start (Method) [Site 2223]
  - Value (Property) [Site 2103]
  - Version (Property) [Site 2105]
Connection (Property)

Definition
Sets or returns the object of the RFC connection.

Remarks

Classification
Connection

← SAPData class [Seite 1918]
← SAPforms Data Plug-In for Workflow [Seite 1897]
← SAPData class [Seite 1975]
← SAPforms Data Plug-In for Function Modules [Seite 1954]
← SAPData class [Seite 2027]
← SAPforms Data Plug-In for IDoc [Seite 2006]
Execute (Method)

Definition
Method that calls the *Execute* function in the R/3 System. The semantics of this function depend on the SAPData object.

Remarks
The method can be called directly via the SAPData object or indirectly via the Start method of the SAPForm object, whose SAPData property is a reference to the SAPData object.

Classification 1
Execute

- SAPData class [Seite 1918]
- SAPforms Data Plug-In for Workflow [Seite 1897]

Properties
Terminates a work item in the R/3 System and supplies its container elements with the data values in the fields list.

Classification 2
Execute

- SAPData class [Seite 2078]
- SAPXmit class [Seite 2106]
- SAPforms Data Plug-In for Demo [Seite 2077]

Properties
Does not execute an action in the R/3 System but rather displays the data of the object in a separate window.

Classification 3
Execute

- SAPForm class [Seite 1772]
- SAPforms Form Plug-In for Visual Basic [Seite 1756]
- SAPForm class [Seite 1872]
- SAPforms Form Plug-In for MS Office [Seite 1856]
- SAPForm class [Seite 1812]
- SAPforms Form Plug-In for MS Outlook [Seite 1796]
Properties
Invokes the Execute method in the associated Xmit or SAPData object. The semantics of this method depend on this object.

Classification 4
Execute

Properties
This method executes a work item synchronously via a remote function call (RFC). Supports SAPData object of type *Container*.
This method causes a certain function module to be executed in the R/3 System. This module receives a table with a complete R3F structure containing the work item information and the container data that are to be supplied from the form.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Data</td>
<td>Object representing the container data.</td>
</tr>
</tbody>
</table>

Classification 5
Execute

Properties
Sends a mail to the R/3 System that contains the information required to terminate a work item (including container data) as an attachment.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Data</td>
<td>Object representing the container data.</td>
</tr>
</tbody>
</table>

Classification 6
Execute

Properties
Overwrites the R3F attachment, thereby updating it.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Pdata</td>
<td>Object that represents the <code>mobjData</code> container data</td>
</tr>
</tbody>
</table>

Classification 7
Execute

Properties

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Properties
Displays the instance data of the associated SAPData object in a window instead of sending it to the R/3 System.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Data</td>
<td>object of class SAPData</td>
</tr>
</tbody>
</table>
Load (Method)

Definition
Loads structure information to SAPforms objects.

Remarks

Classification 1
Load

<table>
<thead>
<tr>
<th>Classification 1</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>From</td>
<td></td>
</tr>
<tr>
<td>Pos</td>
<td></td>
</tr>
</tbody>
</table>

Properties
Loads the property values of an SAPData object from a structured string. As a result, the status of the SAPData object that was previously saved permanently with the Save method is restored. This saves the additional overheads that would result if the structure information had to be loaded each time from the original source. This method is usually called within the Load method of the SAPForm class.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>From</td>
<td>Character string containing the structure information</td>
</tr>
</tbody>
</table>

Classification 2
Load

<table>
<thead>
<tr>
<th>Classification 2</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>From</td>
<td></td>
</tr>
<tr>
<td>Pos</td>
<td></td>
</tr>
</tbody>
</table>

Properties
Loads the property values of a field object from a structured string. As a result, the status of the field object that was previously saved permanently with the Save method is restored. For this method to be called successfully, the next character at the Pos position must be keyword “Fld”.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>From</td>
<td>Structured string</td>
</tr>
<tr>
<td>Pos</td>
<td>Current position within the string. Should point to the number of remaining properties to read.</td>
</tr>
</tbody>
</table>

Classification 3
Load
**Properties**

Loads the property values of a fields object from a structured string. As a result, the status of the fields object that was previously saved permanently with the Save method is restored.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>From</td>
<td>Character string that provides the structure information.</td>
</tr>
<tr>
<td>Pos</td>
<td>Current search position within the character string.</td>
</tr>
</tbody>
</table>

**Classification 4**

Load

**Properties**

Reconstructs the segment object from a structured string. Counterpart of the Save method.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>From</td>
<td>Character string containing structure information.</td>
</tr>
<tr>
<td>Pos</td>
<td>Current search position within the character string.</td>
</tr>
</tbody>
</table>

**Classification 5**

Load

**Properties**

This method reconstructs the segment list from a structured string. Counterpart of the Save method.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>From</td>
<td>Character string containing structure information.</td>
</tr>
<tr>
<td>Pos</td>
<td>Current search position within the character string.</td>
</tr>
</tbody>
</table>

**Classification 6**

Load

**Properties**

This method reconstructs an object of the Link class from the serialized representation. It assumes that the next token is located at the Pos position in the serialized representation of the “Fld” keyword.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>From</td>
<td>Serialization string.</td>
</tr>
</tbody>
</table>
Load (Method)

<table>
<thead>
<tr>
<th>Pos</th>
<th>Current position within the character string.</th>
</tr>
</thead>
</table>

**Classification 7**

*Load*

- SAPForm class [Seite 1772]
- SAPforms Form Plug-In for Visual Basic [Seite 1756]
- SAPForm class [Seite 1812]
- SAPforms Form Plug-In for MS Outlook [Seite 1796]
- SAPForm class [Seite 1872]
- SAPforms Form Plug-In for MS Office [Seite 1856]

**Properties**

Method for loading form controls into Controls Collection from a form or form template. Note that the form information is not stored in the Tag property of this object, but reconstructed each time the form is loaded. This method is called at design time and at runtime. It has to populate the Controls collection for both cases. At runtime, the method is called by the user code in the form's Load method. When the form is designed, the method is called by the SAPForms Designer or from outside SAPForms. If you are using the SAPForms Designer as a stand-alone application, the Load method is called with Mode=2. This instructs the Load method to load the form from a file. If you are using the SAPForms Designer as a plug-in, the Load method is invoked from the hosting environment and receives the proprietary form description as a parameter. The populated SAPForm object should be passed to the SAPForms Designer by the PlugIn method of the Designer class.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Frm</td>
<td>form or form template containing controls.</td>
</tr>
<tr>
<td>Mode</td>
<td>0=design time (add-in), 1=design time (file), 2=runtime.</td>
</tr>
</tbody>
</table>

**Classification 8**

*Load*

- SAPXmit class [Seite 2106]
- SAPforms Data Plug-In for Demo [Seite 2077]
- SAPXmit class [Seite 1837]
- SAPforms Form Plug-In for Outlook [Seite 1796]
- SAPXmit class [Seite 2126]
- SAPforms Transmit Plug-In for RFC [Seite 2125]
- SAPXmit class [Seite 2168]
- SAPforms Transmit Plug-In for MAPI [Seite 2146]
- SAPXmit class [Seite 2147]
- SAPforms Transmit Plug-In for SAPMAPI [Seite 2167]
- SAPXmit class [Seite 2189]
- SAPforms Transmit Plug-In for WinSock [Seite 2188]
- SAPXmit class [Seite 2209]
- SAPforms Transmit Plug-In for Demo [Seite 2208]
**Properties**

Loads the property values of an SAPXmit object from a structured string. As a result, the status of the SAPXmit object that was previously saved permanently with the Save method is restored.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>From</td>
<td>Character string containing the structure information</td>
</tr>
</tbody>
</table>
LoadInstance (Method)

Definition
Method that loads the data content of a SAPData object from a file.

Remarks
The method is usually called when data flows from the R/3 System to the form and is to be displayed by the form. This is the case, for example, when a work item is executed. The form is then started by the SAPforms Helper [Seite 1741] and the R3F file is forwarded as a parameter. The existence of a parameter when the form is started ensures that the LoadInstance method is called in the Init() method of the SAPform object. The GetData method is usually called after the LoadInstance method.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Path</td>
<td>Complete path designation</td>
</tr>
</tbody>
</table>

Classification 1
LoadInstance
- [SAPData Class [Seite 1918]]
- [SAPforms Data Plug-In for Workflow [Seite 1897]]

Properties
Loads the R/3 workflow information from an R3F file to the SAPData object. The current container element values can be interrogated and manipulated via the SAPData.Value and SAPData.Fields(s).Data properties in particular.

Classification 2
LoadInstance
- [SAPData Class [Seite 1975]]
- [SAPforms Data Plug-In for Function Modules [Seite 1954]]

Properties
Implementation of the SAPData interface for R/3 function modules (BAPIs).

Classification 3
LoadInstance
- [SAPData Class [Seite 2027]]
- [SAPforms Data Plug-In for IDoc [Seite 2006]]

Properties
Loads an IDoc instance from a file.
Builds the segments list from the records in the file. This method will only function correctly if the Templates list containing the entire IDoc meta structure exists. The structure of each segment in the Segments list is derived from the corresponding segment in the Templates list.

Classification 4
LoadInstance
Properties

The method saves the contents of a field in the FieldValues array. This method requires that the elements be saved in a fixed order. Even though the field name is contained in the file, it is not evaluated.
**Name (Property)**

**Definition**
Name of the object, field, segment, or control.

**Remarks**
This name, for example, must be used as the root in the SAP structure tree that is built by the `BuildTree` method.

**Classification 1**
**Name**
- SAPData class [Seite 1918]
- SAPforms Data Plug-In for Workflow [Seite 1897]

**Properties**
Returns the name of the workflow to whose task the container description belongs.

**Classification 2**
**Name**
- SAPData class [Seite 1975]
- SAPforms Data Plug-In for Function Modules [Seite 1954]

**Properties**
Name of the R/3 function module that is to be executed.

**Classification 3**
**Name**
- SAPData class [Seite 2027]
- SAPforms Data Plug-In for IDoc [Seite 2006]

**Properties**
Name of the IDoc as it appears in the control record.

**Classification 4**
**Name**
- SAPData class [Seite 2078]
- SAPforms Data Plug-In for Demo [Seite 2077]

**Classification 5**
**Name**
- Field Class [Seite 2007]
- SAPforms Data Plug-In for Workflow [Seite 1897]
- SAPforms Data Plug-In for Function Modules [Seite 1954]
- SAPforms Data Plug-In for IDoc [Seite 2006]
Properties
Name of the field.

Classification 6
Name

← Segment class [Seite 2055]
← SAPforms Data Plug-In for IDoc [Seite 2006]

Properties
Name of the segment.
Used as a key in the Templates list of the IDoc object. Only the first attribute with this name is added to the list of segments with a key. All other segment attributes with the same name are added without a key.

Classification 7
Name

← SAPForm class [Seite 1772]
← SAPforms Form Plug-In for Visual Basic [Seite 1756]
← SAPForm Class [Seite 1812]
← SAPforms Form Plug-In for MS Outlook [Seite 1796]
← SAPForm class [Seite 1872]
← SAPforms Form Plug-In for MS Office [Seite 1856]

Properties
Original name of the control in the real form (e.g. Visual Basic form).
Options (Method)

Definition
Method that outputs the object-specific options dialog box.

Remarks
This dialog box displays either global settings or settings that are valid for the current form. These settings can be changed here. The global settings are read from the registry before they are displayed and are then saved again to the registry.

The method is called within the SAPforms Designer if the required plug-in is active under Properties on the Plug-Ins tab in the Options dialog. You can also use the sample program Options.exe to display the Options dialog of a certain plug-in.

Classification 1
Options

- SAPData class [Seite 1918]
- SAPforms Data Plug-In for Workflow [Seite 1897]
- SAPData class [Seite 1975]
- SAPforms Data Plug-In for Function Modules [Seite 1954]
- SAPData class [Seite 2027]
- SAPforms Data Plug-In for IDoc [Seite 2006]
- SAPData class [Seite 2078]
- SAPforms Data Plug-In for Demo [Seite 2077]

Classification 2
Options

- SAPForm class [Seite 1772]
- SAPforms Form Plug-In for Visual Basic [Seite 1756]
- SAPForm class [Seite 1812]
- SAPforms Form Plug-In for MS Outlook [Seite 1796]
- SAPForm class [Seite 1872]
- SAPforms Form Plug-In for MS Office [Seite 1856]

Classification 3
Options

- SAPXmit class [Seite 2106]
- SAPforms Data Plug-In for Demo [Seite 2077]
- SAPXmit class [Seite 1837]
- SAPforms Form Plug-In for MS Outlook [Seite 1796]
- SAPXmit class [Seite 2126]
<- SAPforms Transmit Plug-In for RFC [Seite 2125]
<- SAPXmit class [Seite 2168]
<- SAPforms Transmit Plug-In for MAPI [Seite 2146]
<- SAPXmit class [Seite 2147]
<- SAPforms Transmit Plug-In for SAPMAPI [Seite 2167]
<- SAPXmit class [Seite 2189]
<- SAPforms Transmit Plug-In for SMTP [Seite 2188]
<- SAPXmit class [Seite 2209]
<- SAPforms Transmit Plug-In for Demo [Seite 2208]
R3DoNotUse (Property)

Definition
Determines whether the R/3 system information, which was stored permanently in the form when it was designed, is to be used when the Start() and Execute() methods are executed.

Remarks

Classification

R3DoNotUse

SAPData class [Seite 1918]
SAPforms Data Plug-In for Workflow [Seite 1897]

SAPData class [Seite 1975]
SAPforms Data Plug-In for Function Modules [Seite 1954]

SAPData class [Seite 2027]
SAPforms Data Plug-In for IDoc [Seite 2006]

SAPXmit class [Seite 2126]
SAPforms Transmit Plug-In for RFC [Seite 2125]

Properties
The default value of this field is FALSE. When the methods are executed, logon takes place to the R/3 System that was used to design the form. This R/3 System selection is skipped during logon. If the same form is to be used with different R/3 Systems, you must set this property to TRUE. You can specify this either in the form's program code or by using the options dialog of the SAPData object.
**SAPXmit (Property)**

**Definition**
Reference to the SAPXmit object that is used to encapsulate the transport of SAPData to the R/3 System.

**Remarks**

**Classification**

SAPXmit

← SAPData class [Seite 1918]
← SAPforms Data Plug-In for Workflow [Seite 1897]
← SAPData class [Seite 1975]
← SAPforms Data Plug-In for Function Modules [Seite 1954]
← SAPData class [Seite 2027]
← SAPforms Data Plug-In for IDoc [Seite 2006]
← SAPData class [Seite 2078]
← SAPforms Data Plug-In for Demo [Seite 2077]
← SAPForm class [Seite 1772]
← SAPforms Form Plug-In for Visual Basic [Seite 1756]
← SAPForm class [Seite 1812]
← SAPforms Form Plug-In for MS Outlook [Seite 1796]
← SAPForm class [Seite 1872]
← SAPforms Form Plug-In for MS Office [Seite 1856]
Save (Method)

Definition
Saves the updated form definition.

Remarks
This method is used by the SAPforms Designer when a form is designed to save the modified form definition after the binding has been defined. The modifications can affect the binding definition or possibly the source code entered in the form.

Classification 1
Save

Properties
Saves the current property values of an SAPData object in a structured string. Counterpart of the Load method.
When a form is created, the SAPforms Designer uses this method to save the structure description retrieved from the R/3 System permanently in the form.
The R/3 structure is restored from the structured string at runtime by the Load method.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Tgt</td>
<td>String in which the status is stored.</td>
</tr>
</tbody>
</table>

Classification 2
Save

Properties
Saves the current property values of an SAPData object in a structured string. Counterpart of the Load method. The string is saved with a form so that the structure information is available when the form is executed.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Tgt</td>
<td>Stores the string structure.</td>
</tr>
</tbody>
</table>
Classification 3

Save

Properties

Loads the property values of a fields object from a structured string. As a result, the status of the fields object that was previously saved permanently with the Save method is restored. The string is saved with the form so that the structure information is available when the form is executed.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Tgt</td>
<td>Stores the string structure.</td>
</tr>
</tbody>
</table>

Classification 4

Save

Properties

This method saves the structure of a segment, including the field list, in a structured string.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Tgt</td>
<td>Stores the string structure.</td>
</tr>
</tbody>
</table>

Classification 5

Save

Properties

This method saves the segment list in a structured string. Counterpart of the Load method.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Tgt</td>
<td>Stores the string structure.</td>
</tr>
</tbody>
</table>

Classification 6

Save

Properties
**Properties**

Method that writes the content of all links to a file. Counterpart of the Load method.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Tgt</td>
<td>Stores the string structure.</td>
</tr>
</tbody>
</table>

**Classification 7**

Save

- [SAPForm class][1772]
- [SAPforms Form Plug-In for Visual Basic][1756]
- [SAPForm class][1872]
- [SAPforms Form Plug-In for MS Office][1856]
- [SAPForm class][1812]
- [SAPforms Form Plug-In for MS Outlook][1796]

**Properties**

Method that saves a form to a file if it was created from a file. Reads the old file and writes updated lines to the new file.

**Classification 8**

Save

- [SAPXmit class][2106]
- [SAPforms Data Plug-In for Demo][2077]
- [SAPXmit class][1837]
- [SAPforms Form Plug-In for Outlook][1796]
- [SAPXmit class][2126]
- [SAPforms Transmit Plug-In for RFC][2125]
- [SAPXmit class][2168]
- [SAPforms Transmit Plug-In for MAPI][2146]
- [SAPXmit class][2147]
- [SAPforms Transmit Plug-In for SAPMAPI][2167]
- [SAPXmit class][2189]
- [SAPforms Transmit Plug-In for WinSock][2188]
- [SAPXmit class][2209]
- [SAPforms Transmit Plug-In for Demo][2208]

**Properties**

Saves the current property values of an SAPXmit object in a structured string. Counterpart of the Load method.

When a form is created, the SAPforms Designer uses this method to save the structure description retrieved from the R/3 System permanently in the form.

The R/3 structure is restored from the structured string at runtime by the *Load* method.
<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Tgt</td>
<td>String in which the status is stored.</td>
</tr>
</tbody>
</table>
SaveInstance (Method)

Definition
This method saves the current data content of the SAPData object to the specified file.

Remarks
This data content of the SAPData object is the result of the PutData method of the SAPForm object or the result of assignments to the SAPData method in the SAPData object. The method is usually called from the SAPXmit object to save the data content in a file before it is sent by email, for example.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Path</td>
<td>Complete path specifying where the data content is to be saved.</td>
</tr>
</tbody>
</table>

Classification 1
SaveInstance

Properties
Method that saves the current contents of the function module to the specified file in R3F format.

Classification 2
SaveInstance

Properties
Saves the current IDoc instance segment-by-segment to the specified file. Uses the Record property of the segment object to generate a record for a segment.

Classification 3
SaveInstance

Properties
Generates a file containing a separate line for each data field.
Start (Method)

**Definition**
Method that executes the *Start* function in R/3. The semantics of this function depend on the concrete SAPData object.

**Remarks**
In the case of the SAPData object for workflow containers, this method starts a workflow and populates its container elements with values from the form fields. This method can be called directly via the SAPData object or indirectly via the *Start* method of the SAPForm object, whose SAPData property is a reference to the SAPData object. If an SAPXmit object is assigned to the SAPData object, the *Start* method of the SAPXmit object is called.

**Classification 1**

**Start**

.performance

*SAPData class [Seite 1918]*

*SAPforms Data Plug-In for Workflow [Seite 1897]*

**Properties**
Starts a workflow in the R/3 System and supplies its container elements with the data values in the fields list.

**Classification 2**

**Start**

*SAPData class [Seite 1975]*

*SAPforms Data Plug-In for Function Modules [Seite 1954]*

**Properties**
This method carries out the current R/3 function module call.

**Classification 3**

**Start**

*SAPData class [Seite 2027]*

*SAPforms Data Plug-In for IDoc [Seite 2006]*

**Properties**
Generates an IDoc file from the data in the segments list and transfers this IDoc to the R/3 System. The transfer method depends on the assigned SAPXmit object. A workflow can be started in the R/3 System when the inbound IDoc is processed.

**Classification 4**

**Start**

*SAPData class [Seite 2078]*

*SAPforms Data Plug-In for Demo [Seite 2077]*
Properties
Does not execute an action in the R/3 System but rather displays the data of the object in a separate window.

Classification 5
Start

\[\rightarrow\text{SAPForm class [Seite 1772]}\]
\[\rightarrow\text{SAPforms Form Plug-In for Visual Basic [Seite 1756]}\]
\[\rightarrow\text{SAPForm class [Seite 1872]}\]
\[\rightarrow\text{SAPforms Form Plug-In for MS Office [Seite 1856]}\]
\[\rightarrow\text{SAPForm class [Seite 1812]}\]
\[\rightarrow\text{SAPforms Form Plug-In for MS Outlook [Seite 1796]}\]

Properties
Invokes the \textit{Start} method in the associated SAPXmit or SAPData object. The semantics of this method depend on this SAPData object. If this object is the \textit{SAPData object for SAP Business Workflow}, this method starts a workflow in an R/3 System.

Classification 6
Start

\[\rightarrow\text{SAPXmit class [Seite 2126]}\]
\[\rightarrow\text{SAPforms Transmit Plug-In for RFC [Seite 2125]}\]

Properties
This method starts a workflow synchronously via RFC. Supports SAPData objects of types "Idoc" and "Container".

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Data</td>
<td>Object of type SAPData.</td>
</tr>
</tbody>
</table>

Classification 7
Start

\[\rightarrow\text{SAPXmit class [Seite 2168]}\]
\[\rightarrow\text{SAPforms Transmit Plug-In for MAPI [Seite 2146]}\]
\[\rightarrow\text{SAPXmit class [Seite 2147]}\]
\[\rightarrow\text{SAPforms Transmit Plug-In for SAPMAPI [Seite 2167]}\]
\[\rightarrow\text{SAPXmit class [Seite 2189]}\]
\[\rightarrow\text{SAPforms Transmit Plug-In for SMTP [Seite 2188]}\]
\[\rightarrow\text{SAPXmit class [Seite 1837]}\]
\[\rightarrow\text{SAPforms Form Plug-In for MS Outlook [Seite 1796]}\]

Properties
Sends a mail to the R/3 System with an attachment containing the information required to start a workflow.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Data</td>
<td>Object of type SAPData.</td>
</tr>
</tbody>
</table>
Classification 8

Start

- SAPXmit class [Seite 2209]
- SAPforms Transmit Plug-In for Demo [Seite 2208]
- SAPXmit class [Seite 2106]
- SAPforms Data Plug-In for Demo [Seite 2077]

Properties

Displays the instance data of the associated SAPData object in a window instead of sending it to the R/3 System.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Data</td>
<td>Object of type SAPData.</td>
</tr>
</tbody>
</table>
Value (Property)

Definition
Value of a specific field type.

Remarks
Classification 1

Value

Properties
The value of an SAP data field, defined in greater detail by the Qual parameter.
The structure and syntax of the Qual parameter depends on the concrete SAP data structure from which this field originates. In the simplest case, the Qual parameter designates the name of a field whose value is to be set or read. If the field is part of a structure (for example, an IDoc segment), Qual must identify the field uniquely within the overall structure. If the structure is repeated within an instance - as is the case with some IDocs - Qual must also contain a selector that identifies the IDoc segment uniquely.

\[
\text{obj.Value("Field1") = "abc".} \\
\text{Obj.Value("Seg1.Field1") = "abc".} \\
\text{X = obj.Value("Seg1.Seg2.Field1").} \\
\text{X = obj.Value("Seg1.Field1 Seg2.Field2 = '123'"')} \\
\]

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Qual</td>
<td>Qualifier and selector that uniquely identifies a particular field.</td>
</tr>
</tbody>
</table>

Classification 2

Value
Properties

Returns the value of control. This property is an abstraction of the different properties used by the original controls to present their content, for example, as Text, Caption, Value, and so on. The implementation of the SAPCtrl must map the original control property in the Value property for each control that is to be supported by SAPforms.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>X</td>
<td>Index (e.g. for grids).</td>
</tr>
<tr>
<td>Return</td>
<td>Value</td>
</tr>
</tbody>
</table>
Version (Property)

Definition
Version of the structure information (workflow) interface (function module)
IDoc class used for compatibility checks.

Remarks

Classification
Version

← SAPData class [Seite 1918]
← SAPforms Data Plug-In for Workflow [Seite 1897]
← SAPData class [Seite 1975]
← SAPforms Data Plug-In for Function Modules [Seite 1954]
← SAPData class [Seite 2027]
← SAPforms Data Plug-In for IDoc [Seite 2006]
← SAPData class [Seite 2078]
← SAPforms Data Plug-In for Demo [Seite 2077]
Segment Class

Definition
Class that represents an individual IDoc segment.

Remarks
This class contains, in particular, a reference to a Fields object that contains a list of the segment fields.

Classification
SAPforms Data Plug-In for IDoc [Seite 2006]

→ Segment Class
  → Fields (Property) [Seite 2056]
  → Load (Method) [Seite 2213]
  → Name (Property) [Seite 2088]
  → Save (Method) [Seite 2219]
Fields (Property)

Definition
List of the elements in an R/3 Structure, i.e.
the container elements in the workflow or work item that is represented by the SAPData object
the parameters contained in the function module
the fields contained in the IDoc segment.

Remarks

Classification
Fields

← SAPData class [Seite 1918]
← SAPforms Data Plug-In for Workflow [Seite 1897]
← SAPData class [Seite 1975]
← SAPforms Data Plug-In for Function Modules [Seite 1954]
← SAPData class [Seite 2027]
← SAPforms Data Plug-In for IDoc [Seite 2006]
Load (Method)

Definition
Loads structure information to SAPforms objects.

Remarks

Classification 1

Load

Properties
Loads the property values of an SAPData object from a structured string. As a result, the status of the SAPData object that was previously saved permanently with the Save method is restored. This saves the additional overheads that would result if the structure information had to be loaded each time from the original source. This method is usually called within the Load method of the SAPForm class.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>From</td>
<td>Character string containing the structure information</td>
</tr>
</tbody>
</table>

Classification 2

Load

Properties
Loads the property values of a field object from a structured string. As a result, the status of the field object that was previously saved permanently with the Save method is restored. For this method to be called successfully, the next character at the Pos position must be keyword “Fld”.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>From</td>
<td>Structured string.</td>
</tr>
<tr>
<td>Pos</td>
<td>Current position within the string. Should point to the number of remaining properties to read.</td>
</tr>
</tbody>
</table>

Classification 3

Load
Properties
Loads the property values of a fields object from a structured string. As a result, the status of the fields object that was previously saved permanently with the Save method is restored.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>From</td>
<td>Character string that provides the structure information.</td>
</tr>
<tr>
<td>Pos</td>
<td>Current search position within the character string.</td>
</tr>
</tbody>
</table>

Classification 4
Load

Properties
Reconstructs the segment object from a structured string. Counterpart of the Save method.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>From</td>
<td>Character string containing structure information.</td>
</tr>
<tr>
<td>Pos</td>
<td>Current search position within the character string.</td>
</tr>
</tbody>
</table>

Classification 5
Load

Properties
This method reconstructs the segment list from a structured string. Counterpart of the Save method.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>From</td>
<td>Character string containing structure information.</td>
</tr>
<tr>
<td>Pos</td>
<td>Current search position within the character string.</td>
</tr>
</tbody>
</table>

Classification 6
Load

Properties
This method reconstructs an object of the Link class from the serialized representation.
It assumes that the next token is located at the Pos position in the serialized representation of the “Fld” keyword.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>From</td>
<td>Serialization string.</td>
</tr>
</tbody>
</table>
Load (Method)

Current position within the character string.

**Classification 7**

**Load**

←SAPForm class [Seite 1772]

←SAPForms Form Plug-In for Visual Basic [Seite 1756]

←SAPForm class [Seite 1812]

←SAPForms Form Plug-In for MS Outlook [Seite 1796]

←SAPForm class [Seite 1872]

←SAPForms Form Plug-In for MS Office [Seite 1856]

**Properties**

Method for loading form controls into Controls Collection from a form or form template. Note that the form information is not stored in the Tag property of this object, but reconstructed each time the form is loaded. This method is called at design time and at runtime. It has to populate the Controls collection for both cases. At runtime, the method is called by the user code in the form's Load method. When the form is designed, the method is called by the SAPForms Designer or from outside SAPForms. If you are using the SAPForms Designer as a stand-alone application, the Load method is called with Mode=2. This instructs the Load method to load the form from a file. If you are using the SAPForms Designer as a plug-in, the Load method is invoked from the hosting environment and receives the proprietary form description as a parameter. The populated SAPForm object should be passed to the SAPForms Designer by the PlugIn method of the Designer class.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Frm</td>
<td>form or form template containing controls.</td>
</tr>
<tr>
<td>Mode</td>
<td>0=design time (add-in), 1=design time (file), 2=runtime.</td>
</tr>
</tbody>
</table>

**Classification 8**

**Load**

←SAPXmit class [Seite 2106]

←SAPForms Data Plug-In for Demo [Seite 2077]

←SAPXmit class [Seite 1837]

←SAPForms Form Plug-In for Outlook [Seite 1796]

←SAPXmit class [Seite 2126]

←SAPForms Transmit Plug-In for RFC [Seite 2125]

←SAPXmit class [Seite 2168]

←SAPForms Transmit Plug-In for MAPI [Seite 2146]

←SAPXmit class [Seite 2147]

←SAPForms Transmit Plug-In for SAPMAPI [Seite 2167]

←SAPXmit class [Seite 2189]

←SAPForms Transmit Plug-In for WinSock [Seite 2188]

←SAPXmit class [Seite 2209]

←SAPForms Transmit Plug-In for Demo [Seite 2208]
Load (Method)

Properties
Loads the property values of an SAPXmit object from a structured string. As a result, the status of the SAPXmit object that was previously saved permanently with the Save method is restored.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>From</td>
<td>Character string containing the structure information</td>
</tr>
</tbody>
</table>
Name (Property)

Definition
Name of the object, field, segment, or control.

Remarks
This name, for example, must be used as the root in the SAP structure tree that is built by the BuildTree method.

Classification 1
Name
- SAPData class [Seite 1918]
- SAPforms Data Plug-In for Workflow [Seite 1897]

Properties
Returns the name of the workflow to whose task the container description belongs.

Classification 2
Name
- SAPData class [Seite 1975]
- SAPforms Data Plug-In for Function Modules [Seite 1954]

Properties
Name of the R/3 function module that is to be executed.

Classification 3
Name
- SAPData class [Seite 2027]
- SAPforms Data Plug-In for IDoc [Seite 2006]

Properties
Name of the IDoc as it appears in the control record.

Classification 4
Name
- SAPData class [Seite 2078]
- SAPforms Data Plug-In for Demo [Seite 2077]

Classification 5
Name
- Field Class [Seite 2007]
- SAPforms Data Plug-In for Workflow [Seite 1897]
- SAPforms Data Plug-In for Function Modules [Seite 1954]
- SAPforms Data Plug-In for IDoc [Seite 2006]
**Properties**
Name of the field.

**Classification 6**

**Name**

- Segment class [Seite 2055]
- SAPforms Data Plug-In for IDoc [Seite 2006]

**Properties**
Name of the segment.
Used as a key in the Templates list of the IDoc object. Only the first attribute with this name is added to the list of segments with a key. All other segment attributes with the same name are added without a key.

**Classification 7**

**Name**

- SAPForm class [Seite 1772]
- SAPforms Form Plug-In for Visual Basic [Seite 1756]
- SAPForm Class [Seite 1812]
- SAPforms Form Plug-In for MS Outlook [Seite 1796]
- SAPForm class [Seite 1872]
- SAPforms Form Plug-In for MS Office [Seite 1856]

**Properties**
Original name of the control in the real form (e.g. Visual Basic form).
Save (Method)

Definition
Saves the updated form definition.

Remarks
This method is used by the SAPforms Designer when a form is designed to save the modified form definition after the binding has been defined. The modifications can affect the binding definition or possibly the source code entered in the form.

Classification 1

Properties
Saves the current property values of an SAPData object in a structured string. Counterpart of the Load method.

When a form is created, the SAPforms Designer uses this method to save the structure description retrieved from the R/3 System permanently in the form.

The R/3 structure is restored from the structured string at runtime by the Load method.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Tgt</td>
<td>String in which the status is stored.</td>
</tr>
</tbody>
</table>

Classification 2

Properties
Saves the current property values of an SAPData object in a structured string. Counterpart of the Load method. The string is saved with a form so that the structure information is available when the form is executed.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Tgt</td>
<td>Stores the string structure.</td>
</tr>
</tbody>
</table>
Classification 3

Save

-&gt; Fields Class [Seite 2018]
-&gt; SAPforms Data Plug-In for Workflow [Seite 1897]
-&gt; SAPforms Data Plug-In for Function Modules [Seite 1954]
-&gt; SAPforms Data Plug-In for IDoc [Seite 2006]

Properties

Loads the property values of a fields object from a structured string. As a result, the status of the fields object that was previously saved permanently with the Save method is restored. The string is saved with the form so that the structure information is available when the form is executed.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Tgt</td>
<td>Stores the string structure.</td>
</tr>
</tbody>
</table>

Classification 4

Save

-&gt; Segment class [Seite 2055]
-&gt; SAPforms Data Plug-In for IDoc [Seite 2006]

Properties

This method saves the structure of a segment, including the field list, in a structured string.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Tgt</td>
<td>Stores the string structure.</td>
</tr>
</tbody>
</table>

Classification 5

Save

-&gt; Segments class [Seite 2067]
-&gt; SAPforms Data Plug-In for IDoc [Seite 2006]

Properties

This method saves the segment list in a structured string. Counterpart of the Load method.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Tgt</td>
<td>Stores the string structure.</td>
</tr>
</tbody>
</table>

Classification 6

Save

-&gt; Links class [Seite 1862]
-&gt; SAPforms Form Plug-In for Visual Basic [Seite 1756]
-&gt; Links class [Seite 1862]
-&gt; SAPforms Form Plug-In for MS Office [Seite 1856]
-&gt; Links class [Seite 1862]
-&gt; SAPforms Form Plug-In for MS Outlook [Seite 1796]
Properties
Method that writes the content of all links to a file. Counterpart of the Load method.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Tgt</td>
<td>Stores the string structure.</td>
</tr>
</tbody>
</table>

Classification 7
Save

- SAPForm class [Seite 1772]
  - SAPforms Form Plug-In for Visual Basic [Seite 1756]
  - SAPForm class [Seite 1872]
    - SAPforms Form Plug-In for MS Office [Seite 1856]
  - SAPForm class [Seite 1812]
    - SAPforms Form Plug-In for MS Outlook [Seite 1796]

Properties
Method that saves a form to a file if it was created from a file.
Reads the old file and writes updated lines to the new file.

Classification 8
Save

- SAPXmit class [Seite 2106]
  - SAPforms Data Plug-In for Demo [Seite 2077]
- SAPXmit class [Seite 1837]
  - SAPforms Form Plug-In for Outlook [Seite 1796]
  - SAPXmit class [Seite 2126]
  - SAPforms Transmit Plug-In for RFC [Seite 2125]
- SAPXmit class [Seite 2168]
  - SAPforms Transmit Plug-In for MAPI [Seite 2146]
- SAPXmit class [Seite 2147]
  - SAPforms Transmit Plug-In for SAPMAPI [Seite 2167]
- SAPXmit class [Seite 2189]
  - SAPforms Transmit Plug-In for WinSock [Seite 2188]
- SAPXmit class [Seite 2209]
  - SAPforms Transmit Plug-In for Demo [Seite 2208]

Properties
Saves the current property values of an SAPXmit object in a structured string. Counterpart of the Load method.
When a form is created, the SAPforms Designer uses this method to save the structure description retrieved from the R/3 System permanently in the form.
The R/3 structure is restored from the structured string at runtime by the Load method.
## Save (Method)

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Tgt</td>
<td>String in which the status is stored.</td>
</tr>
</tbody>
</table>
Segments Class

Definition
List class of the IDoc segments within an IDoc.

Remarks
This class contains a list of the segments in an IDoc. The individual segments in the IDoc can be accessed with the Item property or with a For...Each loop.

Classification
SAPforms Data Plug-In for IDoc [Seite 2006]

→Segments Class

→AddWithTemplate (Method) [Seite 2068]
→Load (Method) [Seite 2213]
→Save (Method) [Seite 2219]
AddWithTemplate (Method)

Definition
Generates a new segment object from the instance data of a segment and adds this segment to the segment list.

Remarks
The properties of the new object are copied from the corresponding segment in the Templates list.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Data</td>
<td>Character string containing segment data. The segment type is taken from this character string.</td>
</tr>
<tr>
<td>Templates</td>
<td>Segment list in which the system searches for the corresponding segment.</td>
</tr>
<tr>
<td>After</td>
<td>Insert the new segment after this segment. Can be an object or a character string. If it is a character string, it specifies the last of these objects in the current list. It must be a higher-level or same-level segment of the new segment. If it is a higher-level segment, it is added to the current list as the last subordinate segment of this segment.</td>
</tr>
</tbody>
</table>

Classification
AddWithTemplate

Segments class [Seite 2067]
SAPforms Data Plug-In for IDoc [Seite 2006]
Load (Method)

Definition
Loads structure information to SAPforms objects.

Remarks

Classification 1
Load

Properties
Loads the property values of an SAPData object from a structured string. As a result, the status of the SAPData object that was previously saved permanently with the Save method is restored. This saves the additional overheads that would result if the structure information had to be loaded each time from the original source. This method is usually called within the Load method of the SAPForm class.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>From</td>
<td>Character string containing the structure information</td>
</tr>
</tbody>
</table>

Classification 2
Load

Properties
Loads the property values of a field object from a structured string. As a result, the status of the field object that was previously saved permanently with the Save method is restored. For this method to be called successfully, the next character at the Pos position must be keyword “Fld”.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>From</td>
<td>Structured string</td>
</tr>
<tr>
<td>Pos</td>
<td>Current position within the string. Should point to the number of remaining properties to read</td>
</tr>
</tbody>
</table>

Classification 3
Load
Properties

Loads the property values of a fields object from a structured string. As a result, the status of the fields object that was previously saved permanently with the Save method is restored.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>From</td>
<td>Character string that provides the structure information.</td>
</tr>
<tr>
<td>Pos</td>
<td>Current search position within the character string.</td>
</tr>
</tbody>
</table>

Classification 4

Load

Properties

Reconstructs the segment object from a structured string. Counterpart of the Save method.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>From</td>
<td>Character string containing structure information.</td>
</tr>
<tr>
<td>Pos</td>
<td>Current search position within the character string.</td>
</tr>
</tbody>
</table>

Classification 5

Load

Properties

This method reconstructs the segment list from a structured string. Counterpart of the Save method.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>From</td>
<td>Character string containing structure information.</td>
</tr>
<tr>
<td>Pos</td>
<td>Current search position within the character string.</td>
</tr>
</tbody>
</table>

Classification 6

Load

Properties

This method reconstructs an object of the Link class from the serialized representation. It assumes that the next token is located at the Pos position in the serialized representation of the “Fld” keyword.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>From</td>
<td>Serialization string.</td>
</tr>
</tbody>
</table>
Load (Method)

Classification 7

Load

←SAPForm class [Seite 1772]
←SAPforms Form Plug-In for Visual Basic [Seite 1756]
←SAPForm class [Seite 1812]
←SAPforms Form Plug-In for MS Outlook [Seite 1796]
←SAPForm class [Seite 1872]
←SAPforms Form Plug-In for MS Office [Seite 1856]

Properties

Method for loading form controls into Controls Collection from a form or form template. Note that the form information is not stored in the Tag property of this object, but reconstructed each time the form is loaded. This method is called at design time and at runtime. It has to populate the Controls collection for both cases. At runtime, the method is called by the user code in the form's Load method. When the form is designed, the method is called by the SAPForms Designer or from outside SAPForms. If you are using the SAPForms Designer as a stand-alone application, the Load method is called with Mode=2. This instructs the Load method to load the form from a file. If you are using the SAPForms Designer as a plug-in, the Load method is invoked from the hosting environment and receives the proprietary form description as a parameter. The populated SAPForm object should be passed to the SAPForms Designer by the PlugIn method of the Designer class.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Frm</td>
<td>form or form template containing controls.</td>
</tr>
<tr>
<td>Mode</td>
<td>0=design time (add-in), 1=design time (file), 2=runtime.</td>
</tr>
</tbody>
</table>

Classification 8

Load

←SAPXmit class [Seite 2106]
←SAPforms Data Plug-In for Demo [Seite 2077]
←SAPXmit class [Seite 1837]
←SAPforms Form Plug-In for Outlook [Seite 1796]
←SAPXmit class [Seite 2126]
←SAPforms Transmit Plug-In for RFC [Seite 2125]
←SAPXmit class [Seite 2168]
←SAPforms Transmit Plug-In for MAPI [Seite 2146]
←SAPXmit class [Seite 2147]
←SAPforms Transmit Plug-In for SAPMAPI [Seite 2167]
←SAPXmit class [Seite 2189]
←SAPforms Transmit Plug-In for WinSock [Seite 2188]
←SAPXmit class [Seite 2209]
←SAPforms Transmit Plug-In for Demo [Seite 2208]
Properties
Loads the property values of an SAPXmit object from a structured string. As a result, the status of the SAPXmit object that was previously saved permanently with the Save method is restored.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>From</td>
<td>Character string containing the structure information</td>
</tr>
</tbody>
</table>
Save (Method)

Definition
Saves the updated form definition.

Remarks
This method is used by the SAPforms Designer when a form is designed to save the modified form definition after the binding has been defined. The modifications can affect the binding definition or possibly the source code entered in the form.

Classification 1
Save

Properties
Saves the current property values of an SAPData object in a structured string. Counterpart of the Load method.
When a form is created, the SAPforms Designer uses this method to save the structure description retrieved from the R/3 System permanently in the form.
The R/3 structure is restored from the structured string at runtime by the Load method.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Tgt</td>
<td>String in which the status is stored.</td>
</tr>
</tbody>
</table>

Classification 2
Save

Properties
Saves the current property values of an SAPData object in a structured string. Counterpart of the Load method. The string is saved with a form so that the structure information is available when the form is executed.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Tgt</td>
<td>Stores the string structure.</td>
</tr>
</tbody>
</table>
Save (Method)

**Classification 3**

**Save**

- **Fields Class [Seite 2018]**
- **SAPforms Data Plug-In for Workflow [Seite 1897]**
- **SAPforms Data Plug-In for Function Modules [Seite 1954]**
- **SAPforms Data Plug-In for IDoc [Seite 2006]**

**Properties**

 Loads the property values of a fields object from a structured string. As a result, the status of the fields object that was previously saved permanently with the Save method is restored.

The string is saved with the form so that the structure information is available when the form is executed.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Tgt</td>
<td>Stores the string structure.</td>
</tr>
</tbody>
</table>

**Classification 4**

**Save**

- **Segment class [Seite 2055]**
- **SAPforms Data Plug-In for IDoc [Seite 2006]**

**Properties**

This method saves the structure of a segment, including the field list, in a structured string.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Tgt</td>
<td>Stores the string structure.</td>
</tr>
</tbody>
</table>

**Classification 5**

**Save**

- **Segments class [Seite 2067]**
- **SAPforms Data Plug-In for IDoc [Seite 2006]**

**Properties**

This method saves the segment list in a structured string. Counterpart of the Load method.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Tgt</td>
<td>Stores the string structure.</td>
</tr>
</tbody>
</table>

**Classification 6**

**Save**

- **Links class [Seite 1862]**
- **SAPforms Form Plug-In for Visual Basic [Seite 1756]**
- **Links class [Seite 1862]**
- **SAPforms Form Plug-In for MS Office [Seite 1856]**
- **Links class [Seite 1862]**
- **SAPforms Form Plug-In for MS Outlook [Seite 1796]**
Properties

Method that writes the content of all links to a file. Counterpart of the Load method.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Tgt</td>
<td>Stores the string structure.</td>
</tr>
</tbody>
</table>

Classification 7

Save

- SAPForm class [Seite 1772]
- SAPforms Form Plug-In for Visual Basic [Seite 1756]
- SAPForm class [Seite 1872]
- SAPforms Form Plug-In for MS Office [Seite 1856]
- SAPForm class [Seite 1812]
- SAPforms Form Plug-In for MS Outlook [Seite 1796]

Properties

Method that saves a form to a file if it was created from a file. Reads the old file and writes updated lines to the new file.

Classification 8

Save

- SAPXmit class [Seite 2106]
- SAPforms Data Plug-In for Demo [Seite 2077]
- SAPXmit class [Seite 1837]
- SAPforms Form Plug-In for Outlook [Seite 1796]
- SAPXmit class [Seite 2126]
- SAPforms Transmit Plug-In for RFC [Seite 2125]
- SAPXmit class [Seite 2168]
- SAPforms Transmit Plug-In for MAPI [Seite 2146]
- SAPXmit class [Seite 2147]
- SAPforms Transmit Plug-In for SAPMAPI [Seite 2167]
- SAPXmit class [Seite 2189]
- SAPforms Transmit Plug-In for WinSock [Seite 2188]
- SAPXmit class [Seite 2209]
- SAPforms Transmit Plug-In for Demo [Seite 2208]

Properties

Saves the current property values of an SAPXmit object in a structured string. Counterpart of the Load method.

When a form is created, the SAPforms Designer uses this method to save the structure description retrieved from the R/3 System permanently in the form.

The R/3 structure is restored from the structured string at runtime by the Load method.
<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Tgt</td>
<td>String in which the status is stored.</td>
</tr>
</tbody>
</table>
SAPforms Data Plug-In for Demo

Definition
SAPData type plug-in for demonstration purposes.

Use
This plug-in is used in the introductory example (Getting Started [Seite 1739]). It simulates the start of a workflow without a connection to an R/3 System. This plug-in must be used with the SAPXmit plug-in for demonstration purposes. This plug-in implements the SAPData interface that provides the SAPforms Designer [Seite 1735] and other SAPforms components with access to certain types of R/3 data structures and interfaces. In other words, this plug-in can be regarded as being synonymous with the associated SAPData class.

Classification
SAPforms Data Plug-In for Demo
- SAPData Class [Seite 2078]
- SAPXmit class [Seite 2106]
**SAPData Class**

**Definition**
Original class of this plug-in.

**Remarks**
SAPData plug-in type classes implement the SAPData interface that provides the SAPforms Designer and other SAPforms components with access to certain types of R/3 data structures and interfaces. In other words, these classes can be regarded as being synonymous with various plug-ins of this type.

A SAPData object is usually also instantiated with the SAPForm object. The SAPData object is then referenced by the SAPData property of the SAPForm object. An object of the SAPData class, however, cannot exist without an object of the SAPForms class.

If you want to develop your own SAPData-type plug-ins, you should use the source code supplied for this plug-in as a reference (template). The template is stored in the directory `\$\apforms\sapddemo`.

**Classification**

SAPforms Data Plug-In for Demo [Seite 2077]

→ **SAPData Class**

→ Execute (Method) [Seite 2210]
→ Load (Method) [Seite 2213]
→ LoadInstance (Method) [Seite 2086]
→ Name (Property) [Seite 2088]
→ Options (Method) [Seite 2217]
→ Receiver (Property) [Seite 2092]
→ SAPXmit (Property) [Seite 2093]
→ Save (Method) [Seite 2219]
→ SaveInstance (Method) [Seite 2098]
→ Start (Method) [Seite 2223]
→ UpdateInstance (Method) [Seite 2102]
→ Value (Property) [Seite 2103]
→ Version (Property) [Seite 2105]
Execute (Method)

Definition
Method that calls the Execute function in the R/3 System. The semantics of this function depend on the SAPData object.

Remarks
The method can be called directly via the SAPData object or indirectly via the Start method of the SAPForm object, whose SAPData property is a reference to the SAPData object.

Classification 1
Execute

Properties
Terminates a work item in the R/3 System and supplies its container elements with the data values in the fields list.

Classification 2
Execute

Properties
Does not execute an action in the R/3 System but rather displays the data of the object in a separate window.

Classification 3
Execute
Properties
Invokes the Execute method in the associated Xmit or SAPData object. The semantics of this method depend on this object.

Classification 4
 Execute

Properties
This method executes a work item synchronously via a remote function call (RFC). Supports SAPData object of type Container.
This method causes a certain function module to be executed in the R/3 System. This module receives a table with a complete R3F structure containing the work item information and the container data that are to be supplied from the form.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Data</td>
<td>Object representing the container data.</td>
</tr>
</tbody>
</table>

Classification 5
 Execute

Properties
Sends a mail to the R/3 System that contains the information required to terminate a work item (including container data) as an attachment.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Data</td>
<td>Object representing the container data.</td>
</tr>
</tbody>
</table>

Classification 6
 Execute

Properties
Overwrites the R3F attachment, thereby updating it.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Pdata</td>
<td>Object that represents the mobiData container data</td>
</tr>
</tbody>
</table>

Classification 7
 Execute

Properties
Properties
Displays the instance data of the associated SAPData object in a window instead of sending it to the R/3 System.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Data</td>
<td>object of class SAPData</td>
</tr>
</tbody>
</table>
Load (Method)

Definition
Loads structure information to SAPforms objects.

Remarks

Classification 1

Load

Properties
Loads the property values of an SAPData object from a structured string. As a result, the status of the SAPData object that was previously saved permanently with the Save method is restored.
This saves the additional overheads that would result if the structure information had to be loaded each time from the original source. This method is usually called within the Load method of the SAPForm class.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>From</td>
<td>Character string containing the structure information</td>
</tr>
</tbody>
</table>

Classification 2

Load

Properties
Loads the property values of a field object from a structured string. As a result, the status of the field object that was previously saved permanently with the Save method is restored.
For this method to be called successfully, the next character at the Pos position must be keyword “Fld”.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>From</td>
<td>Structured string</td>
</tr>
<tr>
<td>Pos</td>
<td>Current position within the string. Should point to the number of remaining properties to read</td>
</tr>
</tbody>
</table>

Classification 3

Load
Properties

Loads the property values of a fields object from a structured string. As a result, the status of the fields object that was previously saved permanently with the Save method is restored.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>From</td>
<td>Character string that provides the structure information.</td>
</tr>
<tr>
<td>Pos</td>
<td>Current search position within the character string.</td>
</tr>
</tbody>
</table>

Classification 4

Load

Properties

Reconstructs the segment object from a structured string. Counterpart of the Save method.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>From</td>
<td>Character string containing structure information.</td>
</tr>
<tr>
<td>Pos</td>
<td>Current search position within the character string.</td>
</tr>
</tbody>
</table>

Classification 5

Load

Properties

This method reconstructs the segment list from a structured string. Counterpart of the Save method.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>From</td>
<td>Character string containing structure information.</td>
</tr>
<tr>
<td>Pos</td>
<td>Current search position within the character string.</td>
</tr>
</tbody>
</table>

Classification 6

Load

Properties

This method reconstructs an object of the Link class from the serialized representation. It assumes that the next token is located at the Pos position in the serialized representation of the “Fld” keyword.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>From</td>
<td>Serialization string.</td>
</tr>
</tbody>
</table>
Load (Method)

Current position within the character string.

Classification 7

Load

\(<\text{SAPForm class [Seite 1772]}\>
\(<\text{SAPforms Form Plug-In for Visual Basic [Seite 1756]}\>
\(<\text{SAPForm class [Seite 1812]}\>
\(<\text{SAPforms Form Plug-In for MS Outlook [Seite 1796]}\>
\(<\text{SAPForm class [Seite 1872]}\>
\(<\text{SAPforms Form Plug-In for MS Office [Seite 1856]}\>

Properties

Method for loading form controls into \textit{Controls} Collection from a form or form template. Note that the form information is not stored in the \textit{Tag} property of this object, but reconstructed each time the form is loaded. This method is called at design time and at runtime. It has to populate the Controls collection for both cases. At runtime, the method is called by the user code in the form's Load method. When the form is designed, the method is called by the SAPForms Designer or from outside SAPForms. If you are using the SAPForms Designer as a stand-alone application, the Load method is called with \textit{Mode}=2. This instructs the Load method to load the form from a file. If you are using the SAPForms Designer as a plug-in, the Load method is invoked from the hosting environment and receives the proprietary form description as a parameter. The populated SAPForm object should be passed to the SAPForms Designer by the \textit{PlugIn} method of the \textit{Designer} class.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Frm</td>
<td>form or form template containing controls.</td>
</tr>
<tr>
<td>Mode</td>
<td>0=design time (add-in), 1=design time (file), 2=runtime.</td>
</tr>
</tbody>
</table>

Classification 8

Load

\(<\text{SAPXmit class [Seite 2106]}\>
\(<\text{SAPforms Data Plug-In for Demo [Seite 2077]}\>
\(<\text{SAPXmit class [Seite 1837]}\>
\(<\text{SAPforms Form Plug-In for Outlook [Seite 1796]}\>
\(<\text{SAPXmit class [Seite 2126]}\>
\(<\text{SAPforms Transmit Plug-In for RFC [Seite 2125]}\>
\(<\text{SAPXmit class [Seite 2168]}\>
\(<\text{SAPforms Transmit Plug-In for MAPI [Seite 2146]}\>
\(<\text{SAPXmit class [Seite 2147]}\>
\(<\text{SAPforms Transmit Plug-In for SAPMAPI [Seite 2167]}\>
\(<\text{SAPXmit class [Seite 2189]}\>
\(<\text{SAPforms Transmit Plug-In for WinSock [Seite 2188]}\>
\(<\text{SAPXmit class [Seite 2209]}\>
\(<\text{SAPforms Transmit Plug-In for Demo [Seite 2208]}\>
Properties
Loads the property values of an SAPXmit object from a structured string. As a result, the status of the SAPXmit object that was previously saved permanently with the Save method is restored.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>From</td>
<td>Character string containing the structure information</td>
</tr>
</tbody>
</table>
LoadInstance (Method)

Definition
Method that loads the data content of a SAPData object from a file.

Remarks
The method is usually called when data flows from the R/3 System to the form and is to be displayed by the form. This is the case, for example, when a work item is executed. The form is then started by the [SAPforms Helper][Seite 1741] and the R3F file is forwarded as a parameter. The existence of a parameter when the form is started ensures that the LoadInstance method is called in the Init() method of the SAPform object. The GetData method is usually called after the LoadInstance method.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Path</td>
<td>Complete path designation</td>
</tr>
</tbody>
</table>

Classification 1

LoadInstance

[SAPData Class][Seite 1918]
[SAPforms Data Plug-In for Workflow][Seite 1897]

Properties
Loads the R/3 workflow information from an R3F file to the SAPData object. The current container element values can be interrogated and manipulated via the SAPData.Value and SAPData.Fields(s).Data properties in particular.

Classification 2

LoadInstance

[SAPData Class][Seite 1975]
[SAPforms Data Plug-In for Function Modules][Seite 1954]

Properties
Implementation of the SAPData interface for R/3 function modules (BAPIs).

Classification 3

LoadInstance

[SAPData Class][Seite 2027]
[SAPforms Data Plug-In for IDoc][Seite 2006]

Properties
Loads an IDoc instance from a file.

Builds the segments list from the records in the file. This method will only function correctly if the Templates list containing the entire IDoc meta structure exists. The structure of each segment in the Segments list is derived from the corresponding segment in the Templates list.

Classification 4

LoadInstance
Properties
The method saves the contents of a field in the FieldValues array. This method requires that the elements be saved in a fixed order. Even though the field name is contained in the file, it is not evaluated.
**Name (Property)**

**Definition**
Name of the object, field, segment, or control.

**Remarks**
This name, for example, must be used as the root in the SAP structure tree that is built by the BuildTree method.

**Classification 1**

**Name**

- SAPData class [Seite 1918]
- SAPforms Data Plug-In for Workflow [Seite 1897]

**Properties**
Returns the name of the workflow to whose task the container description belongs.

**Classification 2**

**Name**

- SAPData class [Seite 1975]
- SAPforms Data Plug-In for Function Modules [Seite 1954]

**Properties**
Name of the R/3 function module that is to be executed.

**Classification 3**

**Name**

- SAPData class [Seite 2027]
- SAPforms Data Plug-In for IDoc [Seite 2006]

**Properties**
Name of the IDoc as it appears in the control record.

**Classification 4**

**Name**

- SAPData class [Seite 2078]
- SAPforms Data Plug-In for Demo [Seite 2077]

**Classification 5**

**Name**

- Field Class [Seite 2007]
- SAPforms Data Plug-In for Workflow [Seite 1897]
- SAPforms Data Plug-In for Function Modules [Seite 1954]
- SAPforms Data Plug-In for IDoc [Seite 2006]
Properties
Name of the field.

Classification 6
Name

- Segment class [Seite 2055]
- SAPforms Data Plug-In for IDoc [Seite 2006]

Properties
Name of the segment.
Used as a key in the Templates list of the IDoc object. Only the first attribute with this name is added to the list of segments with a key. All other segment attributes with the same name are added without a key.

Classification 7
Name

- SAPForm class [Seite 1772]
- SAPforms Form Plug-In for Visual Basic [Seite 1756]
- SAPForm Class [Seite 1812]
- SAPforms Form Plug-In for MS Outlook [Seite 1796]
- SAPForm class [Seite 1872]
- SAPforms Form Plug-In for MS Office [Seite 1856]

Properties
Original name of the control in the real form (e.g. Visual Basic form).
Options (Method)

Definition
Method that outputs the object-specific options dialog box.

Remarks
This dialog box displays either global settings or settings that are valid for the current form. These settings can be changed here. The global settings are read from the registry before they are displayed and are then saved again to the registry.

The method is called within the SAPforms Designer if the required plug-in is active under Properties on the Plug-Ins tab in the Options dialog. You can also use the sample program Options.exe to display the Options dialog of a certain plug-in.

Classification 1
Options
- SAPData class [Seite 1918]
- SAPforms Data Plug-In for Workflow [Seite 1897]
- SAPData class [Seite 1975]
- SAPforms Data Plug-In for Function Modules [Seite 1954]
- SAPData class [Seite 2027]
- SAPforms Data Plug-In for IDoc [Seite 2006]
- SAPData class [Seite 2078]
- SAPforms Data Plug-In for Demo [Seite 2077]

Classification 2
Options
- SAPForm class [Seite 1772]
- SAPforms Form Plug-In for Visual Basic [Seite 1756]
- SAPForm class [Seite 1812]
- SAPforms Form Plug-In for MS Outlook [Seite 1796]
- SAPForm class [Seite 1872]
- SAPforms Form Plug-In for MS Office [Seite 1856]

Classification 3
Options
- SAPXmit class [Seite 2106]
- SAPforms Data Plug-In for Demo [Seite 2077]
- SAPXmit class [Seite 1837]
- SAPforms Form Plug-In for MS Outlook [Seite 1796]
- SAPXmit class [Seite 2126]
Options (Method)

- SAPforms Transmit Plug-In for RFC [Seite 2125]
- SAPXmit class [Seite 2168]
- SAPforms Transmit Plug-In for MAPI [Seite 2146]
- SAPXmit class [Seite 2147]
- SAPforms Transmit Plug-In for SAPMAPI [Seite 2167]
- SAPXmit class [Seite 2189]
- SAPforms Transmit Plug-In for SMTP [Seite 2188]
- SAPXmit class [Seite 2209]
- SAPforms Transmit Plug-In for Demo [Seite 2208]
Receiver (Property)

Definition
Recipient of the message from the perspective of the information creator.

Remarks

Classification
Receiver

Properties
This property corresponds to the property with the same name in an R3F structure. When an R3F structure from the R/3 System is imported in the form, this property receives the mail address of the agent. This property is changed in the SaveInstance method and then receives the mail address of the receiver in the R/3 System.
SAPXmit (Property)

Definition
Reference to the SAPXmit object that is used to encapsulate the transport of SAPData to the R/3 System.

Remarks
Classification
SAPXmit

- SAPData class [Seite 1918]
- SAPforms Data Plug-In for Workflow [Seite 1897]
- SAPData class [Seite 1975]
- SAPforms Data Plug-In for Function Modules [Seite 1954]
- SAPData class [Seite 2027]
- SAPforms Data Plug-In for IDoc [Seite 2006]
- SAPData class [Seite 2078]
- SAPforms Data Plug-In for Demo [Seite 2077]
- SAPForm class [Seite 1772]
- SAPforms Form Plug-In for Visual Basic [Seite 1756]
- SAPForm class [Seite 1812]
- SAPforms Form Plug-In for MS Outlook [Seite 1796]
- SAPForm class [Seite 1872]
- SAPforms Form Plug-In for MS Office [Seite 1856]
Save (Method)

Definition
Saves the updated form definition.

Remarks
This method is used by the SAPforms Designer when a form is designed to save the modified form definition after the binding has been defined. The modifications can affect the binding definition or possibly the source code entered in the form.

Classification 1
Save

Properties
Saves the current property values of an SAPData object in a structured string. Counterpart of the Load method.
When a form is created, the SAPforms Designer uses this method to save the structure description retrieved from the R/3 System permanently in the form.
The R/3 structure is restored from the structured string at runtime by the Load method.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Tgt</td>
<td>String in which the status is stored.</td>
</tr>
</tbody>
</table>

Classification 2
Save

Properties
Saves the current property values of an SAPData object in a structured string. Counterpart of the Load method. The string is saved with a form so that the structure information is available when the form is executed.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Tgt</td>
<td>Stores the string structure.</td>
</tr>
</tbody>
</table>
Classification 3

Save

←Fields Class [Seite 2018]
←SAPforms Data Plug-In for Workflow [Seite 1897]
←SAPforms Data Plug-In for Function Modules [Seite 1954]
←SAPforms Data Plug-In for IDoc [Seite 2006]

Properties

Loads the property values of a fields object from a structured string. As a result, the status of the fields object that was previously saved permanently with the Save method is restored.
The string is saved with the form so that the structure information is available when the form is executed.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Tgt</td>
<td>Stores the string structure.</td>
</tr>
</tbody>
</table>

Classification 4

Save

←Segment class [Seite 2055]
←SAPforms Data Plug-In for IDoc [Seite 2006]

Properties

This method saves the structure of a segment, including the field list, in a structured string.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Tgt</td>
<td>Stores the string structure.</td>
</tr>
</tbody>
</table>

Classification 5

Save

←Segments class [Seite 2067]
←SAPforms Data Plug-In for IDoc [Seite 2006]

Properties

This method saves the segment list in a structured string. Counterpart of the Load method.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Tgt</td>
<td>Stores the string structure.</td>
</tr>
</tbody>
</table>

Classification 6

Save

←Links class [Seite 1862]
←SAPforms Form Plug-In for Visual Basic [Seite 1756]
←Links class [Seite 1862]
←SAPforms Form Plug-In for MS Office [Seite 1856]
←Links class [Seite 1862]
←SAPforms Form Plug-In for MS Outlook [Seite 1796]
Properties
Method that writes the content of all links to a file. Counterpart of the Load method.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Tgt</td>
<td>Stores the string structure.</td>
</tr>
</tbody>
</table>

Classification 7

Save

- SAPform class [Seite 1772]
- SAPforms Form Plug-In for Visual Basic [Seite 1756]
- SAPForm class [Seite 1872]
- SAPforms Form Plug-In for MS Office [Seite 1856]
- SAPForm class [Seite 1812]
- SAPforms Form Plug-In for MS Outlook [Seite 1796]

Properties
Method that saves a form to a file if it was created from a file. Reads the old file and writes updated lines to the new file.

Classification 8

Save

- SAPXmit class [Seite 2106]
- SAPforms Data Plug-In for Demo [Seite 2077]
- SAPXmit class [Seite 1837]
- SAPforms Form Plug-In for Outlook [Seite 1796]
- SAPXmit class [Seite 2126]
- SAPforms Transmit Plug-In for RFC [Seite 2125]
- SAPXmit class [Seite 2168]
- SAPforms Transmit Plug-In for MAPI [Seite 2146]
- SAPXmit class [Seite 2147]
- SAPforms Transmit Plug-In for SAPMAPI [Seite 2167]
- SAPXmit class [Seite 2189]
- SAPforms Transmit Plug-In for WinSock [Seite 2188]
- SAPXmit class [Seite 2209]
- SAPforms Transmit Plug-In for Demo [Seite 2208]

Properties
Saves the current property values of an SAPXmit object in a structured string. Counterpart of the Load method.
When a form is created, the SAPforms Designer uses this method to save the structure description retrieved from the R/3 System permanently in the form. The R/3 structure is restored from the structured string at runtime by the Load method.
### Save (Method)

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Tgt</td>
<td>String in which the status is stored.</td>
</tr>
</tbody>
</table>
SaveInstance (Method)

Definition
This method saves the current data content of the SAPData object to the specified file.

Remarks
This data content of the SAPData object is the result of the PutData method of the SAPForm object or the result of assignments to the SAPData method in the SAPData object. The method is usually called from the SAPXmit object to save the data content in a file before it is sent by email, for example.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Path</td>
<td>Complete path specifying where the data content is to be saved.</td>
</tr>
</tbody>
</table>

Classification 1
SaveInstance

Properties
Method that saves the current contents of the function module to the specified file in R3F format.

Classification 2
SaveInstance

Properties
Saves the current IDoc instance segment-by-segment to the specified file. Uses the Record property of the segment object to generate a record for a segment.

Classification 3
SaveInstance

Properties
Generates a file containing a separate line for each data field.
**Start (Method)**

**Definition**
Method that executes the *Start* function in R/3. The semantics of this function depend on the concrete SAPData object.

**Remarks**
In the case of the SAPData object for workflow containers, this method starts a workflow and populates its container elements with values from the form fields. This method can be called directly via the SAPData object or indirectly via the *Start* method of the SAPForm object, whose SAPData property is a reference to the SAPData object. If an SAPXmit object is assigned to the SAPData object, the *Start* method of the SAPXmit object is called.

**Classification 1**
**Start**

- [SAPData class][1918]
- [SAPforms Data Plug-In for Workflow][1897]

**Properties**
Starts a workflow in the R/3 System and supplies its container elements with the data values in the fields list.

**Classification 2**
**Start**

- [SAPData class][1975]
- [SAPforms Data Plug-In for Function Modules][1954]

**Properties**
This method carries out the current R/3 function module call.

**Classification 3**
**Start**

- [SAPData class][2027]
- [SAPforms Data Plug-In for IDoc][2006]

**Properties**
Generates an IDoc file from the data in the segments list and transfers this IDoc to the R/3 System. The transfer method depends on the assigned SAPXmit object. A workflow can be started in the R/3 System when the inbound IDoc is processed.

**Classification 4**
**Start**

- [SAPData class][2078]
- [SAPforms Data Plug-In for Demo][2077]
Properties
Does not execute an action in the R/3 System but rather displays the data of the object in a separate window.

Classification 5
Start

Properties
Invokes the Start method in the associated SAPXmit or SAPData object. The semantics of this method depend on this SAPData object. If this object is the SAPData object for SAP Business Workflow, this method starts a workflow in an R/3 System.

Classification 6
Start

Properties
This method starts a workflow synchronously via RFC. Supports SAPData objects of types "Idoc" and "Container".

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Data</td>
<td>Object of type SAPData.</td>
</tr>
</tbody>
</table>

Classification 7
Start

Properties
Sends a mail to the R/3 System with an attachment containing the information required to start a workflow.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Data</td>
<td>Object of type SAPData.</td>
</tr>
</tbody>
</table>
Classification 8

Start

- SAPXmit class [Seite 2209]
- SAPforms Transmit Plug-In for Demo [Seite 2208]
- SAPXmit class [Seite 2106]
- SAPforms Data Plug-In for Demo [Seite 2077]

Properties
Displays the instance data of the associated SAPData object in a window instead of sending it to the R/3 System.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Data</td>
<td>Object of type SAPData.</td>
</tr>
</tbody>
</table>
UpdateInstance (Method)

Definition
Updates the fields in the R3F file.

Remarks

Classification 1
UpdateInstance

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Path</td>
<td>Fully qualified target path of storage procedure</td>
</tr>
</tbody>
</table>

Properties
Upates the R3F file that was read by the last LoadInstance call. Unlike the SaveInstance method, this method is designed to update R3F attachments in incoming messages.

Classification 2
UpdateInstance

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Path</td>
<td>Fully qualified target path of storage procedure</td>
</tr>
</tbody>
</table>

Properties
Saves the modified form data to the R3F attachment in a mail.
**Value (Property)**

**Definition**
Value of a specific field type.

**Remarks**

**Classification 1**

**Value**

- [SAPData class](#)
- [SAPforms Data Plug-In for Workflow](#)
- [SAPData class](#)
- [SAPforms Data Plug-In for Function Modules](#)
- [SAPData class](#)
- [SAPforms Data Plug-In for IDoc](#)
- [SAPData class](#)
- [SAPforms Data Plug-In for Demo](#)

**Properties**
The value of an SAP data field, defined in greater detail by the *Qual* parameter. The structure and syntax of the *Qual* parameter depends on the concrete SAP data structure from which this field originates. In the simplest case, the *Qual* parameter designates the name of a field whose value is to be set or read. If the field is part of a structure (for example, an IDoc segment), *Qual* must identify the field uniquely within the overall structure. If the structure is repeated within an instance - as is the case with some IDocs - *Qual* must also contain a selector that identifies the IDoc segment uniquely.

```
obj.Value("Field1") = "abc".
Obj.Value("Seg1.Field1") = "abc".
X = obj.Value("Seg1.Seg2.Field1").
X = obj.Value("Seg1.Field1 Seg2.Field2 = '123'") ' selector.
```

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Qual</td>
<td>Qualifier and selector that uniquely identifies a particular field.</td>
</tr>
</tbody>
</table>

**Classification 2**

**Value**

- [SAPCtrl class](#)
- [SAPforms Form Plug-In for Visual Basic](#)
- [SAPCtrl class](#)
- [SAPforms Form Plug-In for MS Outlook](#)
- [SAPCtrl class](#)
- [SAPforms Form Plug-In for MS Office](#)
Properties

Returns the value of control.
This property is an abstraction of the different properties used by the original controls to present their content, for example, as Text, Caption, Value, and so on. The implementation of the SAP Ctrl must map the original control property in the Value property for each control that is to be supported by SAP forms.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>X</td>
<td>Index (e.g. for grids).</td>
</tr>
<tr>
<td>Return</td>
<td>Value</td>
</tr>
</tbody>
</table>
Version (Property)

Definition
Version of the structure information (workflow) interface (function module) IDoc class used for compatibility checks.

Remarks

Classification

Version

← SAPData class [Seite 1918]
← SAPforms Data Plug-In for Workflow [Seite 1897]
← SAPData class [Seite 1975]
← SAPforms Data Plug-In for Function Modules [Seite 1954]
← SAPData class [Seite 2027]
← SAPforms Data Plug-In for IDoc [Seite 2006]
← SAPData class [Seite 2078]
← SAPforms Data Plug-In for Demo [Seite 2077]
SAPXmit Class

Definition
Simulates communication between forms and an R/3 System.

Remarks
This demo implementation of the SAPXmit interface displays the data of the SAPData object.
If you want to develop your own SAPXmit-type plug-ins, you should use the source code supplied for the
SAPforms Transmit Plug-Ins for Demo as a reference (template). The template is stored in the directory
...\sapforms\sapxdemo.

Classification
SAPforms Data Plug-In for Demo [Seite 2077]
→SAPXmit Class
  → Execute (Method) [Seite 2210]
  → Load (Method) [Seite 2213]
  → Options (Method) [Seite 2217]
  → Save (Method) [Seite 2219]
  → Start (Method) [Seite 2223]
Execute (Method)

Definition
Method that calls the Execute function in the R/3 System. The semantics of this function depend on the SAPData object.

Remarks
The method can be called directly via the SAPData object or indirectly via the Start method of the SAPForm object, whose SAPData property is a reference to the SAPData object.

In the case of the SAPData object for workflow containers, for example, this method executes a work item in an SAP Business Workflow and populates its container elements with values from the form fields.

Classification 1
Execute

Properties
Terminates a work item in the R/3 System and supplies its container elements with the data values in the fields list.

Classification 2
Execute

Properties
Does not execute an action in the R/3 System but rather displays the data of the object in a separate window.

Classification 3
Execute
Properties
Invokes the Execute method in the associated Xmit or SAPData object. The semantics of this method depend on this object.

Classification 4
Execute

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Data</td>
<td>Object representing the container data.</td>
</tr>
</tbody>
</table>

Properties
This method executes a work item synchronously via a remote function call (RFC). Supports SAPData object of type Container.
This method causes a certain function module to be executed in the R/3 System. This module receives a table with a complete R3F structure containing the work item information and the container data that are to be supplied from the form.

Classification 5
Execute

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Data</td>
<td>Object representing the container data.</td>
</tr>
</tbody>
</table>

Properties
Sends a mail to the R/3 System that contains the information required to terminate a work item (including container data) as an attachment.

Classification 6
Execute

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Pdata</td>
<td>Object that represents the mobjData container data</td>
</tr>
</tbody>
</table>

Properties
Overwrites the R3F attachment, thereby updating it.

Classification 7
Execute

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Pdata</td>
<td>Object that represents the mobjData container data</td>
</tr>
</tbody>
</table>

Properties
Properties
Displays the instance data of the associated SAPData object in a window instead of sending it to the R/3 System.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Data</td>
<td>object of class SAPData</td>
</tr>
</tbody>
</table>
Load (Method)

Definition
Loads structure information to SAPforms objects.

Remarks

Classification 1
Load

Properties
Loads the property values of a field object from a structured string. As a result, the status of the
field object that was previously saved permanently with the Save method is restored.
For this method to be called successfully, the next character at the Pos position must be keyword “Fld”.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>From</td>
<td>Structured string.</td>
</tr>
<tr>
<td>Pos</td>
<td>Current position within the string. Should point to the number of remaining properties to read.</td>
</tr>
</tbody>
</table>

Classification 2
Load

Properties
Loads the property values of an SAPData object from a structured string. As a result, the status of the
SAPData object that was previously saved permanently with the Save method is restored.
This saves the additional overheads that would result if the structure information had to be loaded each
time from the original source. This method is usually called within the Load method of the SAPForm class.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>From</td>
<td>Character string containing the structure information</td>
</tr>
</tbody>
</table>

Classification 3
Load
Load (Method)

Properties
Loads the property values of a fields object from a structured string. As a result, the status of the fields object that was previously saved permanently with the Save method is restored.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>From</td>
<td>Character string that provides the structure information.</td>
</tr>
<tr>
<td>Pos</td>
<td>Current search position within the character string.</td>
</tr>
</tbody>
</table>

Classification 4
Load

Properties
Reconstructs the segment object from a structured string. Counterpart of the Save method.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>From</td>
<td>Character string containing structure information.</td>
</tr>
<tr>
<td>Pos</td>
<td>Current search position within the character string.</td>
</tr>
</tbody>
</table>

Classification 5
Load

Properties
This method reconstructs the segment list from a structured string. Counterpart of the Save method.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>From</td>
<td>Character string containing structure information.</td>
</tr>
<tr>
<td>Pos</td>
<td>Current search position within the character string.</td>
</tr>
</tbody>
</table>

Classification 6
Load

Properties
This method reconstructs an object of the Link class from the serialized representation. It assumes that the next token is located at the Pos position in the serialized representation of the “Fld” keyword.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>From</td>
<td>Serialization string.</td>
</tr>
</tbody>
</table>
Load (Method)

Classification 7

Load

- SAPForm class [Seite 1772]
  - SAPforms Form Plug-In for Visual Basic [Seite 1756]
- SAPForm class [Seite 1812]
  - SAPforms Form Plug-In for MS Outlook [Seite 1796]
- SAPForm class [Seite 1872]
  - SAPforms Form Plug-In for MS Office [Seite 1856]

Properties

Method for loading form controls into Controls Collection from a form or form template. Note that the form information is not stored in the Tag property of this object, but reconstructed each time the form is loaded. This method is called at design time and at runtime. It has to populate the Controls collection for both cases. At runtime, the method is called by the user code in the form's Load method. When the form is designed, the method is called by the SAPForms Designer or from outside SAPForms. If you are using the SAPForms Designer as a stand-alone application, the Load method is called with Mode=2. This instructs the Load method to load the form from a file. If you are using the SAPForms Designer as a plug-in, the Load method is invoked from the hosting environment and receives the proprietary form description as a parameter. The populated SAPForm object should be passed to the SAPForms Designer by the PlugIn method of the Designer class.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Frm</td>
<td>form or form template containing controls.</td>
</tr>
<tr>
<td>Mode</td>
<td>0=design time (add-in), 1=design time (file), 2=runtime.</td>
</tr>
</tbody>
</table>

Classification 8

Load

- SAPXmit class [Seite 2106]
  - SAPforms Data Plug-In for Demo [Seite 2077]
- SAPXmit class [Seite 1837]
  - SAPforms Form Plug-In for Outlook [Seite 1796]
- SAPXmit class [Seite 2126]
  - SAPforms Transmit Plug-In for RFC [Seite 2125]
- SAPXmit class [Seite 2168]
  - SAPforms Transmit Plug-In for MAPI [Seite 2146]
- SAPXmit class [Seite 2147]
  - SAPforms Transmit Plug-In for SAPMAPI [Seite 2167]
- SAPXmit class [Seite 2189]
  - SAPforms Transmit Plug-In for WinSock [Seite 2188]
- SAPXmit class [Seite 2209]
  - SAPforms Transmit Plug-In for Demo [Seite 2208]
Properties
Loads the property values of an SAPXmit object from a structured string. As a result, the status of the SAPXmit object that was previously saved permanently with the Save method is restored.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>From</td>
<td>Character string containing the structure information</td>
</tr>
</tbody>
</table>
Options (Method)

Definition
Method that outputs the object-specific options dialog box.

Remarks
This dialog box displays either global settings or settings that are valid for the current form. These settings can be changed here. The global settings are read from the registry before they are displayed and are then saved again to the registry.
The method is called within the SAPforms Designer if the required plug-in is active under Properties on the Plug-Ins tab in the Options dialog. You can also use the sample program Options.exe to display the Options dialog of a certain plug-in.

Classification 1
Options

- SAPData class [Seite 1918]
- SAPforms Data Plug-In for Workflow [Seite 1897]
- SAPData class [Seite 1975]
- SAPforms Data Plug-In for Function Modules [Seite 1954]
- SAPData class [Seite 2027]
- SAPforms Data Plug-In for IDoc [Seite 2006]
- SAPData class [Seite 2078]
- SAPforms Data Plug-In for Demo [Seite 2077]

Classification 2
Options

- SAPForm class [Seite 1772]
- SAPforms Form Plug-In for Visual Basic [Seite 1756]
- SAPForm class [Seite 1812]
- SAPforms Form Plug-In for MS Outlook [Seite 1796]
- SAPForm class [Seite 1872]
- SAPforms Form Plug-In for MS Office [Seite 1856]

Classification 3
Options

- SAPXmit class [Seite 2106]
- SAPforms Data Plug-In for Demo [Seite 2077]
- SAPXmit class [Seite 1837]
- SAPforms Form Plug-In for MS Outlook [Seite 1796]
- SAPXmit class [Seite 2126]
Options (Method)

- SAPforms Transmit Plug-In for RFC [Seite 2125]
- SAPXmit class [Seite 2168]
- SAPforms Transmit Plug-In for MAPI [Seite 2146]
- SAPXmit class [Seite 2147]
- SAPforms Transmit Plug-In for SAPMAPI [Seite 2167]
- SAPXmit class [Seite 2189]
- SAPforms Transmit Plug-In for SMTP [Seite 2188]
- SAPXmit class [Seite 2209]
- SAPforms Transmit Plug-In for Demo [Seite 2208]
Save (Method)

Definition
Saves the updated form definition.

Remarks
This method is used by the SAPforms Designer when a form is designed to save the modified form definition after the binding has been defined. The modifications can affect the binding definition or possibly the source code entered in the form.

Classification 1
Save

Properties
Saves the current property values of an SAPData object in a structured string. Counterpart of the Load method.
When a form is created, the SAPforms Designer uses this method to save the structure description retrieved from the R/3 System permanently in the form.
The R/3 structure is restored from the structured string at runtime by the Load method.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Tgt</td>
<td>String in which the status is stored.</td>
</tr>
</tbody>
</table>

Classification 2
Save

Properties
Saves the current property values of an SAPData object in a structured string. Counterpart of the Load method. The string is saved with a form so that the structure information is available when the form is executed.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Tgt</td>
<td>Stores the string structure.</td>
</tr>
</tbody>
</table>
**Classification 3**

**Save**

- Fields Class [Seite 2018]
- SAPforms Data Plug-In for Workflow [Seite 1897]
- SAPforms Data Plug-In for Function Modules [Seite 1954]
- SAPforms Data Plug-In for IDoc [Seite 2006]

**Properties**

Loads the property values of a fields object from a structured string. As a result, the status of the fields object that was previously saved permanently with the Save method is restored. The string is saved with the form so that the structure information is available when the form is executed.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Tgt</td>
<td>Stores the string structure.</td>
</tr>
</tbody>
</table>

**Classification 4**

**Save**

- Segment class [Seite 2055]
- SAPforms Data Plug-In for IDoc [Seite 2006]

**Properties**

This method saves the structure of a segment, including the field list, in a structured string.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Tgt</td>
<td>Stores the string structure.</td>
</tr>
</tbody>
</table>

**Classification 5**

**Save**

- Segments class [Seite 2067]
- SAPforms Data Plug-In for IDoc [Seite 2006]

**Properties**

This method saves the segment list in a structured string. Counterpart of the Load method.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Tgt</td>
<td>Stores the string structure.</td>
</tr>
</tbody>
</table>

**Classification 6**

**Save**

- Links class [Seite 1862]
- SAPforms Form Plug-In for Visual Basic [Seite 1756]
- Links class [Seite 1862]
- SAPforms Form Plug-In for MS Office [Seite 1856]
- Links class [Seite 1862]
- SAPforms Form Plug-In for MS Outlook [Seite 1796]
Properties
Method that writes the content of all links to a file. Counterpart of the Load method.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Tgt</td>
<td>Stores the string structure.</td>
</tr>
</tbody>
</table>

Classification 7

Save

\[ \text{SAPForm class [Seite 1772]} \]
\[ \text{SAPforms Form Plug-In for Visual Basic [Seite 1756]} \]
\[ \text{SAPForm class [Seite 1872]} \]
\[ \text{SAPforms Form Plug-In for MS Office [Seite 1856]} \]
\[ \text{SAPForm class [Seite 1812]} \]
\[ \text{SAPforms Form Plug-In for MS Outlook [Seite 1796]} \]

Properties
Method that saves a form to a file if it was created from a file. Reads the old file and writes updated lines to the new file.

Classification 8

Save

\[ \text{SAPXmit class [Seite 2106]} \]
\[ \text{SAPforms Data Plug-In for Demo [Seite 2077]} \]
\[ \text{SAPXmit class [Seite 1837]} \]
\[ \text{SAPforms Form Plug-In for Outlook [Seite 1796]} \]
\[ \text{SAPXmit class [Seite 2126]} \]
\[ \text{SAPforms Transmit Plug-In for RFC [Seite 2125]} \]
\[ \text{SAPXmit class [Seite 2168]} \]
\[ \text{SAPforms Transmit Plug-In for MAPI [Seite 2146]} \]
\[ \text{SAPXmit class [Seite 2147]} \]
\[ \text{SAPforms Transmit Plug-In for SAPMAPI [Seite 2167]} \]
\[ \text{SAPXmit class [Seite 2189]} \]
\[ \text{SAPforms Transmit Plug-In for WinSock [Seite 2188]} \]
\[ \text{SAPXmit class [Seite 2209]} \]
\[ \text{SAPforms Transmit Plug-In for Demo [Seite 2208]} \]

Properties
Saves the current property values of an SAPXmit object in a structured string. Counterpart of the Load method.
When a form is created, the SAPforms Designer uses this method to save the structure description retrieved from the R/3 System permanently in the form.
The R/3 structure is restored from the structured string at runtime by the \textit{Load} method.
<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Tgt</td>
<td>String in which the status is stored.</td>
</tr>
</tbody>
</table>
Start (Method)

Definition
Method that executes the *Start* function in R/3. The semantics of this function depend on the concrete SAPData object.

Remarks
In the case of the SAPData object for workflow containers, this method starts a workflow and populates its container elements with values from the form fields. This method can be called directly via the SAPData object or indirectly via the *Start* method of the SAPForm object, whose SAPData property is a reference to the SAPData object. If an SAPXmit object is assigned to the SAPData object, the *Start* method of the SAPXmit object is called.

Classification 1

Start

- SAPData class [Seite 1918]
- SAPforms Data Plug-In for Workflow [Seite 1897]

Properties
Starts a workflow in the R/3 System and supplies its container elements with the data values in the fields list.

Classification 2

Start

- SAPData class [Seite 1975]
- SAPforms Data Plug-In for Function Modules [Seite 1954]

Properties
This method carries out the current R/3 function module call.

Classification 3

Start

- SAPData class [Seite 2027]
- SAPforms Data Plug-In for IDoc [Seite 2006]

Properties
Generates an IDoc file from the data in the segments list and transfers this IDoc to the R/3 System. The transfer method depends on the assigned SAPXmit object. A workflow can be started in the R/3 System when the inbound IDoc is processed.

Classification 4

Start

- SAPData class [Seite 2078]
- SAPforms Data Plug-In for Demo [Seite 2077]
Properties
Does not execute an action in the R/3 System but rather displays the data of the object in a separate window.

Classification 5
Start

SAPForm class [Seite 1772]
SAPforms Form Plug-In for Visual Basic [Seite 1756]
SAPForm class [Seite 1872]
SAPforms Form Plug-In for MS Office [Seite 1856]
SAPForm class [Seite 1812]
SAPforms Form Plug-In for MS Outlook [Seite 1796]

Properties
Invokes the Start method in the associated SAPXmit or SAPData object. The semantics of this method depend on this SAPData object. If this object is the SAPData object for SAP Business Workflow, this method starts a workflow in an R/3 System.

Classification 6
Start

SAPXmit class [Seite 2126]
SAPforms Transmit Plug-In for RFC [Seite 2125]

Properties
This method starts a workflow synchronously via RFC. Supports SAPData objects of types "Idoc" and "Container".

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Data</td>
<td>Object of type SAPData.</td>
</tr>
</tbody>
</table>

Classification 7
Start

SAPXmit class [Seite 2168]
SAPforms Transmit Plug-In for MAPI [Seite 2146]
SAPXmit class [Seite 2147]
SAPforms Transmit Plug-In for SAPMAPI [Seite 2167]
SAPXmit class [Seite 2189]
SAPforms Transmit Plug-In for SMTP [Seite 2188]
SAPXmit class [Seite 1837]
SAPforms Form Plug-In for MS Outlook [Seite 1796]

Properties
Sends a mail to the R/3 System with an attachment containing the information required to start a workflow.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Data</td>
<td>Object of type SAPData.</td>
</tr>
</tbody>
</table>
Classification 8

Start

- SAPXmit class [Seite 2209]
- SAPforms Transmit Plug-In for Demo [Seite 2208]
- SAPXmit class [Seite 2106]
- SAPforms Data Plug-In for Demo [Seite 2077]

Properties

Displays the instance data of the associated SAPData object in a window instead of sending it to the R/3 System.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Data</td>
<td>Object of type SAPData.</td>
</tr>
</tbody>
</table>
**SAPXmit Plug-In Type**

**Definition**

This plug-in is provided or each communication method that is to be used to exchange data between the form and the R/3 System.

*The plug-in type SAXmit in relation to the various plug-in types [Seite 1746].*
Use

Communication Methods

Communication between the form and the R/3 System can take place both online via RFC (SAPgui), offline via mail, and via SAP MAPI. The SAPforms Designer [Seite 1735] provides the following plug-ins for this purpose:

<table>
<thead>
<tr>
<th>Plug-In</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>SAPforms Transmit Plug-In for MAPI [Seite 2146]</td>
<td>Plug-in for asynchronous communication via a MAPI client that supports Internet mail.</td>
</tr>
<tr>
<td>SAPforms Transmit Plug-In for SAPMAPI [Seite 2167]</td>
<td>Plug-in for communication via the SAPMAPI service provider.</td>
</tr>
<tr>
<td>SAPforms Transmit Plug-In for SMTP via WinSock [Seite 2188]</td>
<td>Plug-in for asynchronous communication with Internet mail.</td>
</tr>
<tr>
<td>SAPforms Transmit Plug-In for RFC [Seite 2125]</td>
<td>Plug-in for synchronous communication via Remote Function Call (RFC).</td>
</tr>
<tr>
<td>SAPforms Transmit Plug-In Demo [Seite 2208]</td>
<td>Plug-in for demonstration purposes (no R/3 access needed)</td>
</tr>
</tbody>
</table>
SAPforms Transmit Plug-In for RFC

Definition
SAPXmit type plug-in for synchronous data transfer via Remote Function Call (RFC).

Use
You use this plug-in whenever a network connection can be set up from the user’s PC to provide online access to the R/3 System via RFC. SAPGUI does not need to be installed on the PC in question. You can use this plug-in with the SAPData plug-ins for SAP Business Workflow and for IDocs. This plug-in implements the SAPXmit interface through which other SAPForm and SAPData objects are enabled to transmit data to R/3. In other words, this plug-in can be regarded as being synonymous with the associated SAPXmit class.

Classification
SAPforms Transmit Plug-In for RFC

Properties
You can maintain the following properties for this plug-in:

Always bring up R/3 system selection:

If you have selected this indicator, you must specify the R/3 System, in which the R/3 functionality is to be executed, every time you execute the form. If you have not selected this indicator, the RFC is executed in the system you specified when you defined the form.
SAPXmit Class

Definition
Original class of the SAPforms Transmit Plug-In for RFC.

Remarks
This class implements the SAPXmit interface through which other SAPForm and SAPData objects are enabled to transmit data to R/3. In other words, these classes can be regarded as being synonymous with various plug-ins of this type. Communication between the form and the R/3 System be either synchronous (via RFC) or asynchronous (via mail). The class supplies methods for communication via RFC.

If you want to develop your own SAPXmit-type plug-ins, you should use the source code supplied for the SAPforms Transmit Plug-Ins for Demo as a reference (template). The template is stored in the directory .../sapforms/sapxdemo.

Classification
SAPforms Transmit Plug-In for RFC [Seite 2125]

→ SAPXmit Class
   → Execute (Method) [Seite 2210]
   → LastR3Error (Property) [Seite 2130]
   → Load (Method) [Seite 2213]
   → Options (Method) [Seite 2217]
   → R3DoNotUse (Property) [Seite 2137]
   → RfcHandle (Property) [Seite 2138]
   → Save (Method) [Seite 2219]
   → Start (Method) [Seite 2223]
Execute (Method)

Definition
Method that calls the *Execute* function in the R/3 System. The semantics of this function depend on the SAPData object.

Remarks
The method can be called directly via the SAPData object or indirectly via the Start method of the SAPForm object, whose SAPData property is a reference to the SAPData object.

In the case of the SAPData object for workflow containers, for example, this method executes a work item in an SAP Business Workflow and populates its container elements with values from the form fields.

Classification 1
Execute

- SAPData class [Seite 1918]
- SAPforms Data Plug-In for Workflow [Seite 1897]

Properties
Terminates a work item in the R/3 System and supplies its container elements with the data values in the fields list.

Classification 2
Execute

- SAPData class [Seite 2078]
- SAPXmit class [Seite 2106]
- SAPforms Data Plug-In for Demo [Seite 2077]

Properties
Does not execute an action in the R/3 System but rather displays the data of the object in a separate window.

Classification 3
Execute

- SAPForm class [Seite 1772]
- SAPforms Form Plug-In for Visual Basic [Seite 1756]
- SAPForm class [Seite 1872]
- SAPforms Form Plug-In for MS Office [Seite 1856]
- SAPForm class [Seite 1812]
- SAPforms Form Plug-In for MS Outlook [Seite 1796]

Properties
Invokes the Execute method in the associated Xmit or SAPData object. The semantics of this method depend on this object.
**Classification 4**

**Execute**

- **SAPXmit class [Seite 2126]**
- **SAPforms Transmit Plug-In for RFC [Seite 2125]**

**Properties**

This method executes a work item synchronously via a remote function call (RFC). Supports SAPData object of type *Container*.

This method causes a certain function module to be executed in the R/3 System. This module receives a table with a complete R3F structure containing the work item information and the container data that are to be supplied from the form.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Data</td>
<td>Object representing the container data.</td>
</tr>
</tbody>
</table>

**Classification 5**

**Execute**

- **SAPXmit class [Seite 2168]**
- **SAPforms Transmit Plug-In for MAPI [Seite 2146]**
- **SAPXmit class [Seite 2189]**
- **SAPforms Transmit Plug-In for WinSock [Seite 2188]**

**Properties**

Sends a mail to the R/3 System that contains the information required to terminate a work item (including container data) as an attachment.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Data</td>
<td>Object representing the container data.</td>
</tr>
</tbody>
</table>

**Classification 6**

**Execute**

- **SAPXmit class [Seite 2147]**
- **SAPforms Transmit Plug-In for SAPMAPI [Seite 2167]**

**Properties**

Overwrites the R3F attachment, thereby updating it.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Pdata</td>
<td>Object that represents the <em>mobjData</em> container data</td>
</tr>
</tbody>
</table>

**Classification 7**

**Execute**

- **SAPXmit class [Seite 2209]**
- **SAPforms Transmit Plug-In for Demo [Seite 2208]**

**Properties**

Displays the instance data of the associated SAPData object in a window instead of sending it to the R/3 System.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Data</td>
<td>object of class SAPData</td>
</tr>
</tbody>
</table>
LastR3Error (Property)

Definition
Returns the message text of the last R/3 error that occurred during an RFC call.

Remarks

Responsibility

LastR3Error

← SAPXmit class [Seite 2126]

← SAPforms Transmit Plug-In for RFC [Seite 2125]
Load (Method)

Definition
Loads structure information to SAPforms objects.

Remarks

Classification 1

Load

Properties
Loads the property values of an SAPData object from a structured string. As a result, the status of the SAPData object that was previously saved permanently with the Save method is restored. This saves the additional overheads that would result if the structure information had to be loaded each time from the original source. This method is usually called within the Load method of the SAPForm class.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>From</td>
<td>Character string containing the structure information</td>
</tr>
</tbody>
</table>

Classification 2

Load

Properties
Loads the property values of a field object from a structured string. As a result, the status of the field object that was previously saved permanently with the Save method is restored. For this method to be called successfully, the next character at the Pos position must be keyword “Fld”.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>From</td>
<td>Structured string</td>
</tr>
<tr>
<td>Pos</td>
<td>Current position within the string. Should point to the number of remaining properties to read</td>
</tr>
</tbody>
</table>

Classification 3

Load
Properties
Loads the property values of a fields object from a structured string. As a result, the status of the fields object that was previously saved permanently with the Save method is restored.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>From</td>
<td>Character string that provides the structure information.</td>
</tr>
<tr>
<td>Pos</td>
<td>Current search position within the character string.</td>
</tr>
</tbody>
</table>

Classification 4
Load

Properties
Reconstructs the segment object from a structured string. Counterpart of the Save method.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>From</td>
<td>Character string containing structure information.</td>
</tr>
<tr>
<td>Pos</td>
<td>Current search position within the character string.</td>
</tr>
</tbody>
</table>

Classification 5
Load

Properties
This method reconstructs the segment list from a structured string. Counterpart of the Save method.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>From</td>
<td>Character string containing structure information.</td>
</tr>
<tr>
<td>Pos</td>
<td>Current search position within the character string.</td>
</tr>
</tbody>
</table>

Classification 6
Load

Properties
This method reconstructs an object of the Link class from the serialized representation. It assumes that the next token is located at the Pos position in the serialized representation of the “Fld” keyword.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>From</td>
<td>Serialization string.</td>
</tr>
</tbody>
</table>
Properties
Method for loading form controls into Controls Collection from a form or form template.
Note that the form information is not stored in the Tag property of this object, but reconstructed each time the form is loaded. This method is called at design time and at runtime. It has to populate the Controls collection for both cases. At runtime, the method is called by the user code in the form's Load method. When the form is designed, the method is called by the SAPForms Designer or from outside SAPForms. If you are using the SAPForms Designer as a stand-alone application, the Load method is called with Mode=2. This instructs the Load method to load the form from a file. If you are using the SAPForms Designer as a plug-in, the Load method is invoked from the hosting environment and receives the proprietary form description as a parameter. The populated SAPForm object should be passed to the SAPForms Designer by the PlugIn method of the Designer class.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Frm</td>
<td>form or form template containing controls.</td>
</tr>
<tr>
<td>Mode</td>
<td>0=design time (add-in), 1=design time (file), 2=runtime.</td>
</tr>
</tbody>
</table>

Classification 8
Load

SAPXmit class [Seite 2106]
SAPforms Data Plug-In for Demo [Seite 2077]
SAPForm class [Seite 1837]
SAPforms Form Plug-In for Outlook [Seite 1796]
SAPXmit class [Seite 2126]
SAPforms Transmit Plug-In for RFC [Seite 2125]
SAPXmit class [Seite 2168]
SAPforms Transmit Plug-In for MAPI [Seite 2146]
SAPXmit class [Seite 2147]
SAPforms Transmit Plug-In for SAPMAPI [Seite 2167]
SAPXmit class [Seite 2189]
SAPforms Transmit Plug-In for WinSock [Seite 2188]
SAPXmit class [Seite 2209]
SAPforms Transmit Plug-In for Demo [Seite 2208]
Load (Method)

Properties
Loads the property values of an SAPXmit object from a structured string. As a result, the status of the SAPXmit object that was previously saved permanently with the Save method is restored.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>From</td>
<td>Character string containing the structure information</td>
</tr>
</tbody>
</table>
Options (Method)

Definition
Method that outputs the object-specific options dialog box.

Remarks
This dialog box displays either global settings or settings that are valid for the current form. These settings can be changed here. The global settings are read from the registry before they are displayed and are then saved again to the registry.
The method is called within the SAPforms Designer if the required plug-in is active under Properties on the Plug-Ins tab in the Options dialog. You can also use the sample program Options.exe to display the Options dialog of a certain plug-in.

Classification 1
Options

- SAPData class [Seite 1918]
- SAPforms Data Plug-In for Workflow [Seite 1897]
- SAPData class [Seite 1975]
- SAPforms Data Plug-In for Function Modules [Seite 1954]
- SAPData class [Seite 2027]
- SAPforms Data Plug-In for IDoc [Seite 2006]
- SAPData class [Seite 2078]
- SAPforms Data Plug-In for Demo [Seite 2077]

Classification 2
Options

- SAPForm class [Seite 1772]
- SAPforms Form Plug-In for Visual Basic [Seite 1756]
- SAPForm class [Seite 1812]
- SAPforms Form Plug-In for MS Outlook [Seite 1796]
- SAPForm class [Seite 1872]
- SAPforms Form Plug-In for MS Office [Seite 1856]

Classification 3
Options

- SAPXmit class [Seite 2106]
- SAPforms Data Plug-In for Demo [Seite 2077]
- SAPXmit class [Seite 1837]
- SAPforms Form Plug-In for MS Outlook [Seite 1796]
- SAPXmit class [Seite 2126]
Options (Method)

⇔ SAPforms Transmit Plug-In for RFC [Seite 2125]
⇔ SAPXmit class [Seite 2168]
⇔ SAPforms Transmit Plug-In for MAPI [Seite 2146]
⇔ SAPXmit class [Seite 2147]
⇔ SAPforms Transmit Plug-In for SAPMAPI [Seite 2167]
⇔ SAPXmit class [Seite 2189]
⇔ SAPforms Transmit Plug-In for SMTP [Seite 2188]
⇔ SAPXmit class [Seite 2209]
⇔ SAPforms Transmit Plug-In for Demo [Seite 2208]
R3DoNotUse (Property)

Definition
Determined whether the R/3 system information, which was stored permanently in the form when it was designed, is to be used when the Start() and Execute() methods are executed.

Remarks
Classification
R3DoNotUse

Properties
The default value of this field is FALSE. When the methods are executed, logon takes place to the R/3 System that was used to design the form. This R/3 System selection is skipped during logon. If the same form is to be used with different R/3 Systems, you must set this property to TRUE. You can specify this either in the form's program code or by using the options dialog of the SAPData object.
RfcHandle (Property)

Definition
Represents the reference to a connection to an R/3 System.

Remarks

Classification

Properties
This reference is used if it exists. Otherwise, an R/3 Logon is carried out.
Save (Method)

Definition
Saves the updated form definition.

Remarks
This method is used by the SAPforms Designer when a form is designed to save the modified form definition after the binding has been defined. The modifications can affect the binding definition or possibly the source code entered in the form.

Classification 1
Save

<table>
<thead>
<tr>
<th>Classification</th>
<th>Save</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Properties
Saves the current property values of an SAPData object in a structured string. Counterpart of the Load method.
When a form is created, the SAPforms Designer uses this method to save the structure description retrieved from the R/3 System permanently in the form.
The R/3 structure is restored from the structured string at runtime by the Load method.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Tgt</td>
<td>String in which the status is stored.</td>
</tr>
</tbody>
</table>

Classification 2
Save

<table>
<thead>
<tr>
<th>Classification</th>
<th>Save</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Properties
Saves the current property values of an SAPData object in a structured string. Counterpart of the Load method. The string is saved with a form so that the structure information is available when the form is executed.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Tgt</td>
<td>Stores the string structure.</td>
</tr>
</tbody>
</table>
**Classification 3**

**Save**

- `Fields Class [Seite 2018]`
- `SAPforms Data Plug-In for Workflow [Seite 1897]`
- `SAPforms Data Plug-In for Function Modules [Seite 1954]`
- `SAPforms Data Plug-In for IDoc [Seite 2006]`

**Properties**

Loads the property values of a fields object from a structured string. As a result, the status of the fields object that was previously saved permanently with the Save method is restored. The string is saved with the form so that the structure information is available when the form is executed.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Tgt</td>
<td>Stores the string structure.</td>
</tr>
</tbody>
</table>

**Classification 4**

**Save**

- `Segment class [Seite 2055]`
- `SAPforms Data Plug-In for IDoc [Seite 2006]`

**Properties**

This method saves the structure of a segment, including the field list, in a structured string.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Tgt</td>
<td>Stores the string structure.</td>
</tr>
</tbody>
</table>

**Classification 5**

**Save**

- `Segments class [Seite 2067]`
- `SAPforms Data Plug-In for IDoc [Seite 2006]`

**Properties**

This method saves the segment list in a structured string. Counterpart of the Load method.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Tgt</td>
<td>Stores the string structure.</td>
</tr>
</tbody>
</table>

**Classification 6**

**Save**

- `Links class [Seite 1862]`
- `SAPforms Form Plug-In for Visual Basic [Seite 1756]`
- `Links class [Seite 1862]`
- `SAPforms Form Plug-In for MS Office [Seite 1856]`
- `Links class [Seite 1862]`
- `SAPforms Form Plug-In for MS Outlook [Seite 1796]`
**Properties**

Method that writes the content of all links to a file. Counterpart of the Load method.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Tgt</td>
<td>Stores the string structure.</td>
</tr>
</tbody>
</table>

**Classification 7**

**Save**

- **SAPForm class** [Seite 1772]
  - **SAPforms Form Plug-In for Visual Basic** [Seite 1756]
- **SAPForm class** [Seite 1872]
  - **SAPforms Form Plug-In for MS Office** [Seite 1856]
- **SAPForm class** [Seite 1812]
  - **SAPforms Form Plug-In for MS Outlook** [Seite 1796]

**Properties**

Method that saves a form to a file if it was created from a file. Reads the old file and writes updated lines to the new file.

**Classification 8**

**Save**

- **SAPXmit class** [Seite 2106]
  - **SAPforms Data Plug-In for Demo** [Seite 2077]
- **SAPXmit class** [Seite 1837]
  - **SAPforms Form Plug-In for Outlook** [Seite 1796]
- **SAPXmit class** [Seite 2126]
  - **SAPforms Transmit Plug-In for RFC** [Seite 2125]
- **SAPXmit class** [Seite 2168]
  - **SAPforms Transmit Plug-In for MAPI** [Seite 2146]
- **SAPXmit class** [Seite 2147]
  - **SAPforms Transmit Plug-In for SAPMAPI** [Seite 2167]
- **SAPXmit class** [Seite 2189]
  - **SAPforms Transmit Plug-In for WinSock** [Seite 2188]
- **SAPXmit class** [Seite 2209]
  - **SAPforms Transmit Plug-In for Demo** [Seite 2208]

**Properties**

Saves the current property values of an SAPXmit object in a structured string. Counterpart of the Load method.

When a form is created, the SAPforms Designer uses this method to save the structure description retrieved from the R/3 System permanently in the form.

The R/3 structure is restored from the structured string at runtime by the *Load* method.
### Save (Method)

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Tgt</td>
<td>String in which the status is stored.</td>
</tr>
</tbody>
</table>
Start (Method)

Definition
Method that executes the Start function in R/3. The semantics of this function depend on the concrete SAPData object.

Remarks
In the case of the SAPData object for workflow containers, this method starts a workflow and populates its container elements with values from the form fields. This method can be called directly via the SAPData object or indirectly via the Start method of the SAPForm object, whose SAPData property is a reference to the SAPData object. If an SAPXmit object is assigned to the SAPData object, the Start method of the SAPXmit object is called.

Classification 1
Start
- SAPData class [Seite 1918]
- SAPforms Data Plug-In for Workflow [Seite 1897]

Properties
Starts a workflow in the R/3 System and supplies its container elements with the data values in the fields list.

Classification 2
Start
- SAPData class [Seite 1975]
- SAPforms Data Plug-In for Function Modules [Seite 1954]

Properties
This method carries out the current R/3 function module call.

Classification 3
Start
- SAPData class [Seite 2027]
- SAPforms Data Plug-In for IDoc [Seite 2006]

Properties
Generates an IDoc file from the data in the segments list and transfers this IDoc to the R/3 System. The transfer method depends on the assigned SAPXmit object. A workflow can be started in the R/3 System when the inbound IDoc is processed.

Classification 4
Start
- SAPData class [Seite 2078]
- SAPforms Data Plug-In for Demo [Seite 2077]
Properties
Does not execute an action in the R/3 System but rather displays the data of the object in a separate window.

Classification 5
Start

- SAPForm class [Seite 1772]
- SAPforms Form Plug-In for Visual Basic [Seite 1756]
- SAPForm class [Seite 1872]
- SAPforms Form Plug-In for MS Office [Seite 1856]
- SAPForm class [Seite 1812]
- SAPforms Form Plug-In for MS Outlook [Seite 1796]

Properties
Invokes the Start method in the associated SAPXmit or SAPData object. The semantics of this method depend on this SAPData object. If this object is the SAPData object for SAP Business Workflow, this method starts a workflow in an R/3 System.

Classification 6
Start

- SAPXmit class [Seite 2126]
- SAPforms Transmit Plug-In for RFC [Seite 2125]

Properties
This method starts a workflow synchronously via RFC. Supports SAPData objects of types "Idoc" and "Container".

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Data</td>
<td>Object of type SAPData.</td>
</tr>
</tbody>
</table>

Classification 7
Start

- SAPXmit class [Seite 2168]
- SAPforms Transmit Plug-In for MAPI [Seite 2146]
- SAPXmit class [Seite 2147]
- SAPforms Transmit Plug-In for SAPMAPI [Seite 2167]
- SAPXmit class [Seite 2189]
- SAPforms Transmit Plug-In for SMTP [Seite 2188]
- SAPXmit class [Seite 1837]
- SAPforms Form Plug-In for MS Outlook [Seite 1796]

Properties
Sends a mail to the R/3 System with an attachment containing the information required to start a workflow.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Data</td>
<td>Object of type SAPData.</td>
</tr>
</tbody>
</table>
### Classification 8

**Start**

- SAPXmit class [Seite 2209]
- SAPforms Transmit Plug-In for Demo [Seite 2208]
- SAPXmit class [Seite 2106]
- SAPforms Data Plug-In for Demo [Seite 2077]

**Properties**
Displays the instance data of the associated SAPData object in a window instead of sending it to the R/3 System.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Data</td>
<td>Object of type SAPData.</td>
</tr>
</tbody>
</table>
SAPforms Transmit Plug-In for MAPI

Definition
SAPXmit type communication plug-in for asynchronous communication with MAPI mail.

Use
You use this plug-in if a MAPI client that supports Internet mail is installed on the user’s PC. This plug-in can be used with the SAPData plug-in for workflow.

The worklist appears on the client in the form of incoming mails. The Visual Basic form is then started via the R3F attachment in this mail. This form must be stored in the directory ...\SAPforms\Forms or in the active directory.

See also SAPforms Helper [Seite 1741].

This plug-in implements the SAPXmit interface through which other SAPForm and SAPData objects are enabled to transmit data to R/3. In other words, this plug-in can be regarded as being synonymous with the associated SAPXmit class.

Classification
SAPforms Transmit Plug-In for MAPI

⇒ SAPXmit class [Seite 2147]

Properties
You can maintain the following properties for this plug-in:

Use Default MAPI Profile:

The profile of the MAPI client determines which services are active and contain Internet-capable mail services. If you select this indicator, the system will not ask you to specify a profile when you execute the form.
**SAPXmit Class**

**Definition**

Original class of the SAPforms Transmit Plug-In for MAPI.

**Remarks**

This class implements the SAPXmit interface through which other SAPForm and SAPData objects are enabled to transmit data to R/3. In other words, these classes can be regarded as being synonymous with various plug-ins of this type. Communication between the form and the R/3 System be either synchronous (via RFC) or asynchronous (via mail). The class supplies methods for communication via MAPI.

If you want to develop your own SAPXmit-type plug-ins, you should use the source code supplied for the *SAPforms Transmit Plug-Ins for Demo* as a reference (template). The template is stored in the directory `.../sapforms/sapxdemo`.

**Classification**

*SAPforms Transmit Plug-In for MAPI [Seite 2146]*

→ SAPXmit Class

→ **Execute (Method) [Seite 2210]**
→ **Load (Method) [Seite 2213]**
→ **MailItem (Property) [Seite 2176]**
→ **Options (Method) [Seite 2217]**
→ **ProfileName (Property) [Seite 2179]**
→ **Save (Method) [Seite 2219]**
→ **Start (Method) [Seite 2223]**
→ **Subject (Property) [Seite 2207]**
Execute (Method)

Definition
Method that calls the *Execute* function in the R/3 System. The semantics of this function depend on the SAPData object.

Remarks
The method can be called directly via the SAPData object or indirectly via the Start method of the SAPForm object, whose SAPData property is a reference to the SAPData object.

In the case of the SAPData object for workflow containers, for example, this method executes a work item in an SAP Business Workflow and populates its container elements with values from the form fields.

Classification 1
Execute

- SAPData class [Seite 1918]
- SAPforms Data Plug-In for Workflow [Seite 1897]

Properties
Terminates a work item in the R/3 System and supplies its container elements with the data values in the fields list.

Classification 2
Execute

- SAPData class [Seite 2078]
- SAPXmit class [Seite 2106]
- SAPforms Data Plug-In for Demo [Seite 2077]

Properties
Does not execute an action in the R/3 System but rather displays the data of the object in a separate window.

Classification 3
Execute

- SAPForm class [Seite 1772]
- SAPforms Form Plug-In for Visual Basic [Seite 1756]
- SAPForm class [Seite 1872]
- SAPforms Form Plug-In for MS Office [Seite 1856]
- SAPForm class [Seite 1812]
- SAPforms Form Plug-In for MS Outlook [Seite 1796]

Properties
Invokes the Execute method in the associated Xmit or SAPData object. The semantics of this method depend on this object.
Classification 4

Execute

- SAPXmit class [Seite 2126]
- SAPforms Transmit Plug-In for RFC [Seite 2125]

Properties
This method executes a work item synchronously via a remote function call (RFC). Supports SAPData object of type Container.

This method causes a certain function module to be executed in the R/3 System. This module receives a table with a complete R3F structure containing the work item information and the container data that are to be supplied from the form.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Data</td>
<td>Object representing the container data.</td>
</tr>
</tbody>
</table>

Classification 5

Execute

- SAPXmit class [Seite 2168]
- SAPforms Transmit Plug-In for MAPI [Seite 2146]
- SAPXmit class [Seite 2189]
- SAPforms Transmit Plug-In for WinSock [Seite 2188]

Properties
Sends a mail to the R/3 System that contains the information required to terminate a work item (including container data) as an attachment.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Data</td>
<td>Object representing the container data.</td>
</tr>
</tbody>
</table>

Classification 6

Execute

- SAPXmit class [Seite 2147]
- SAPforms Transmit Plug-In for SAPMAPI [Seite 2167]

Properties
Overwrites the R3F attachment, thereby updating it.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Pdata</td>
<td>Object that represents the mobjData container data</td>
</tr>
</tbody>
</table>

Classification 7

Execute

- SAPXmit class [Seite 2209]
- SAPforms Transmit Plug-In for Demo [Seite 2208]

Properties
Displays the instance data of the associated SAPData object in a window instead of sending it to the R/3 System.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Data</td>
<td>object of class SAPData</td>
</tr>
</tbody>
</table>
Load (Method)

Definition
Loads structure information to SAPforms objects.

Remarks

Classification 1
Load

Properties
Loads the property values of an SAPData object from a structured string. As a result, the status of the SAPData object that was previously saved permanently with the Save method is restored. This saves the additional overheads that would result if the structure information had to be loaded each time from the original source. This method is usually called within the Load method of the SAPForm class.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>From</td>
<td>Character string containing the structure information</td>
</tr>
</tbody>
</table>

Classification 2
Load

Properties
Loads the property values of a field object from a structured string. As a result, the status of the field object that was previously saved permanently with the Save method is restored. For this method to be called successfully, the next character at the Pos position must be keyword “Fld”.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>From</td>
<td>Structured string</td>
</tr>
<tr>
<td>Pos</td>
<td>Current position within the string. Should point to the number of remaining properties to read</td>
</tr>
</tbody>
</table>

Classification 3
Load
Load (Method)

Properties

Loads the property values of a fields object from a structured string. As a result, the status of the fields object that was previously saved permanently with the Save method is restored.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>From</td>
<td>Character string that provides the structure information.</td>
</tr>
<tr>
<td>Pos</td>
<td>Current search position within the character string.</td>
</tr>
</tbody>
</table>

Classification 4

Load

Properties

Reconstructs the segment object from a structured string. Counterpart of the Save method.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>From</td>
<td>Character string containing structure information.</td>
</tr>
<tr>
<td>Pos</td>
<td>Current search position within the character string.</td>
</tr>
</tbody>
</table>

Classification 5

Load

Properties

This method reconstructs the segment list from a structured string. Counterpart of the Save method.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>From</td>
<td>Character string containing structure information.</td>
</tr>
<tr>
<td>Pos</td>
<td>Current search position within the character string.</td>
</tr>
</tbody>
</table>

Classification 6

Load

Properties

This method reconstructs an object of the Link class from the serialized representation. It assumes that the next token is located at the Pos position in the serialized representation of the “Fld” keyword.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>From</td>
<td>Serialization string.</td>
</tr>
</tbody>
</table>
Load (Method)

Current position within the character string.

Classification 7

Load

- SAPForm class [Seite 1772]
- SAPforms Form Plug-In for Visual Basic [Seite 1756]
- SAPForm class [Seite 1812]
- SAPforms Form Plug-In for MS Outlook [Seite 1796]
- SAPForm class [Seite 1872]
- SAPforms Form Plug-In for MS Office [Seite 1856]

Properties

Method for loading form controls into Controls Collection from a form or form template. Note that the form information is not stored in the Tag property of this object, but reconstructed each time the form is loaded. This method is called at design time and at runtime. It has to populate the Controls collection for both cases. At runtime, the method is called by the user code in the form's Load method. When the form is designed, the method is called by the SAPForms Designer or from outside SAPForms. If you are using the SAPForms Designer as a stand-alone application, the Load method is called with Mode=2. This instructs the Load method to load the form from a file. If you are using the SAPForms Designer as a plug-in, the Load method is invoked from the hosting environment and receives the proprietary form description as a parameter. The populated SAPForm object should be passed to the SAPForms Designer by the PlugIn method of the Designer class.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Frm</td>
<td>form or form template containing controls.</td>
</tr>
<tr>
<td>Mode</td>
<td>0=design time (add-in), 1=design time (file), 2=runtime.</td>
</tr>
</tbody>
</table>

Classification 8

Load

- SAPXmit class [Seite 2106]
- SAPforms Data Plug-In for Demo [Seite 2077]
- SAPXmit class [Seite 1837]
- SAPforms Form Plug-In for Outlook [Seite 1796]
- SAPXmit class [Seite 2126]
- SAPforms Transmit Plug-In for RFC [Seite 2125]
- SAPXmit class [Seite 2168]
- SAPforms Transmit Plug-In for MAPI [Seite 2146]
- SAPXmit class [Seite 2147]
- SAPforms Transmit Plug-In for SAPMAPI [Seite 2167]
- SAPXmit class [Seite 2189]
- SAPforms Transmit Plug-In for WinSock [Seite 2188]
- SAPXmit class [Seite 2209]
- SAPforms Transmit Plug-In for Demo [Seite 2208]
Properties
Loads the property values of an SAPXmit object from a structured string. As a result, the status of the SAPXmit object that was previously saved permanently with the Save method is restored.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>From</td>
<td>Character string containing the structure information</td>
</tr>
</tbody>
</table>
**MailItem (Property)**

**Definition**
Reference to the mail that is used to transport the form data.

**Remarks**
The property must be populated “externally” before the *Start* or *Execute* method is executed.

**Classification**

**MailItem**

- SAPXmit class [Seite 1837]
- SAPforms Form Plug-In for MS Outlook [Seite 1796]
- SAPXmit class [Seite 2168]
- SAPforms Transmit Plug-In for MAPI [Seite 2146]
- SAPXmit class [Seite 2147]
- SAPforms Transmit Plug-In for SAPMAPI [Seite 2167]
Options (Method)

Definition
Method that outputs the object-specific options dialog box.

Remarks
This dialog box displays either global settings or settings that are valid for the current form. These settings can be changed here. The global settings are read from the registry before they are displayed and are then saved again to the registry.

The method is called within the SAPforms Designer if the required plug-in is active under Properties on the Plug-Ins tab in the Options dialog. You can also use the sample program Options.exe to display the Options dialog of a certain plug-in.

Classification 1
Options

- SAPData class [Seite 1918]
- SAPForms Data Plug-In for Workflow [Seite 1897]
- SAPData class [Seite 1975]
- SAPForms Data Plug-In for Function Modules [Seite 1954]
- SAPData class [Seite 2027]
- SAPForms Data Plug-In for IDoc [Seite 2006]
- SAPData class [Seite 2078]
- SAPForms Data Plug-In for Demo [Seite 2077]

Classification 2
Options

- SAPForm class [Seite 1772]
- SAPForms Form Plug-In for Visual Basic [Seite 1756]
- SAPForm class [Seite 1812]
- SAPForms Form Plug-In for MS Outlook [Seite 1796]
- SAPForm class [Seite 1872]
- SAPForms Form Plug-In for MS Office [Seite 1856]

Classification 3
Options

- SAPXmit class [Seite 2106]
- SAPForms Data Plug-In for Demo [Seite 2077]
- SAPXmit class [Seite 1837]
- SAPForms Form Plug-In for MS Outlook [Seite 1796]
- SAPXmit class [Seite 2126]
- SAPforms Transmit Plug-In for RFC [Seite 2125]
- SAPXmit class [Seite 2168]
- SAPforms Transmit Plug-In for MAPI [Seite 2146]
- SAPXmit class [Seite 2147]
- SAPforms Transmit Plug-In for SAPMAPI [Seite 2167]
- SAPXmit class [Seite 2189]
- SAPforms Transmit Plug-In for SMTP [Seite 2188]
- SAPXmit class [Seite 2209]
- SAPforms Transmit Plug-In for Demo [Seite 2208]
**ProfileName (Property)**

**Definition**
Name of the MAPI profile to be used for transmission.

**Remarks**
You should specify this name if more than one profile is installed on the computer, since otherwise the profile selection dialog box could be displayed.

**Classification**

- SAPXmit class [Seite 2168]
- SAPforms Transmit Plug-In for MAPI [Seite 2146]
- SAPXmit class [Seite 2147]
- SAPforms Transmit Plug-In for SAPMAPI [Seite 2167]
Save (Method)

Definition
Saves the updated form definition.

Remarks
This method is used by the SAPforms Designer when a form is designed to save the modified form definition after the binding has been defined. The modifications can affect the binding definition or possibly the source code entered in the form.

Classification 1

Save

Properties
Saves the current property values of an SAPData object in a structured string. Counterpart of the Load method.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Tgt</td>
<td>String in which the status is stored.</td>
</tr>
</tbody>
</table>

Classification 2

Save

Properties
Saves the current property values of an SAPData object in a structured string. Counterpart of the Load method. The string is saved with a form so that the structure information is available when the form is executed.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Tgt</td>
<td>Stores the string structure.</td>
</tr>
</tbody>
</table>
Classification 3
Save

Properties
Loads the property values of a fields object from a structured string. As a result, the status of the fields object that was previously saved permanently with the Save method is restored.
The string is saved with the form so that the structure information is available when the form is executed.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Tgt</td>
<td>Stores the string structure.</td>
</tr>
</tbody>
</table>

Classification 4
Save

Properties
This method saves the structure of a segment, including the field list, in a structured string.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Tgt</td>
<td>Stores the string structure.</td>
</tr>
</tbody>
</table>

Classification 5
Save

Properties
This method saves the segment list in a structured string. Counterpart of the Load method.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Tgt</td>
<td>Stores the string structure.</td>
</tr>
</tbody>
</table>

Classification 6
Save
Properties
Method that writes the content of all links to a file. Counterpart of the Load method.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Tgt</td>
<td>Stores the string structure.</td>
</tr>
</tbody>
</table>

Classification 7
Save

`SAPForm class [Seite 1772]`

`SAPforms Form Plug-In for Visual Basic [Seite 1756]`

`SAPForm class [Seite 1872]`

`SAPforms Form Plug-In for MS Office [Seite 1856]`

`SAPForm class [Seite 1812]`

`SAPforms Form Plug-In for MS Outlook [Seite 1796]`

Properties
Method that saves a form to a file if it was created from a file.
Reads the old file and writes updated lines to the new file.

Classification 8
Save

`SAPXmit class [Seite 2106]`

`SAPforms Data Plug-In for Demo [Seite 2077]`

`SAPXmit class [Seite 1837]`

`SAPforms Form Plug-In for Outlook [Seite 1796]`

`SAPXmit class [Seite 2126]`

`SAPforms Transmit Plug-In for RFC [Seite 2125]`

`SAPXmit class [Seite 2168]`

`SAPforms Transmit Plug-In for MAPI [Seite 2146]`

`SAPXmit class [Seite 2147]`

`SAPforms Transmit Plug-In for SAPMAPI [Seite 2167]`

`SAPXmit class [Seite 2189]`

`SAPforms Transmit Plug-In for WinSock [Seite 2188]`

`SAPXmit class [Seite 2209]`

`SAPforms Transmit Plug-In for Demo [Seite 2208]`

Properties
Saves the current property values of an SAPXmit object in a structured string. Counterpart of the Load method.
When a form is created, the SAPforms Designer uses this method to save the structure description retrieved from the R/3 System permanently in the form.
The R/3 structure is restored from the structured string at runtime by the Load method.
<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Tgt</td>
<td>String in which the status is stored.</td>
</tr>
</tbody>
</table>
Start (Method)

Definition
Method that executes the Start function in R/3. The semantics of this function depend on the concrete SAPData object.

Remarks
In the case of the SAPData object for workflow containers, this method starts a workflow and populates its container elements with values from the form fields. This method can be called directly via the SAPData object or indirectly via the Start method of the SAPForm object, whose SAPData property is a reference to the SAPData object. If an SAPXmit object is assigned to the SAPData object, the Start method of the SAPXmit object is called.

Classification 1
Start

Properties
Starts a workflow in the R/3 System and supplies its container elements with the data values in the fields list.

Classification 2
Start

Properties
This method carries out the current R/3 function module call.

Classification 3
Start

Properties
Generates an IDoc file from the data in the segments list and transfers this IDoc to the R/3 System. The transfer method depends on the assigned SAPXmit object. A workflow can be started in the R/3 System when the inbound IDoc is processed.

Classification 4
Start

Properties

Properties
Does not execute an action in the R/3 System but rather displays the data of the object in a separate window.

Classification 5
Start

- SAPForm class [Seite 1772]
  - SAPforms Form Plug-In for Visual Basic [Seite 1756]
- SAPForm class [Seite 1872]
  - SAPforms Form Plug-In for MS Office [Seite 1856]
- SAPForm class [Seite 1812]
  - SAPforms Form Plug-In for MS Outlook [Seite 1796]

Properties
Invokes the Start method in the associated SAPXmit or SAPData object. The semantics of this method depend on this SAPData object. If this object is the SAPData object for SAP Business Workflow, this method starts a workflow in an R/3 System.

Classification 6
Start

- SAPXmit class [Seite 2126]
  - SAPforms Transmit Plug-In for RFC [Seite 2125]

Properties
This method starts a workflow synchronously via RFC. Supports SAPData objects of types "Idoc" and "Container".

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Data</td>
<td>Object of type SAPData.</td>
</tr>
</tbody>
</table>

Classification 7
Start

- SAPXmit class [Seite 2168]
  - SAPforms Transmit Plug-In for MAPI [Seite 2146]
- SAPXmit class [Seite 2147]
  - SAPforms Transmit Plug-In for SAPMAPI [Seite 2167]
- SAPXmit class [Seite 2189]
  - SAPforms Transmit Plug-In for SMTP [Seite 2188]
- SAPXmit class [Seite 1837]
  - SAPforms Form Plug-In for MS Outlook [Seite 1796]

Properties
Sends a mail to the R/3 System with an attachment containing the information required to start a workflow.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Data</td>
<td>Object of type SAPData.</td>
</tr>
</tbody>
</table>
Classification 8

Start

- SAPXmit class [Seite 2209]
- SAPforms Transmit Plug-In for Demo [Seite 2208]
- SAPXmit class [Seite 2106]
- SAPforms Data Plug-In for Demo [Seite 2077]

Properties
Displays the instance data of the associated SAPData object in a window instead of sending it to the R/3 System.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Data</td>
<td>Object of type SAPData.</td>
</tr>
</tbody>
</table>
Subject (Property)

Definition
“Subject” property of the message sent to the R/3 System by mail.

Remarks
If you do not specify a subject explicitly, the system chooses a subject that can be read by this property after the mail has been sent.

Classification
Subject

← SAPXmit class [Seite 2168]
← SAPforms Transmit Plug-In for MAPI [Seite 2146]
← SAPXmit class [Seite 2147]
← SAPforms Transmit Plug-In for SAPMAPI [Seite 2167]
← SAPXmit class [Seite 2189]
← SAPforms Transmit Plug-In for SMTP [Seite 2188]
← SAPXmit class [Seite 1837]
← SAPforms Form Plug-In for MS Outlook [Seite 1796]
**SAPforms Transmit Plug-In for SAPMAPI**

**Definition**
SAPXmit type communication plug-in for communication via SAPMAPI.

**Use**
You use this plug-in if the SAPMAPI service provider is installed on the user’s PC. This plug-in can only be used with the SAPData plug-in for workflow.
When you use the SAPMAPI service provider, the worklist is displayed in the client’s inbox directly. When you double-click a work item, the *Outlook form for processing work items* (supplied by SAP) is launched. You can then process a Visual Basic form you have already created by choosing *Execute*. This form must be stored in the directory `...\SAPforms\Forms` or in the active directory.
See also *SAPforms Helper [Seite 1741]*
This plug-in implements the SAPXmit interface through which other SAPForm and SAPData objects are enabled to transmit data to R/3. In other words, this plug-in can be regarded as being synonymous with the associated SAPXmit class.

**Classification**
*SAPforms Transmit Plug-In for SAPMAPI*

> [SAPXmit class [Seite 2168]]

**Properties**
You can maintain the following properties for this plug-in:

*Use Default MAPI Profile:*

The profile of the MAPI client determines which services are active and contain Internet-capable mail services. If you select this indicator, the system will not ask you to specify a profile when you execute the form.
SAP AG
SAPXmit Class

SAPXmit Class
Definition
Original class of the SAPforms Transmit Plug-In for SAPMAPI.

Remarks
This class implements the SAPXmit interface through which other SAPForm and SAPData objects are
enabled to transmit data to R/3. In other words, these classes can be regarded as being synonymous with
various plug-ins of this type. Communication between the form and the R/3 System be either synchronous
(via RFC) or asynchronous (via mail). The class supplies methods for communication via SAPMAPI.
If you want to develop your own SAPXmit-type plug-ins, you should use the source code supplied for the
SAPforms Transmit Plug-Ins for Demo as a reference (template). The template is stored in the directory
...\sapforms\sapxdemo.
SAPforms Transmit Plug-In for SAPMAPI [Seite 2167]
àSAPXmit Class
àExecute (Method) [Seite 2210]
àLoad (Method) [Seite 2213]
àMailItem (Property) [Seite 2176]
àOptions (Method) [Seite 2217]
àProfileName (Property) [Seite 2179]
àSave (Method) [Seite 2219]
àStart (Method) [Seite 2223]
àSubject (Property) [Seite 2207]

April 2001

2168


Execute (Method)

Definition
Method that calls the *Execute* function in the R/3 System. The semantics of this function depend on the SAPData object.

Remarks
The method can be called directly via the SAPData object or indirectly via the Start method of the SAPForm object, whose SAPData property is a reference to the SAPData object.

In the case of the SAPData object for workflow containers, for example, this method executes a work item in an SAP Business Workflow and populates its container elements with values from the form fields.

Classification 1
Execute

- SAPData class [Seite 1918]
- SAPforms Data Plug-In for Workflow [Seite 1897]

Properties
Terminates a work item in the R/3 System and supplies its container elements with the data values in the fields list.

Classification 2
Execute

- SAPData class [Seite 2078]
- SAPXmit class [Seite 2106]

Properties
Does not execute an action in the R/3 System but rather displays the data of the object in a separate window.

Classification 3
Execute

- SAPForm class [Seite 1772]
- SAPforms Form Plug-In for Visual Basic [Seite 1756]
- SAPForm class [Seite 1872]
- SAPforms Form Plug-In for MS Office [Seite 1856]
- SAPForm class [Seite 1812]

Properties
Invokes the Execute method in the associated Xmit or SAPData object. The semantics of this method depend on this object.
Classification 4

Execute

- SAPXmit class [Seite 2126]
- SAPforms Transmit Plug-In for RFC [Seite 2125]

Properties
This method executes a work item synchronously via remote function call (RFC). Supports SAPData object of type Container.

This method causes a certain function module to be executed in the R/3 System. This module receives a table with a complete R3F structure containing the work item information and the container data that are to be supplied from the form.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Data</td>
<td>Object representing the container data.</td>
</tr>
</tbody>
</table>

Classification 5

Execute

- SAPXmit class [Seite 2168]
- SAPforms Transmit Plug-In for MAPI [Seite 2146]
- SAPXmit class [Seite 2189]
- SAPforms Transmit Plug-In for WinSock [Seite 2188]

Properties
Sends a mail to the R/3 System that contains the information required to terminate a work item (including container data) as an attachment.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Data</td>
<td>Object representing the container data.</td>
</tr>
</tbody>
</table>

Classification 6

Execute

- SAPXmit class [Seite 2147]
- SAPforms Transmit Plug-In for SAPMAPI [Seite 2167]

Properties
Overwrites the R3F attachment, thereby updating it.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Pdata</td>
<td>Object that represents the mobjData container data</td>
</tr>
</tbody>
</table>

Classification 7

Execute

- SAPXmit class [Seite 2209]
- SAPforms Transmit Plug-In for Demo [Seite 2208]

Properties
Displays the instance data of the associated SAPData object in a window instead of sending it to the R/3 System.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Data</td>
<td>object of class SAPData</td>
</tr>
</tbody>
</table>
Load (Method)

Definition
Loads structure information to SAPforms objects.

Remarks

Classification 1

Load

Properties
Loads the property values of an SAPData object from a structured string. As a result, the status of the SAPData object that was previously saved permanently with the Save method is restored. This saves the additional overheads that would result if the structure information had to be loaded each time from the original source. This method is usually called within the Load method of the SAPForm class.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>From</td>
<td>Character string containing the structure information</td>
</tr>
</tbody>
</table>

Classification 2

Load

Properties
Loads the property values of a field object from a structured string. As a result, the status of the field object that was previously saved permanently with the Save method is restored. For this method to be called successfully, the next character at the Pos position must be keyword “Fld”.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>From</td>
<td>Structured string</td>
</tr>
<tr>
<td>Pos</td>
<td>Current position within the string. Should point to the number of remaining properties to read</td>
</tr>
</tbody>
</table>

Classification 3

Load
Load (Method)

Properties
Loads the property values of a fields object from a structured string. As a result, the status of the fields object that was previously saved permanently with the Save method is restored.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>From</td>
<td>Character string that provides the structure information.</td>
</tr>
<tr>
<td>Pos</td>
<td>Current search position within the character string.</td>
</tr>
</tbody>
</table>

Classification 4
Load

Properties
Reconstructs the segment object from a structured string. Counterpart of the Save method.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>From</td>
<td>Character string containing structure information.</td>
</tr>
<tr>
<td>Pos</td>
<td>Current search position within the character string.</td>
</tr>
</tbody>
</table>

Classification 5
Load

Properties
This method reconstructs the segment list from a structured string. Counterpart of the Save method.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>From</td>
<td>Character string containing structure information.</td>
</tr>
<tr>
<td>Pos</td>
<td>Current search position within the character string.</td>
</tr>
</tbody>
</table>

Classification 6
Load

Properties
This method reconstructs an object of the Link class from the serialized representation. It assumes that the next token is located at the Pos position in the serialized representation of the “Fld” keyword.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>From</td>
<td>Serialization string.</td>
</tr>
</tbody>
</table>
Load (Method)

Classification 7
Load

Properties
Method for loading form controls into Controls Collection from a form or form template. Note that the form information is not stored in the Tag property of this object, but reconstructed each time the form is loaded. This method is called at design time and at runtime. It has to populate the Controls collection for both cases. At runtime, the method is called by the user code in the form's Load method. When the form is designed, the method is called by the SAPForms Designer or from outside SAPForms. If you are using the SAPForms Designer as a stand-alone application, the Load method is called with Mode=2. This instructs the Load method to load the form from a file. If you are using the SAPForms Designer as a plug-in, the Load method is invoked from the hosting environment and receives the proprietary form description as a parameter. The populated SAPForm object should be passed to the SAPForms Designer by the PlugIn method of the Designer class.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Frm</td>
<td>form or form template containing controls.</td>
</tr>
<tr>
<td>Mode</td>
<td>0=design time (add-in), 1=design time (file), 2=runtime.</td>
</tr>
</tbody>
</table>

Classification 8
Load

Properties
**Properties**

Loads the property values of an SAPXmit object from a structured string. As a result, the status of the SAPXmit object that was previously saved permanently with the Save method is restored.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>From</td>
<td>Character string containing the structure information</td>
</tr>
</tbody>
</table>
MailItem (Property)

**Definition**
Reference to the mail that is used to transport the form data.

**Remarks**
The property must be populated “externally” before the *Start or Execute* method is executed.

**Classification**

- SAPXmit class [Seite 1837]
- SAPforms Form Plug-In for MS Outlook [Seite 1796]
- SAPXmit class [Seite 2168]
- SAPforms Transmit Plug-In for MAPI [Seite 2146]
- SAPXmit class [Seite 2147]
- SAPforms Transmit Plug-In for SAPMAPI [Seite 2167]
Options (Method)

Definition
Method that outputs the object-specific options dialog box.

Remarks
This dialog box displays either global settings or settings that are valid for the current form. These settings can be changed here. The global settings are read from the registry before they are displayed and are then saved again to the registry.

The method is called within the SAPforms Designer if the required plug-in is active under Properties on the Plug-Ins tab in the Options dialog. You can also use the sample program Options.exe to display the Options dialog of a certain plug-in.

Classification 1
Options

- SAPData class [Seite 1918]
- SAPforms Data Plug-In for Workflow [Seite 1897]
- SAPData class [Seite 1975]
- SAPforms Data Plug-In for Function Modules [Seite 1954]
- SAPData class [Seite 2027]
- SAPforms Data Plug-In for IDoc [Seite 2006]
- SAPData class [Seite 2078]
- SAPforms Data Plug-In for Demo [Seite 2077]

Classification 2
Options

- SAPForm class [Seite 1772]
- SAPforms Form Plug-In for Visual Basic [Seite 1756]
- SAPForm class [Seite 1812]
- SAPforms Form Plug-In for MS Outlook [Seite 1796]
- SAPForm class [Seite 1872]
- SAPforms Form Plug-In for MS Office [Seite 1856]

Classification 3
Options

- SAPXmit class [Seite 2106]
- SAPforms Data Plug-In for Demo [Seite 2077]
- SAPXmit class [Seite 1837]
- SAPforms Form Plug-In for MS Outlook [Seite 1796]
- SAPXmit class [Seite 2126]
Options (Method)

- SAPforms Transmit Plug-In for RFC [Seite 2125]
- SAPXmit class [Seite 2168]
- SAPforms Transmit Plug-In for MAPI [Seite 2146]
- SAPXmit class [Seite 2147]
- SAPforms Transmit Plug-In for SAPMAPI [Seite 2167]
- SAPXmit class [Seite 2189]
- SAPforms Transmit Plug-In for SMTP [Seite 2188]
- SAPXmit class [Seite 2209]
- SAPforms Transmit Plug-In for Demo [Seite 2208]
ProfileName (Property)

Definition
Name of the MAPI profile to be used for transmission.

Remarks
You should specify this name if more than one profile is installed on the computer, since otherwise the profile selection dialog box could be displayed.

Classification
ProfileName

← SAPXmit class [Seite 2168]
← SAPforms Transmit Plug-In for MAPI [Seite 2146]
← SAPXmit class [Seite 2147]
← SAPforms Transmit Plug-In for SAPMAPI [Seite 2167]
Save (Method)

Definition
Saves the updated form definition.

Remarks
This method is used by the SAPforms Designer when a form is designed to save the modified form definition after the binding has been defined. The modifications can affect the binding definition or possibly the source code entered in the form.

Classification 1
Save

Properties
Saves the current property values of an SAPData object in a structured string. Counterpart of the Load method. When a form is created, the SAPforms Designer uses this method to save the structure description retrieved from the R/3 System permanently in the form. The R/3 structure is restored from the structured string at runtime by the Load method.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Tgt</td>
<td>String in which the status is stored.</td>
</tr>
</tbody>
</table>

Classification 2
Save

Properties
Saves the current property values of an SAPData object in a structured string. Counterpart of the Load method. The string is saved with a form so that the structure information is available when the form is executed.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Tgt</td>
<td>Stores the string structure.</td>
</tr>
</tbody>
</table>
Classification 3

**Save**

- **Fields Class [Seite 2018]**
- **SAPforms Data Plug-In for Workflow [Seite 1897]**
- **SAPforms Data Plug-In for Function Modules [Seite 1954]**
- **SAPforms Data Plug-In for IDoc [Seite 2006]**

**Properties**

Loads the property values of a fields object from a structured string. As a result, the status of the fields object that was previously saved permanently with the Save method is restored.

The string is saved with the form so that the structure information is available when the form is executed.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Tgt</td>
<td>Stores the string structure.</td>
</tr>
</tbody>
</table>

Classification 4

**Save**

- **Segment class [Seite 2055]**
- **SAPforms Data Plug-In for IDoc [Seite 2006]**

**Properties**

This method saves the structure of a segment, including the field list, in a structured string.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Tgt</td>
<td>Stores the string structure.</td>
</tr>
</tbody>
</table>

Classification 5

**Save**

- **Segments class [Seite 2067]**
- **SAPforms Data Plug-In for IDoc [Seite 2006]**

**Properties**

This method saves the segment list in a structured string. Counterpart of the Load method.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Tgt</td>
<td>Stores the string structure.</td>
</tr>
</tbody>
</table>

Classification 6

**Save**

- **Links class [Seite 1862]**
- **SAPforms Form Plug-In for Visual Basic [Seite 1756]**
- **Links class [Seite 1862]**
- **SAPforms Form Plug-In for MS Office [Seite 1856]**
- **Links class [Seite 1862]**
- **SAPforms Form Plug-In for MS Outlook [Seite 1796]**)
Properties
Method that writes the content of all links to a file. Counterpart of the Load method.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Tgt</td>
<td>Stores the string structure.</td>
</tr>
</tbody>
</table>

Classification 7

Save

- SAPForm class [Seite 1772]
- SAPforms Form Plug-In for Visual Basic [Seite 1756]
- SAPForm class [Seite 1872]
- SAPforms Form Plug-In for MS Office [Seite 1856]
- SAPForm class [Seite 1812]
- SAPforms Form Plug-In for MS Outlook [Seite 1796]

Properties
Method that saves a form to a file if it was created from a file.
Reads the old file and writes updated lines to the new file.

Classification 8

Save

- SAPXmit class [Seite 2106]
- SAPforms Data Plug-In for Demo [Seite 2077]
- SAPXmit class [Seite 1837]
- SAPforms Form Plug-In for Outlook [Seite 1796]
- SAPXmit class [Seite 2126]
- SAPforms Transmit Plug-In for RFC [Seite 2125]
- SAPXmit class [Seite 2168]
- SAPforms Transmit Plug-In for MAPI [Seite 2146]
- SAPXmit class [Seite 2147]
- SAPforms Transmit Plug-In for SAPMAPI [Seite 2167]
- SAPXmit class [Seite 2189]
- SAPforms Transmit Plug-In for WinSock [Seite 2188]
- SAPXmit class [Seite 2209]
- SAPforms Transmit Plug-In for Demo [Seite 2208]

Properties
Saves the current property values of an SAPXmit object in a structured string. Counterpart of the Load method.
When a form is created, the SAPforms Designer uses this method to save the structure description retrieved from the R/3 System permanently in the form.
The R/3 structure is restored from the structured string at runtime by the Load method.
<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Tgt</td>
<td>String in which the status is stored.</td>
</tr>
</tbody>
</table>
Start (Method)

Definition
Method that executes the Start function in R/3. The semantics of this function depend on the concrete SAPData object.

Remarks
In the case of the SAPData object for workflow containers, this method starts a workflow and populates its container elements with values from the form fields. This method can be called directly via the SAPData object or indirectly via the Start method of the SAPForm object, whose SAPData property is a reference to the SAPData object. If an SAPXmit object is assigned to the SAPData object, the Start method of the SAPXmit object is called.

Classification 1
Start

Properties
Starts a workflow in the R/3 System and supplies its container elements with the data values in the fields list.

Classification 2
Start

Properties
This method carries out the current R/3 function module call.

Classification 3
Start

Properties
Generates an IDoc file from the data in the segments list and transfers this IDoc to the R/3 System. The transfer method depends on the assigned SAPXmit object. A workflow can be started in the R/3 System when the inbound IDoc is processed.

Classification 4
Start

Properties

April 2001
Properties
Does not execute an action in the R/3 System but rather displays the data of the object in a separate window.

Classification 5
Start

Properties
Invokes the Start method in the associated SAPXmit or SAPData object. The semantics of this method depend on this SAPData object. If this object is the SAPData object for SAP Business Workflow, this method starts a workflow in an R/3 System.

Classification 6
Start

Properties
This method starts a workflow synchronously via RFC. Supports SAPData objects of types "Idoc" and "Container".

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Data</td>
<td>Object of type SAPData.</td>
</tr>
</tbody>
</table>

Classification 7
Start

Properties
Sends a mail to the R/3 System with an attachment containing the information required to start a workflow.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Data</td>
<td>Object of type SAPData.</td>
</tr>
</tbody>
</table>
Classification 8

Start

 asserts SAPXmit class [Seite 2209]
 asserts SAPforms Transmit Plug-In for Demo [Seite 2208]
 asserts SAPXmit class [Seite 2106]
 asserts SAPforms Data Plug-In for Demo [Seite 2077]

Properties

Displays the instance data of the associated SAPData object in a window instead of sending it to the R/3 System.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Data</td>
<td>Object of type SAPData.</td>
</tr>
</tbody>
</table>
Subject (Property)

Definition
“Subject” property of the message sent to the R/3 System by mail.

Remarks
If you do not specify a subject explicitly, the system chooses a subject that can be read by this property after the mail has been sent.

Classification
Subject

- SAPXmit class [Seite 2168]
- SAPforms Transmit Plug-In for MAPI [Seite 2146]
- SAPXmit class [Seite 2147]
- SAPforms Transmit Plug-In for SAPMAPI [Seite 2167]
- SAPXmit class [Seite 2189]
- SAPforms Transmit Plug-In for SMTP [Seite 2188]
- SAPXmit class [Seite 1837]
- SAPforms Form Plug-In for MS Outlook [Seite 1796]
SAPforms Transmit Plug-In for SMTP via WinSock

Definition
SAPXmit type communication plug-in for asynchronous communication with Internet mail.

Use
You use this plug-in if a MAPI client is not installed and you want to address the mail server directly. This plug-in can only be used with the SAPData plug-in for workflow. This plug-in implements the SAPXmit interface through which other SAPForm and SAPData objects are enabled to transmit data to R/3. In other words, this plug-in can be regarded as being synonymous with the associated SAPXmit class.

Classification
SAPforms Transmit Plug-In for SMTP via WinSock
➔ SAPXmit class [Seite 2189]

Properties
You can maintain the following properties for this plug-in:

Mail Server
Enter the name of your mail server here.

Mail User
Enter the name of the mail user.
SAPXmit Class

Definition
Original class of the SAPforms Transmit Plug-In for SMTP.

Remarks
This class implements the SAPXmit interface through which other SAPForm and SAPData objects are enabled to transmit data to R/3. Communication between the form and the R/3 System be either synchronous (via RFC) or asynchronous (via mail). The class supplies methods for communication via SMTP.

If you want to develop your own SAPXmit-type plug-ins, you should use the source code supplied for the SAPforms Transmit Plug-Ins for Demo as a reference (template). The template is stored in the directory .../sapforms/sapxdemo.

Classification
SAPforms Transmit Plug-In for SMTP via WinSock [Seite 2188]

→ SAPXmit Class
   → Execute (Method) [Seite 2210]
   → Load (Method) [Seite 2213]
   → MailServer (Property) [Seite 2197]
   → Options (Method) [Seite 2217]
   → Save (Method) [Seite 2219]
   → Start (Method) [Seite 2223]
   → Subject (Property) [Seite 2207]
Execute (Method)

Definition
Method that calls the Execute function in the R/3 System. The semantics of this function depend on the SAPData object.

Remarks
The method can be called directly via the SAPData object or indirectly via the Start method of the SAPForm object, whose SAPData property is a reference to the SAPData object.

In the case of the SAPData object for workflow containers, for example, this method executes a work item in an SAP Business Workflow and populates its container elements with values from the form fields.

Classification 1
Execute
- SAPData class [Seite 1918]
  - SAPforms Data Plug-In for Workflow [Seite 1897]

Properties
Terminates a work item in the R/3 System and supplies its container elements with the data values in the fields list.

Classification 2
Execute
- SAPData class [Seite 2078]
- SAPXmit class [Seite 2106]
  - SAPforms Data Plug-In for Demo [Seite 2077]

Properties
Does not execute an action in the R/3 System but rather displays the data of the object in a separate window.

Classification 3
Execute
- SAPForm class [Seite 1772]
  - SAPforms Form Plug-In for Visual Basic [Seite 1756]
- SAPForm class [Seite 1872]
  - SAPforms Form Plug-In for MS Office [Seite 1856]
- SAPForm class [Seite 1812]
  - SAPforms Form Plug-In for MS Outlook [Seite 1796]

Properties
Invokes the Execute method in the associated Xmit or SAPData object. The semantics of this method depend on this object.
Classification 4

Execute

- SAPXmit class [Seite 2126]
- SAPforms Transmit Plug-In for RFC [Seite 2125]

Properties

This method executes a work item synchronously via a remote function call (RFC). Supports SAPData object of type Container.

This method causes a certain function module to be executed in the R/3 System. This module receives a table with a complete R3F structure containing the work item information and the container data that are to be supplied from the form.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Data</td>
<td>Object representing the container data.</td>
</tr>
</tbody>
</table>

Classification 5

Execute

- SAPXmit class [Seite 2168]
- SAPforms Transmit Plug-In for MAPI [Seite 2146]
- SAPXmit class [Seite 2189]
- SAPforms Transmit Plug-In for WinSock [Seite 2188]

Properties

Sends a mail to the R/3 System that contains the information required to terminate a work item (including container data) as an attachment.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Data</td>
<td>Object representing the container data.</td>
</tr>
</tbody>
</table>

Classification 6

Execute

- SAPXmit class [Seite 2147]
- SAPforms Transmit Plug-In for SAPMAPI [Seite 2167]

Properties

Overwrites the R3F attachment, thereby updating it.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Pdata</td>
<td>Object that represents the mobjData container data</td>
</tr>
</tbody>
</table>

Classification 7

Execute

- SAPXmit class [Seite 2209]
- SAPforms Transmit Plug-In for Demo [Seite 2208]

Properties

Displays the instance data of the associated SAPData object in a window instead of sending it to the R/3 System.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Data</td>
<td>object of class SAPData</td>
</tr>
</tbody>
</table>
Load (Method)

Definition
Loads structure information to SAPforms objects.

Remarks

Classification 1

Load

Properties
Loads the property values of an SAPData object from a structured string. As a result, the status of the SAPData object that was previously saved permanently with the Save method is restored. This saves the additional overheads that would result if the structure information had to be loaded each time from the original source. This method is usually called within the Load method of the SAPForm class.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>From</td>
<td>Character string containing the structure information</td>
</tr>
</tbody>
</table>

Classification 2

Load

Properties
Loads the property values of a field object from a structured string. As a result, the status of the field object that was previously saved permanently with the Save method is restored. For this method to be called successfully, the next character at the Pos position must be keyword “Fld”.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>From</td>
<td>Structured string</td>
</tr>
<tr>
<td>Pos</td>
<td>Current position within the string. Should point to the number of remaining properties to read</td>
</tr>
</tbody>
</table>

Classification 3

Load
Properties
Loads the property values of a fields object from a structured string. As a result, the status of the fields object that was previously saved permanently with the Save method is restored.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>From</td>
<td>Character string that provides the structure information.</td>
</tr>
<tr>
<td>Pos</td>
<td>Current search position within the character string.</td>
</tr>
</tbody>
</table>

Classification 4
Load

Properties
Reconstructs the segment object from a structured string. Counterpart of the Save method.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>From</td>
<td>Character string containing structure information.</td>
</tr>
<tr>
<td>Pos</td>
<td>Current search position within the character string.</td>
</tr>
</tbody>
</table>

Classification 5
Load

Properties
This method reconstructs the segment list from a structured string. Counterpart of the Save method.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>From</td>
<td>Character string containing structure information.</td>
</tr>
<tr>
<td>Pos</td>
<td>Current search position within the character string.</td>
</tr>
</tbody>
</table>

Classification 6
Load

Properties
This method reconstructs an object of the Link class from the serialized representation. It assumes that the next token is located at the Pos position in the serialized representation of the “Fld” keyword.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>From</td>
<td>Serialization string.</td>
</tr>
</tbody>
</table>
Load (Method)

Current position within the character string.

Classification 7

Load

- SAPForm class [Seite 1772]
- SAPForm class [Seite 1812]
- SAPForm class [Seite 1872]

Properties

Method for loading form controls into Controls Collection from a form or form template.
Note that the form information is not stored in the Tag property of this object, but reconstructed each time
the form is loaded. This method is called at design time and at runtime. It has to populate the Controls
collection for both cases. At runtime, the method is called by the user code in the form's Load method.
When the form is designed, the method is called by the SAPForms Designer or from outside SAPForms. If
you are using the SAPForms Designer as a stand-alone application, the Load method is called with
Mode=2. This instructs the Load method to load the form from a file. If you are using the SAPForms
Designer as a plug-in, the Load method is invoked from the hosting environment and receives the
proprietary form description as a parameter. The populated SAPForm object should be passed to the
SAPForms Designer by the PlugIn method of the Designer class.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Frm</td>
<td>form or form template containing controls.</td>
</tr>
<tr>
<td>Mode</td>
<td>0=design time (add-in), 1=design time (file), 2=runtime.</td>
</tr>
</tbody>
</table>

Classification 8

Load

- SAPXmit class [Seite 2106]
- SAPXmit class [Seite 1837]
- SAPXmit class [Seite 2168]
- SAPXmit class [Seite 2147]
- SAPXmit class [Seite 2189]
- SAPXmit class [Seite 2209]
- SAPforms Data Plug-In for Demo [Seite 2077]
- SAPforms Form Plug-In for Outlook [Seite 1796]
- SAPforms Transmit Plug-In for RFC [Seite 2125]
- SAPforms Transmit Plug-In for MAPI [Seite 2146]
- SAPforms Transmit Plug-In for SAPMAPI [Seite 2167]
- SAPforms Transmit Plug-In for WinSock [Seite 2188]
- SAPforms Transmit Plug-In for Demo [Seite 2208]
Properties
Loads the property values of an SAPXmit object from a structured string. As a result, the status of the SAPXmit object that was previously saved permanently with the Save method is restored.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>From</td>
<td>Character string containing the structure information</td>
</tr>
</tbody>
</table>
**MailServer (Property)**

**Function**
Logical or IP address of the mail server that is responsible for delivering emails to the R/3 System.

**Remarks**
The form developer can set this property in the form using program code. Otherwise, the setting in the Registry is used. If the Registry does not contain this entry, the end user of the form is asked to enter the mail server in a popup window.

**Structure**
- MailServer
  - SAPXmit class [Seite 2189]
  - SAPforms Transmit Plug-In for SMTP [Seite 2188]
Options (Method)

Definition
Method that outputs the object-specific options dialog box.

Remarks
This dialog box displays either global settings or settings that are valid for the current form. These settings can be changed here. The global settings are read from the registry before they are displayed and are then saved again to the registry.
The method is called within the SAPforms Designer if the required plug-in is active under Properties on the Plug-Ins tab in the Options dialog. You can also use the sample program Options.exe to display the Options dialog of a certain plug-in.

Classification 1
Options

- SAPData class [Seite 1918]
  - SAPforms Data Plug-In for Workflow [Seite 1897]
- SAPData class [Seite 1975]
  - SAPforms Data Plug-In for Function Modules [Seite 1954]
- SAPData class [Seite 2027]
  - SAPforms Data Plug-In for IDoc [Seite 2006]
- SAPData class [Seite 2078]
  - SAPforms Data Plug-In for Demo [Seite 2077]

Classification 2
Options

- SAPForm class [Seite 1772]
  - SAPforms Form Plug-In for Visual Basic [Seite 1756]
- SAPForm class [Seite 1812]
  - SAPforms Form Plug-In for MS Outlook [Seite 1796]
- SAPForm class [Seite 1872]
  - SAPforms Form Plug-In for MS Office [Seite 1856]

Classification 3
Options

- SAPXmit class [Seite 2106]
  - SAPforms Data Plug-In for Demo [Seite 2077]
- SAPXmit class [Seite 1837]
  - SAPforms Form Plug-In for MS Outlook [Seite 1796]
Options (Method)

- SAPforms Transmit Plug-In for RFC [Seite 2125]
- SAPXmit class [Seite 2168]
- SAPforms Transmit Plug-In for MAPI [Seite 2146]
- SAPXmit class [Seite 2147]
- SAPforms Transmit Plug-In for SAPMAPI [Seite 2167]
- SAPXmit class [Seite 2189]
- SAPforms Transmit Plug-In for SMTP [Seite 2188]
- SAPXmit class [Seite 2209]
- SAPforms Transmit Plug-In for Demo [Seite 2208]
Save (Method)

Definition
Saves the updated form definition.

Remarks
This method is used by the SAPforms Designer when a form is designed to save the modified form
definition after the binding has been defined. The modifications can affect the binding definition or
possibly the source code entered in the form.

Classification 1
Save

Properties
Saves the current property values of an SAPData object in a structured string. Counterpart of the Load
method.
When a form is created, the SAPforms Designer uses this method to save the structure description retrieved
from the R/3 System permanently in the form.
The R/3 structure is restored from the structured string at runtime by the Load method.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Tgt</td>
<td>String in which the status is stored.</td>
</tr>
</tbody>
</table>

Classification 2
Save

Properties
Saves the current property values of an SAPData object in a structured string. Counterpart of the Load
method. The string is saved with a form so that the structure information is available when the form is
executed.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Tgt</td>
<td>Stores the string structure.</td>
</tr>
</tbody>
</table>
Classification 3

Save

Properties

Loads the property values of a fields object from a structured string. As a result, the status of the fields object that was previously saved permanently with the Save method is restored. The string is saved with the form so that the structure information is available when the form is executed.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Tgt</td>
<td>Stores the string structure.</td>
</tr>
</tbody>
</table>

Classification 4

Save

Properties

This method saves the structure of a segment, including the field list, in a structured string.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Tgt</td>
<td>Stores the string structure.</td>
</tr>
</tbody>
</table>

Classification 5

Save

Properties

This method saves the segment list in a structured string. Counterpart of the Load method.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Tgt</td>
<td>Stores the string structure.</td>
</tr>
</tbody>
</table>

Classification 6

Save

Properties
Properties
Method that writes the content of all links to a file. Counterpart of the Load method.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Tgt</td>
<td>Stores the string structure.</td>
</tr>
</tbody>
</table>

Classification 7

Save

- SAPForm class [Seite 1772]
- SAPforms Form Plug-In for Visual Basic [Seite 1756]
- SAPForm class [Seite 1872]
- SAPforms Form Plug-In for MS Office [Seite 1856]
- SAPForm class [Seite 1812]
- SAPforms Form Plug-In for MS Outlook [Seite 1796]

Properties
Method that saves a form to a file if it was created from a file. Reads the old file and writes updated lines to the new file.

Classification 8

Save

- SAPXmit class [Seite 2106]
- SAPforms Data Plug-In for Demo [Seite 2077]
- SAPXmit class [Seite 1837]
- SAPforms Form Plug-In for Outlook [Seite 1796]
- SAPXmit class [Seite 2126]
- SAPforms Transmit Plug-In for RFC [Seite 2125]
- SAPXmit class [Seite 2168]
- SAPforms Transmit Plug-In for MAPI [Seite 2146]
- SAPXmit class [Seite 2147]
- SAPforms Transmit Plug-In for SAPMAPI [Seite 2167]
- SAPXmit class [Seite 2189]
- SAPforms Transmit Plug-In for WinSock [Seite 2188]
- SAPXmit class [Seite 2209]
- SAPforms Transmit Plug-In for Demo [Seite 2208]

Properties
Saves the current property values of an SAPXmit object in a structured string. Counterpart of the Load method.
When a form is created, the SAPforms Designer uses this method to save the structure description retrieved from the R/3 System permanently in the form.
The R/3 structure is restored from the structured string at runtime by the Load method.
## Save (Method)

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Tgt</td>
<td>String in which the status is stored.</td>
</tr>
</tbody>
</table>
Start (Method)

Definition
Method that executes the Start function in R/3. The semantics of this function depend on the concrete SAPData object.

Remarks
In the case of the SAPData object for workflow containers, this method starts a workflow and populates its container elements with values from the form fields. This method can be called directly via the SAPData object or indirectly via the Start method of the SAPForm object, whose SAPData property is a reference to the SAPData object. If an SAPXmit object is assigned to the SAPData object, the Start method of the SAPXmit object is called.

Classification 1
Start

Properties
Starts a workflow in the R/3 System and supplies its container elements with the data values in the fields list.

Classification 2
Start

Properties
This method carries out the current R/3 function module call.

Classification 3
Start

Properties
Generates an IDoc file from the data in the segments list and transfers this IDoc to the R/3 System. The transfer method depends on the assigned SAPXmit object. A workflow can be started in the R/3 System when the inbound IDoc is processed.

Classification 4
Start
Properties
Does not execute an action in the R/3 System but rather displays the data of the object in a separate window.

Classification 5
Start

Properties
Invokes the Start method in the associated SAPXmit or SAPData object. The semantics of this method depend on this SAPData object. If this object is the SAPData object for SAP Business Workflow, this method starts a workflow in an R/3 System.

Classification 6
Start

Properties
This method starts a workflow synchronously via RFC. Supports SAPData objects of types "Idoc" and "Container".

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Data</td>
<td>Object of type SAPData.</td>
</tr>
</tbody>
</table>

Classification 7
Start

Properties
Sends a mail to the R/3 System with an attachment containing the information required to start a workflow.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Data</td>
<td>Object of type SAPData.</td>
</tr>
</tbody>
</table>
Classification 8

Start

Start

- SAPXmit class [Seite 2209]
- SAPforms Transmit Plug-In for Demo [Seite 2208]
- SAPXmit class [Seite 2106]
- SAPforms Data Plug-In for Demo [Seite 2077]

Properties

Displays the instance data of the associated SAPData object in a window instead of sending it to the R/3 System.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Data</td>
<td>Object of type SAPData.</td>
</tr>
</tbody>
</table>
Subject (Property)

Definition
“Subject” property of the message sent to the R/3 System by mail.

Remarks
If you do not specify a subject explicitly, the system chooses a subject that can be read by this property after the mail has been sent.

Classification
Subject

← SAPXmit class [Seite 2168]
← SAPforms Transmit Plug-In for MAPI [Seite 2146]
← SAPXmit class [Seite 2147]
← SAPforms Transmit Plug-In for SAPMAPI [Seite 2167]
← SAPXmit class [Seite 2189]
← SAPforms Transmit Plug-In for SMTP [Seite 2188]
← SAPXmit class [Seite 1837]
← SAPforms Form Plug-In for MS Outlook [Seite 1796]
SAPforms Transmit Plug-In Demo

Definition
SAPXmit type communication plug-in for demonstration purposes.

Use
This communication plug-in is used in the introductory example (Getting Started [Seite 1739]). This plug-in must be used with the SAPData Plug-In for Demo. This plug-in implements the SAPXmit interface through which other SAPForm and SAPData objects are enabled to transmit data to R/3. In other words, this plug-in can be regarded as being synonymous with the associated SAPXmit class.

Classification
SAPforms Transmit Plug-In Demo

→ SAPXmit class [Seite 2209]
SAPXmit Class

Definition
Original class of the SAPforms Transmit Plug-In for Demo.

Remarks
This demo implementation of the SAPXmit interface displays the data of the SAPData object.
If you want to develop your own SAPXmit-type plug-ins, you should use the source code supplied for the SAPforms Transmit Plug-Ins for Demo as a reference (template). The template is stored in the directory ...\sapforms\sapxdemo.

Classification
SAPforms Transmit Plug-In Demo [Seite 2125]

→ SAPXmit Class
  → Execute (Method) [Seite 2210]
  → Load (Method) [Seite 2213]
  → Options (Method) [Seite 2217]
  → Save (Method) [Seite 2219]
  → Start (Method) [Seite 2223]
Execution (Method)

Definition
Method that calls the Execute function in the R/3 System. The semantics of this function depend on the SAPData object.

Remarks
The method can be called directly via the SAPData object or indirectly via the Start method of the SAPForm object, whose SAPData property is a reference to the SAPData object.

Classification 1
Execute

Properties
Terminates a work item in the R/3 System and supplies its container elements with the data values in the fields list.

Classification 2
Execute

Properties
Does not execute an action in the R/3 System but rather displays the data of the object in a separate window.

Classification 3
Execute
Properties
Invokes the Execute method in the associated Xmit or SAPData object. The semantics of this method depend on this object.

Classification 4
Execute

Properties
This method executes a work item synchronously via a remote function call (RFC). Supports SAPData object of type Container.
This method causes a certain function module to be executed in the R/3 System. This module receives a table with a complete R3F structure containing the work item information and the container data that are to be supplied from the form.

Parameter | Description
---|---
Data | Object representing the container data.

Classification 5
Execute

Properties
Sends a mail to the R/3 System that contains the information required to terminate a work item (including container data) as an attachment.

Parameter | Description
---|---
Data | Object representing the container data.

Classification 6
Execute

Properties
Overwrites the R3F attachment, thereby updating it.

Parameter | Description
---|---
Pdata | Object that represents the mobjData container data

Classification 7
Execute

Properties


SAPXmit class [Seite 2126]
SAPforms Transmit Plug-In for RFC [Seite 2125]
SAPXmit class [Seite 2209]
SAPforms Transmit Plug-In for Demo [Seite 2208]
Properties
Displays the instance data of the associated SAPData object in a window instead of sending it to the R/3 System.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Data</td>
<td>object of class SAPData</td>
</tr>
</tbody>
</table>
Load (Method)

Definition
Loads structure information to SAPforms objects.

Remarks
Classification 1
Load

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>From</td>
<td>Character string containing the structure information</td>
</tr>
</tbody>
</table>

Properties
Loads the property values of an SAPData object from a structured string. As a result, the status of the SAPData object that was previously saved permanently with the Save method is restored. This saves the additional overheads that would result if the structure information had to be loaded each time from the original source. This method is usually called within the Load method of the SAPForm class.

Classification 2
Load

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>From</td>
<td>Structured string</td>
</tr>
<tr>
<td>Pos</td>
<td>Current position within the string. Should point to the number of remaining properties to read</td>
</tr>
</tbody>
</table>

Properties
Loads the property values of a field object from a structured string. As a result, the status of the field object that was previously saved permanently with the Save method is restored. For this method to be called successfully, the next character at the Pos position must be keyword “Fld”.

Classification 3
Load
Properties
Loads the property values of a fields object from a structured string. As a result, the status of the fields object that was previously saved permanently with the Save method is restored.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>From</td>
<td>Character string that provides the structure information.</td>
</tr>
<tr>
<td>Pos</td>
<td>Current search position within the character string.</td>
</tr>
</tbody>
</table>

Classification 4
Load

Properties
Reconstructs the segment object from a structured string. Counterpart of the Save method.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>From</td>
<td>Character string containing structure information.</td>
</tr>
<tr>
<td>Pos</td>
<td>Current search position within the character string.</td>
</tr>
</tbody>
</table>

Classification 5
Load

Properties
This method reconstructs the segment list from a structured string. Counterpart of the Save method.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>From</td>
<td>Character string containing structure information.</td>
</tr>
<tr>
<td>Pos</td>
<td>Current search position within the character string.</td>
</tr>
</tbody>
</table>

Classification 6
Load

Properties
This method reconstructs an object of the Link class from the serialized representation. It assumes that the next token is located at the Pos position in the serialized representation of the “Fld” keyword.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>From</td>
<td>Serialization string.</td>
</tr>
</tbody>
</table>
Classification 7
Load

- SAPForm class [Seite 1772]
- SAPforms Form Plug-In for Visual Basic [Seite 1756]
- SAPForm class [Seite 1812]
- SAPforms Form Plug-In for MS Outlook [Seite 1796]
- SAPForm class [Seite 1872]
- SAPforms Form Plug-In for MS Office [Seite 1856]

Properties
Method for loading form controls into Controls Collection from a form or form template. Note that the form information is not stored in the Tag property of this object, but reconstructed each time the form is loaded. This method is called at design time and at runtime. It has to populate the Controls collection for both cases. At runtime, the method is called by the user code in the form's Load method. When the form is designed, the method is called by the SAPForms Designer or from outside SAPForms. If you are using the SAPForms Designer as a stand-alone application, the Load method is called with Mode=2. This instructs the Load method to load the form from a file. If you are using the SAPForms Designer as a plug-in, the Load method is invoked from the hosting environment and receives the proprietary form description as a parameter. The populated SAPForm object should be passed to the SAPForms Designer by the PlugIn method of the Designer class.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Frm</td>
<td>form or form template containing controls.</td>
</tr>
<tr>
<td>Mode</td>
<td>0=design time (add-in), 1=design time (file), 2=runtime.</td>
</tr>
</tbody>
</table>

Classification 8
Load

- SAPXmit class [Seite 2106]
- SAPforms Data Plug-In for Demo [Seite 2077]
- SAPXmit class [Seite 1837]
- SAPforms Form Plug-In for Outlook [Seite 1796]
- SAPXmit class [Seite 2126]
- SAPforms Transmit Plug-In for RFC [Seite 2125]
- SAPXmit class [Seite 2168]
- SAPforms Transmit Plug-In for MAPI [Seite 2146]
- SAPXmit class [Seite 2147]
- SAPforms Transmit Plug-In for SAPMAPI [Seite 2167]
- SAPXmit class [Seite 2189]
- SAPforms Transmit Plug-In for WinSock [Seite 2188]
- SAPXmit class [Seite 2209]
- SAPforms Transmit Plug-In for Demo [Seite 2208]
Load (Method)

Properties
Loads the property values of an SAPXmit object from a structured string. As a result, the status of the SAPXmit object that was previously saved permanently with the Save method is restored.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>From</td>
<td>Character string containing the structure information</td>
</tr>
</tbody>
</table>
Options (Method)

Definition
Method that outputs the object-specific options dialog box.

Remarks
This dialog box displays either global settings or settings that are valid for the current form. These settings can be changed here. The global settings are read from the registry before they are displayed and are then saved again to the registry.
The method is called within the SAPforms Designer if the required plug-in is active under Properties on the Plug-Ins tab in the Options dialog. You can also use the sample program Options.exe to display the Options dialog of a certain plug-in.

Classification 1
Options

- SAPData class [Seite 1918]
- SAPforms Data Plug-In for Workflow [Seite 1897]
- SAPData class [Seite 1975]
- SAPforms Data Plug-In for Function Modules [Seite 1954]
- SAPData class [Seite 2027]
- SAPforms Data Plug-In for IDoc [Seite 2006]
- SAPData class [Seite 2078]
- SAPforms Data Plug-In for Demo [Seite 2077]

Classification 2
Options

- SAPForm class [Seite 1772]
- SAPforms Form Plug-In for Visual Basic [Seite 1756]
- SAPForm class [Seite 1812]
- SAPforms Form Plug-In for MS Outlook [Seite 1796]
- SAPForm class [Seite 1872]
- SAPforms Form Plug-In for MS Office [Seite 1856]

Classification 3
Options

- SAPXmit class [Seite 2106]
- SAPforms Data Plug-In for Demo [Seite 2077]
- SAPXmit class [Seite 1837]
- SAPforms Form Plug-In for MS Outlook [Seite 1796]
- SAPXmit class [Seite 2126]
• SAPforms Transmit Plug-In for RFC [Seite 2125]
• SAPXmit class [Seite 2168]
• SAPforms Transmit Plug-In for MAPI [Seite 2146]
• SAPXmit class [Seite 2147]
• SAPforms Transmit Plug-In for SAPMAPI [Seite 2167]
• SAPXmit class [Seite 2189]
• SAPforms Transmit Plug-In for SMTP [Seite 2188]
• SAPXmit class [Seite 2209]
• SAPforms Transmit Plug-In for Demo [Seite 2208]
Save (Method)

Definition
Saves the updated form definition.

Remarks
This method is used by the SAPforms Designer when a form is designed to save the modified form definition after the binding has been defined. The modifications can affect the binding definition or possibly the source code entered in the form.

Classification 1
Save

Properties
Saves the current property values of an SAPData object in a structured string. Counterpart of the Load method. When a form is created, the SAPforms Designer uses this method to save the structure description retrieved from the R/3 System permanently in the form. The R/3 structure is restored from the structured string at runtime by the Load method.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Tgt</td>
<td>String in which the status is stored.</td>
</tr>
</tbody>
</table>

Classification 2
Save

Properties
Saves the current property values of an SAPData object in a structured string. Counterpart of the Load method. The string is saved with a form so that the structure information is available when the form is executed.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Tgt</td>
<td>Stores the string structure.</td>
</tr>
</tbody>
</table>
Classification 3

Save

Properties

Loads the property values of a fields object from a structured string. As a result, the status of the fields object that was previously saved permanently with the Save method is restored.

The string is saved with the form so that the structure information is available when the form is executed.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Tgt</td>
<td>Stores the string structure.</td>
</tr>
</tbody>
</table>

Classification 4

Save

Properties

This method saves the structure of a segment, including the field list, in a structured string.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Tgt</td>
<td>Stores the string structure.</td>
</tr>
</tbody>
</table>

Classification 5

Save

Properties

This method saves the segment list in a structured string. Counterpart of the Load method.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Tgt</td>
<td>Stores the string structure.</td>
</tr>
</tbody>
</table>

Classification 6

Save

Properties
Properties
Method that writes the content of all links to a file. Counterpart of the Load method.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Tgt</td>
<td>Stores the string structure.</td>
</tr>
</tbody>
</table>

Classification 7

Save

 presidente de SAPForm class [Seite 1772]
 presidente de SAPforms Form Plug-In for Visual Basic [Seite 1756]
 presidente de SAPForm class [Seite 1872]
 presidente de SAPforms Form Plug-In for MS Office [Seite 1856]
 presidente de SAPForm class [Seite 1812]
 presidente de SAPforms Form Plug-In for MS Outlook [Seite 1796]

Properties
Method that saves a form to a file if it was created from a file.
Reads the old file and writes updated lines to the new file.

Classification 8

Save

 presidente de SAPXmit class [Seite 2106]
 presidente de SAPforms Data Plug-In for Demo [Seite 2077]
 presidente de SAPXmit class [Seite 1837]
 presidente de SAPforms Form Plug-In for Outlook [Seite 1796]
 presidente de SAPXmit class [Seite 2126]
 presidente de SAPforms Transmit Plug-In for RFC [Seite 2125]
 presidente de SAPXmit class [Seite 2168]
 presidente de SAPforms Transmit Plug-In for MAPI [Seite 2146]
 presidente de SAPXmit class [Seite 2147]
 presidente de SAPforms Transmit Plug-In for SAPMAPI [Seite 2167]
 presidente de SAPXmit class [Seite 2189]
 presidente de SAPforms Transmit Plug-In for WinSock [Seite 2188]
 presidente de SAPXmit class [Seite 2209]
 presidente de SAPforms Transmit Plug-In for Demo [Seite 2208]

Properties
Saves the current property values of an SAPXmit object in a structured string. Counterpart of the Load method.
When a form is created, the SAPforms Designer uses this method to save the structure description retrieved from the R/3 System permanently in the form.
The R/3 structure is restored from the structured string at runtime by the Load method.
<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Tgt</td>
<td>String in which the status is stored.</td>
</tr>
</tbody>
</table>
Start (Method)

Definition
Method that executes the Start function in R/3. The semantics of this function depend on the concrete SAPData object.

Remarks
In the case of the SAPData object for workflow containers, this method starts a workflow and populates its container elements with values from the form fields. This method can be called directly via the SAPData object or indirectly via the Start method of the SAPForm object, whose SAPData property is a reference to the SAPData object. If an SAPXmit object is assigned to the SAPData object, the Start method of the SAPXmit object is called.

Classification 1
Start

Properties
Starts a workflow in the R/3 System and supplies its container elements with the data values in the fields list.

Classification 2
Start

Properties
This method carries out the current R/3 function module call.

Classification 3
Start

Properties
Generates an IDoc file from the data in the segments list and transfers this IDoc to the R/3 System. The transfer method depends on the assigned SAPXmit object. A workflow can be started in the R/3 System when the inbound IDoc is processed.

Classification 4
Start

Properties

Properties
Does not execute an action in the R/3 System but rather displays the data of the object in a separate window.

Classification 5
Start

Properties
Invokes the Start method in the associated SAPXmit or SAPData object. The semantics of this method depend on this SAPData object. If this object is the SAPData object for SAP Business Workflow, this method starts a workflow in an R/3 System.

Classification 6
Start

Properties
This method starts a workflow synchronously via RFC. Supports SAPData objects of types "Idoc" and "Container".

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Data</td>
<td>Object of type SAPData.</td>
</tr>
</tbody>
</table>

Classification 7
Start

Properties
Sends a mail to the R/3 System with an attachment containing the information required to start a workflow.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Data</td>
<td>Object of type SAPData.</td>
</tr>
</tbody>
</table>
Classification 8

Start

- SAPXmit class [Seite 2209]
- SAPforms Transmit Plug-In for Demo [Seite 2208]
- SAPXmit class [Seite 2106]
- SAPforms Data Plug-In for Demo [Seite 2077]

Properties
Displays the instance data of the associated SAPData object in a window instead of sending it to the R/3 System.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Data</td>
<td>Object of type SAPData.</td>
</tr>
</tbody>
</table>
Defining and Generating a Binding

Prerequisites
You have created a form with Visual Basic or MS Outlook. You have launched the SAPforms Designer as a Visual Basic add-in or via the Submit control and are in the main window of the SAPforms Designer. The form controls are shown in the table displayed in the bottom control. The interface (container elements, function module parameters, or segment description of the IDoc) is displayed as a tree structure in the top control in the main window.

Procedure

Define a binding
Assign the individual structure elements to the fields in the current Visual Basic form (“binding definition”). This assignment is stored as part of the form and is used at runtime to transfer data from the form to the R/3 System.
There are two ways of defining a binding.
Simply drag one leaf from the tree structure and drop it onto one table line (“drag & drop”).
You can link several data elements to the same form control using “drag and drop”. This is useful, for example, if a value from the R/3 System is to be displayed more than once on your form.
You can also use the “drag & drop” method to add a leaf from the tree structure to the table in the bottom control by simply dropping it on the SAP Data field column heading. Two form controls (SAPLabel, SAPField) are then also added to your Visual Basic form.
Select a leaf and a table row and choose Binding → To data element.
The name of the leaf is then entered in the row containing the control under the SAP Data Field column.
You can release a binding by selecting the required row in the table and choosing Binding → Release. To release all of the bindings, choose Binding → Release All.

Generating a Binding
Once you have defined all the bindings, you must generate it.
Choose File → Generate.
The binding is generated. If you do not use the SAPforms Designer as an Add In, continue with step 4.

If you use the SAPforms Designer as an Add In (Visual Basic), the tab page Generate in the dialog box SAPforms Options is displayed.
By selecting the Insert initialization code into Load()- method and button to trigger the insert action options on the Generate tab page, you can instruct the Designer to add buttons with assigned event procedures on your Visual Basic form.
This is only recommended the first time you generate your form and if the form does not already have these buttons. Otherwise, you might receive an error when compiling the form because some of the declarations and event procedures are contained twice in the form. You must remove duplicate code from the form by hand.

Choose OK.

All of the necessary information is then entered in the TAG property of the form.

You can undo the form generation by choosing Binding → Reset. The entry in the TAG property is then also deleted.

Once a form has been generated, you can edit it or generate it again at any time. You might need to do so if you have added an input field to your form, for example.

Choose File → Exit to exit the SAPforms Designer.

**Result**

Once you have generated the form successfully, you must recompile the Visual Basic project that contains the form (assuming that you are using Visual Basic). You can then make the resulting EXE file available where it is required.

The procedure is the same if you are using the stand-alone version of the SAPforms Designer.

For a more detailed description of this procedure, please refer to:

- Defining and Generating a Binding: Workflow [Seite 2256]
- Defining and Generating a Binding: BAPI [Seite 2263]
- Defining and Generating a Binding: IDoc [Seite 2278]
Programming Forms

Use
In the case of Visual Basic forms, SAPforms Designer [Seite 1735] inserts a few lines of code into your form to handle communication between the form and the R/3 System (Code inserted by the SAPforms Designer [Seite 2230]). This code is sufficient to support communication with the R/3 System. For more complex forms, however, you may need to insert your own code. Typical situations where you may have to add code are:
You want to design a specific form layout
You want to display context-specific information from the R/3 System
You want to program your own input checks.

Input Checks
In the Process notification of absence example (Workflow Examples [Seite 2258]), the system checks whether a leave request was accepted or rejected. For this purpose, a specific value (A for “Accept”, R for “Reject”) is assigned to a container variable (CheckResult) in the VB code. The assignment is made via the following code:

```vbnet
objSAPForm.sapdata.Value("CheckResult") = "A"
and
objSAPForm.sapdata.Value("CheckResult") = "R"
```
This variable is interrogated when the workflow is executed and is evaluated accordingly.

You want to populate certain SAP data elements from constants or configuration data rather than from form fields.

Data Retrieval
In the examples on Creating a customer master record [Seite 2264] and Creating a customer master record via IDoc [Seite 2280], the information required to call the BAPI but is not to be entered on the form but rather imported from a local file called CUSTOMER.TXT via a VB routine.

You want to program an error handling routine.

For more information see Error Handling Routines [Seite 2233]
You can also call the SAPforms interface [Seite 2231] directly. In this case, you add the required operations (initialize interface, define binding) directly in the code of the Visual Basic form.

Prerequisites
You need a thorough knowledge of programming with Visual Basic.
If the code you want to insert references SAPforms objects, you also need to be extremely familiar with the properties and methods of these objects.

Detailed explanations of the properties and methods is available under the Object Browser in Visual Basic (View → Object Browser). You must include the references to the required SAPforms components in your project (Project → References..). The most important properties and methods are also explained in this documentation. See Plug-Ins [Seite 1746].
Code Inserted by the SAPforms Designer

If you have decided to insert code when the form is generated, the SAPforms Designer generates the following code:

```vbscript
Dim objSAPForm As Object
```

This line declares a global SAPForm object which acts as an anchor to all other SAPforms objects during the lifetime of the form.

```vbscript
Private Sub Form_Load()
    Set objSAPForm = CreateObject("SAPformsFVB4.SAPForm")
    objForm.Init Me, Command
End Sub
```

This object is instantiated by the corresponding class in the Load event of the form and is then initialized. The Init method receives Me, representing a Visual Basic form object, and Command representing the command line parameters. These parameters are filled when the application is started. The Init method creates one entry in the Controls list of the SAPForm object from each control in the Visual Basic form object. It also analyzes the Tag property of the form and reconstructs the SAPData object and the SAPXmit object from its contents. After the Init call, these objects can be accessed via the SAPData and SAPXmit property of the SAPForm object.

The Command parameter is used to decide whether the form is invoked to process a work item or to start a workflow. If a work item is to be executed, the form must be started indirectly by the SAPforms Helper, which itself is activated when an R3F mail attachment is started, for example. In some cases, it can be more advantageous to replace the code generated in the Load event with your own code, especially if the logic around the Command parameter is not suitable for your purposes.

```vbscript
Private Sub cmdStart_Click()
    ' Submit via selected Transmit object
    objSAPForm.Start
End Sub
```

This code is generated in the “Click” event of the Send button. It activates the Start method of the SAPForm object which fills the elements of the SAPData object from the form controls, writes data to an appropriate file, and sends the file to the R/3 System. To do so, it uses the SAPXmit object.

The generated code does not contain any error handling routines and does not return a positive confirmation. This must be added by the form programmer.
Calling the SAPforms Interface Directly

Instead of defining the interface selection and binding permanently using the SAPforms Designer [Seite 1735], you can also save this information using a different method.

Please note, however, that the SAPforms Designer is the simpler and faster method and is therefore recommended.

To understand this procedure, you should be familiar with the SAPforms components that are used here. These components correspond to Visual Basic classes that must be instantiated in the Visual Basic program.

SAPForm [Seite 1755] represents a global SAPForm object that acts as an anchor to all other SAPForm objects.

SAPData [Seite 1895] designates the interface used.

SAPXmit [Seite 2123] defines the communication method.

These objects must be instantiated in the Visual Basic program and initialized accordingly. You must then open the corresponding structure. Finally, you must define the bindings and transfer the data to the R/3 System.

Procedure

In Visual Basic, choose Project → References...

On the dialog box displayed, select the references that you need for your form. You carry out this activity in the SAPforms Designer when you are maintaining the plug-ins. Make sure that you only choose one reference for each object type.

Enter the following code in the load method for your form:

```vba
1 Private Sub Form_Load()
2    Set objSAPForm = New SAPForm
3    Set objSAPForm.SAPData = New SAPData
4
5    Set objSAPForm.SAPXmit = New SAPXmit
6 End Sub
```

This code uses a so-called “early binding”. The term “late binding” refers to code inserted by the SAPforms Designer [Seite 2230].

To import an IDoc structure from a file [filename], enter the following code in line 4:

```vba
4 objSAPForm.SAPData.ImportStructureFromFile ([Filename])
```

The object browser (View → Object-Browser) shows the methods that match each object type.

To assign data from controls to a container element or segment field, enter the following code:
Calling the SAPforms Interface Directly

```
ObjSAPForm.SAPData.Value ([ContainerElement]) = [FormControl]

[ContainerElement] stands for the name of a container element. [FormControl]
stands for the name of the form control or a concrete value (as a string).

This assignment should be made in the Click event (Private Sub
cmdStart_Click()) of the start or execute button, since the form does not contain any
values beforehand.

To transfer the data to the R/3 System, enter the following line to the code at the end of the Click
event:

```
ObjSAPForm.Start

If you want to use the form to process a work item, enter Execute (instead of Start).

---

Example: Creating a customer via an IDoc

This example (SENDIDO1.VBP) is located in the directory ...\SAPforms\Samples.

General procedure:
The IDoc definition is loaded by uploading the definition file. The name of the file in this example
is CCREATE.DES.

The individual segment fields are set by calling the interface directly.

  If the IDoc is to be sent now, it must first be filled with data (header data, useful
data).

  It is important that this data is set in a certain sequence. When the fields in a
segment are set, the actual instance of the segment is created and the sequence in
which the IDoc is sent to the R/3 System is defined. The IDoc will encounter an error
if this sequence does not correspond to the one in the IDoc definition. You can
specify this sequence in the download file or by choosing the transaction WEDI in the
R/3 System.

  In our example, the segments must be populated in the following sequence:

Control record (header data)
Segment E1BPKNA101000
Segment E1BPKNA102000

For further information, refer to the VB code.
Error Handling Routines

The code generated by the SAPforms Designer [Seite 1735] does not contain any error handling routines. As is standard in Visual Basic, you must program your own error handling routines, for example by displaying a Visual Basic error message in a message window using `MsgBox Err.Description`.

Error handling is important to inform form users in a suitable manner of any errors they might have arisen as a result of their inputs.

There are two errors categories - warnings and errors:

**Warnings** are usually output in response to user input errors. The warning notifies the user.

**Errors** are mostly caused by program faults that result from errors in the form code or in SAP Forms components.

The user enters a non-numeric input in a numeric field.

Logon to the R/3 System fails because the password was incorrect. A message window to this effect is to be displayed.

The SAP Forms components give you full control over the errors that are specific to SAPforms. An example of an error generated by SAPforms are those output for field checks that are carried out on user inputs by the SAPData Plug-In [Seite 1895]. If a non-numeric value is entered in a numeric field, for example, an error is output. This error should be recorded and processed in the VB code of the form.

It is advisable to define a corresponding reference to help you program the errors generated by SAPforms:

In Visual Basic, choose `Project → References...`. The `References` dialog box appears.

Choose the `SAPforms constants` reference.

If this reference is not listed, choose the `Browse` option in the `References` dialog box. Then load the file `WDFCONT.TLB` from the directory `\SAPforms\Bin` to your reference library.

Choose `View → Object Browser` and choose the `SAPformsConstants` library. The `sapErrorCodes` class contains the errors generated by SAPforms.

You can use the following VB code to check user inputs, as described in the example above:

```vbnet
On Error GoTo trap
...
trap:
if err.number = sapErr_IllegalNumeric then
    MsgBox "[ErrorMessage]"
...
```

You can use an error number instead of a concrete name. In the same way, you can program all of the fields in the `sapErrorCodes` class in Visual Basic.
Starting Workflows with Electronic Forms

Use
An electronic form that is available locally on the PC or via a server is executed by an employee and sent to the R/3 System.

The task [Extern] (single or multistep task), whose ID was specified when the form was defined, is then started in R/3. The corresponding import elements of the container for this task are then supplied with input data from the form.

Prerequisites

Extend Container
The workflow and/or task container must be extended to include those elements that are to be filled using the electronic form. These elements must be defined as elements with a field reference. At present, object references cannot be filled using a form.

Create a Form and Process it with the SAPforms Designer
After you have created the electronic form [Seite 2237], you must prepare it for communication with the R/3 System. This is the task of the SAPforms Designer [Seite 1735].

Features

Starting tasks
The form is sent to the R/3 System after it has been started and filled with data.

If the task is a multistep task, the workflow container is filled with the values received from the form. These values are transferred to the method container via the task container of the corresponding task. The method then usually creates an object.

Information for a leave application, for example, can be submitted using the form. The CreateWithoutDialog method is then executed in the background in the workflow and creates an object of the type “leave application” from this data. A predefined approval procedure can be carried out in the remaining steps of the workflow. See Workflow Examples [Seite 2258].

The intermediate step via the workflow container does not apply to single-step tasks.

Data Transfer to the R/3 System
Data can be transferred to the R/3 System either synchronously or asynchronously. This is defined with the form.
**Data Transfer by RFC (Synchronous Communication)**

The following steps are processed after the form has been executed:

When the *Submit* button is selected, an R3F file is generated from the form data.

R3F (R/3 Form) is a file format that is used to transfer workflow information from an R/3 System to a form and vice versa.

The R3F file is transferred to the R/3 System via RFC. Here, the data in the file is transferred to the task container by means of a function module call.

If the user who has logged onto the system via RFC is one of the possible agents [Extern] of the task specified in the parameters, the task is started.

This type of data transfer should be selected whenever a network connection can be set up from the user’s PC to provide online access to the R/3 System via RFC. SAPGUI does not need to be installed on the PC in question.

**Data Transfer by Internet Mail (Asynchronous Communication)**

The following steps are processed after the form has been executed:

When the *Submit button* is selected, an Internet mail with an R3F attachment is created and transferred to the background user (usually WF_BATCH) of the logical destination WORKFLOW_LOCAL_<Clt.> in the inbound distribution function.

The background user WF_BATCH forwards the incoming message to the “handler for messages to work items”.

The handler transfers the data in the R3F attachment to the container of the task to be started by calling a function module.

If the sender of the mail is one of the possible agents of the task specified in the attachment, the task is started.
Each of the users involved must maintain the automatic forwarding function correctly, otherwise this check will not be successful. The significance of the automatic forwarding function will be explained with other prerequisites in SAPforms Administration [Seite 1727].

In mail systems with offline capability (for example Microsoft Exchange), the mails are always transferred as soon as the user establishes a connection to the mail server. You can, therefore, submit your form even if your PC is currently not connected to the Exchange Server.

**Exception Handling**

If an exception should occur in one of the steps described above, the person who processed the form is notified via RFC or mail. The two most common exceptions are:

1. The sender is not one of the possible agents of the task.
2. Individual data in the R3F attachment could not be assigned to the container of the task to be started because the data types are incompatible.
Designing an Electronic Form to Start a Workflow

Purpose
An employee who does not have SAPGUI installed on his or her PC would like to create a leave application. A workflow for this scenario already exists in the R/3 System. This workflow can now be started with an electronic form either offline via a mail client or online by calling the form application directly.

Process Flow
In the process described here, it is assumed that you are using Visual Basic. This also applies to the individual sections in this chapter. However, you can also design forms with MS Outlook [Seite 1731], MS Word or MS Excel [Seite 1733].
To start a workflow in the R/3 System using an electronic form, the following steps are necessary:

In the R/3 System:
- Choose or define the relevant workflows
- Create the organizational plan and assign agents
- Extend the container definition
- Define the bindings.
  Defining a task to be started with a form [Seite 2238]

In Visual Basic:
- Design the electronic form
- Call the SAPforms Designer as an add-in or
  Designing forms with Visual Basic [Seite 1730]

In the SAPforms Designer:
- Maintain the plug-ins.
  Maintaining Plug-Ins and Opening a Structure: Workflow [Seite 2254]
- Define the binding
- Generate
  Defining and Generating a Binding: Workflow [Seite 2256]

Result
Once your form has been designed and processed by the SAP Forms Designer, you can make the form accessible to the end users.
Defining a Task to be Started with a Form

Prerequisites
You are familiar with the procedure for defining tasks [Seite 1172]. For this reason, a recap of only the most important steps will be provided here.
You want to define the task (single or multistep task) that is to be started by the electronic form.

The part of the task definition that is relevant to forms essentially entails extending the container definition to include the elements that are to be processed as fields on your form.

Procedure
To create the task choose Tools → Business Workflow → Development → Definition tools → Tasks/ Task groups → Create.
Enter the type of task you want to create in the Task type field.
Then choose Task → Create.
Define the basic data of your task.

Extending the Container Definition
Now extend the container of your task to include the elements that are to be filled via the form.
To launch the container editor, choose Goto → Container.
The container already contains the standard system elements. To add elements to the container, choose Edit → Create.
You can only transfer single values (not object references) from the form. If you want to add container elements for this purpose, therefore, you must always create them as single-line elements with a data type reference to a table field in the dictionary.
Create a suitable container element for all of the fields you want to fill in the form. Define these elements as import elements.
Since you cannot use elements that reference objects or structures or multiple line elements, you may need to make extensive changes to the container.

Further Procedure
Complete the task definition:
Declare the possible agents [Extern] that are allowed to start the task by filling out and submitting the form.
Save your task definition in the development class that you entered in the Customizing activity for maintaining the namespace.
If the task is a multistep task, maintain the associated workflow definition.
Maintaining Plug-Ins and Opening a Structure: Workflow

Prerequisites
You have created a form with Visual Basic.
You have either defined a single-step task to be executed as a form and copied this task to your workflow definition or have defined a single or multistep task that is to be started by a form.
You have launched the SAPforms Designer as a Visual Basic add-in and are in the main window. The form controls are shown in the table displayed in the bottom control.

The procedure described below is the same if you are using the Submit control or the stand-alone version of the SAPforms Designer.

Procedure
Maintain Plug-Ins
Choose Tools → Options.

The Options dialog box of the SAPforms Designer is then displayed.
Select the Plug-Ins tab page.
Enter SAPforms Data Plug-In for Workflow in the Active Plug-In field in the SAP Forms Data Plug-In box.

The value in the SAPforms Plug-In box is set by default according to the form development environment you are using. If you are using the stand-alone version of the SAPforms Designer, you must also maintain this plug-in.

You can enter one of the following values in the Active Plug-In field in the SAP Transmit Plug-In box.

<table>
<thead>
<tr>
<th>Communication Plug-Ins</th>
</tr>
</thead>
<tbody>
<tr>
<td>SAPforms Transmit Plug-In for RFC</td>
</tr>
<tr>
<td>SAPforms Transmit Plug-In for MAPI</td>
</tr>
<tr>
<td>SAPforms Transmit Plug-In for SMTP via WinSock</td>
</tr>
</tbody>
</table>

Once it has been processed, the work item can communicate with the R/3 System either synchronously via RFC or asynchronously via mail. The same applies to starting a task. The decision is made by choosing the appropriate communication plug-in.

Note that certain default settings must be defined so that you can exchange data via mail. For more information, see SAPforms Administration.

Exit the dialog box.
The Workflow Task Description entry has been added to the File menu in the main window of the SAPforms Designer.
Opening a Structure

Choose File → Workflow Task Description.

The SAPforms RFC Logon - System Selection dialog box is then displayed.

Choose the R/3 System you want to log on to from the System tab page.

You can find out which R/3 Systems you are already logged on to on the tab page Recent. You can also select an existing logon.

Choose Next and log on.

The Select (Workflow) Task dialog box appears.

Enter the number of the task in the Selected Task field and choose OK.

You can search for a workflow in R/3 by using the pushbutton Search in the Search area. Enter the task code in the field Search Key.

Result

The container definition for this task is exported from R/3 and displayed as a tree in the top control of the main SAPforms Designer window. You can now define the binding [Seite 2256].
Defining and Generating a Binding: Workflow

Prerequisites
You have created a form with Visual Basic.
You have either defined a single-step task to be executed as a form and copied this task to your workflow definition or have defined a single or multistep task that is to be started by a form. You have maintained the plug-ins for the interface and the communication method.
You have launched the SAPforms Designer and are in the main window. The form controls are shown in the table displayed in the bottom control. The top control displays the container definition for your task as a tree structure.

Procedure
Defining and Generating a Binding
You must first assign the individual in the container structure to the fields in the current Visual Basic form. This assignment is stored as part of the form and is evaluated at runtime to transfer data from the form to the R/3 System.
Expand the container structure.
The elements are displayed.
Assign (using “drag and drop”) the parameter that you want to fill with the form to the form controls.
You can release the binding by selecting the required row in the table and choosing Binding → Release.
Choose File → Generate.
The Generate tab page on the SAPforms Options dialog box is then displayed.
Select Insert initialization code in Load()-method and button to trigger the insert action.
Choose OK.

⚠️
This is only recommended the first time you generate your form and if the form does not already have these buttons. Otherwise, you might receive an error when compiling the form because some of the declarations and event procedures are contained twice in the form. You must remove duplicate code from the form by hand.

Choose File → Exit to exit the SAPforms Designer.

Result
Your form is now complete. You can now start it in Visual Basic by choosing Run or compile an EXE file.

The procedure is the same if you are using the stand-alone version of the SAPforms Designer.
If you want to process a work item with the form using the Submit control [Seite 1747], you must insert the following line in the load event of the VB code.

SAPformsSubmit1.SAPform.Init Me, Command
Workflow Examples

The directory ...\SAPforms\Samples contains two examples (APPROVE.VBP, CUSTOMER.VBP) of approval workflows that illustrate the functions for starting workflows and processing work items. These workflows cover the scenarios: Process notification of absence (WS 30000161) and Example workflow for customer creation (WS 20000084).

Also compare the documentation to workflow WS 30000161 Demo Example: Edit Absence Notification as Form [Seite 72].

Scenarios

In both scenarios, an employee starts a form as an EXE file on a local PC. He or she completes the form and sends it to the R/3 System by clicking the Submit button. This example is designed for communication via RFC. In other words, an online connection to the R/3 System is required.

Example: Scenario WS 20000084 (Simplified)
The employee receives the message "Your request has been submitted successfully". The corresponding workflow is then started in the R/3 System. The employee responsible for approving the notification then receives a work item in his or her Workflow inbox [Seite 1408]. If he or she chooses Execute, the work item starts the form again. This time, however, only values are displayed. The form now also contains a Reject and an Accept button. Depending on the superior's decision, the employee receives a rejection or an approval in his or her workflow inbox.
SAP AG
Workflow Examples

For the scenario example workflow for creating customers (WS 20000084) to
function, the file CUSTOMER.TXT must also be located in the directory, in which the
executable EXE-file (the form) is located, as the form reads initial values from this file.
The file is stored in the directory ...SAPforms\Forms.
It concerns the data for state, reference customer sales organization, division and
distribution channel ("*"). Except for state this data is system-specific and must be
determined separately. In order for the example to work, you must maintain the
appropriate data for your system in the file CUSTOMER.TXT. See also: Determining
a Reference Customer [Seite 2282].

Process Flow
To use these examples, you need to carry out the following steps:

Start Visual Basic and load the file APPROVE.VBP or CUSTOMER.VBP from the
...\SAPforms\Samples directory.
Start the SAPforms Designer as a Visual Basic add-in.
The structure of the workflow WS 30000161 or WS 20000084 is displayed in the top
control. The bottom control lists the form controls in a table. The binding is already
defined and generated.

The additions to the VB code required for this scenario have already been integrated
and commented. These are, for example, the program lines (Private Sub
Form_Load ()) that ensure that a Reject and Accept button is added to the form
when it is processed as a work item. The result of this query must also be processed
(Private Sub cmdAccept_Click(), Private Sub cmdReject_Click()).
A certain value, therefore, is assigned to the container element CheckResult
(APPROVE.VBP) and ProcessingState (CUSTOMER.VBP) and is interrogated later in
the workflow.
Compile the APPROVE.EXE or CUSTOMER.EXE file and save it in the ...\SAPforms\Forms
directory.
Close Visual Basic and maintain an organizational plan in your R/3 System. Assign agents to the
tasks.
Start the APPROVE.EXE or CUSTOMER.EXE file.
One of the scenarios described above is then started.

April 2001

2244


Executing a Work Item as an Electronic Form

Use
You can define a step in a workflow with a form task so that it must be processed with an electronic form. Even though the work item is still displayed in the user’s workflow inbox in the R/3 System, it is actually processed with an electronic form that is stored as a local file on the PC.

Prerequisites
Extend Container
The container of the form task must be extended to include those elements that are to be displayed or processed on the form. Only elements that reference a field (and not an object) are permitted. The elements are declared as import or export elements in accordance with the standard workflow logic. If data is to be entered via the form, the corresponding elements in the task container must be declared as export elements. This data can be transferred from the task container to the workflow container.

Elements declared as import elements can only be displayed, since changes to the data do not affect the workflow container.

Create a Form and Process it with the SAPforms Designer
After you have created the electronic form, you must prepare it for communication with the R/3 System. This is the task of the SAPforms Designer.

Features
A form task only makes sense as a step in a workflow definition. The information that is to appear and possibly be modified on the form is created in the previous steps. Depending on the way in which the form has been configured, the user is allowed to change values on the form (and thus indirectly in the work item container) and therefore affect how the workflow is processed. When the form is executed, the modified values are returned to the “waiting” work item in the R/3 System. When the data is received in R/3,

- the work item container is filled with the values from the modified form
- the work item for which the form was created is terminated.

The changes to the contents of the workflow container are then evaluated in one of the subsequent workflow steps.
Executing a Form Task as Work Item in the workflow inbox

When a form task is executed in a workflow, the associated work item is then displayed in the workflow inbox [Seite 1408] of the agent responsible [Extern] for this step in the usual way.

The following steps are then carried out:

If the agent chooses “Execute”, the form task transfers an R3F file to the SAPforms Helper [Seite 1741].

R3F (R/3 Form) is a file format that is used to transfer workflow information from an R/3 System to a form and vice versa. The R3F file contains the data from the form task container as well as the name of the form file.

The form is started.

Once the agent has completed and submitted the form, the modified R3F file is returned to the R/3 System where it enters the modified data in the task container. For further details on transferring data to the R/3 System, please refer to the section on Starting Workflows with Electronic Forms [Seite 2234].
Executing a Form Task as an Internet Mail in an External Mail System

In addition to the work item that is created as standard, an Internet mail, which contains the data from the work item container in an R3F attachment is sent to the selected agents.

When the attachment is opened in the external mail system, the SAPforms Helper [Seite 1741] is started and presents the form to the user with the values contained in the attachment.

Once the agent has completed and submitted the form, the modified R3F file is returned to the R/3 System where it enters the modified data in the task container. For further details on transferring data to the R/3 System, please refer to the section on Starting Workflows with Electronic Forms [Seite 2234].

The following default settings determine whether this Internet mail is created at all and to which of the selected agents it is sent:

Every agent in the user master who wants to use this functionality must make sure that the "automatic forwarding" function is activated to his or her Internet address.

A report must be started to search the system for relevant work items at periodic intervals and sends a mail for the work item to the Internet address defined under the automatic forwarding settings.

For more information, see SAPforms Administration [Seite 1727].
Form task

Definition
Single-step task that executes the Process method of the FORM object type [Seite 74].

Use
When a form task is executed as a step in a workflow, a work item is displayed in the workflow inbox of the recipients for this step in the usual way. When this work item is executed, its container is processed with the associated electronic form using this method.
If the system has been set correctly, the recipients of the work item will also receive a mail sent to their Internet address. This mail contains the data from the work item container in the form of an R3F attachment.
There are a wide range of form tasks for different applications. These tasks differ in particular with regard to their task containers, the elements of which define the fields that can be displayed or edited with the electronic form.
Designing a Form for Execution as a Work Item

Purpose
You want to process a work item with an electronic form. This can be useful, for example, if an agent of a certain workflow wants to execute a workflow step offline using his or her mail client. The agent, therefore, only needs the form application on his or her local PC.

Process Flow
In the process described here, it is assumed that you are using Visual Basic. This also applies to the individual sections in this chapter. However, you can also design forms with MS Outlook.

To execute an electronic form as a work item, you need to carry out the following steps:

In R/3:
Define a workflow
Define the form task (and assign the relevant agents)
Add the elements to the form task container that are to be entered or displayed on the form
Enter the file name and type of the form file (see below) in the initial values of the FORM.Process object method
Add the form task to the workflow definition as an activity (including the assigned agents)
Define the binding between the workflow container and the elements in the task container

Adding a Task to a Workflow Definition

You can also use the Include “Execute form” wizard instead of following the steps described above. You can start this wizard in the Workflow Builder.

In Visual Basic:
Design the electronic form
Save the form file in the ...\SAPforms\Forms directory
The form file must be stored in the ...\SAPforms\Forms directory or in the active directory so that it can be found by the SAPforms Helper. If you want to save your form application in a different directory, you must change the settings in the system.

Call the SAPforms Designer as an add-in or
Use the Submit control.

In the SAPforms Designer:
Maintain the plug-ins.

Define the binding
Generate

Defining and Generating a Binding: Workflow [Seite 2256]

Result
Once your form has been designed and processed by the SAPforms Designer [Seite 1735], you can make the form accessible to the end users.
Defining a Task to be Executed as an Electronic Form

Prerequisites
You are familiar with the procedure for defining tasks [Seite 1172]. For this reason, a recap of only the most important steps will be provided here.
You have defined a workflow and now want to add a single-step task that can be processed with an electronic form.

The part of the task definition that is relevant to forms essentially entails extending the container definition to include the elements that are to be processed as fields on your form.

Using the FORM.Process object method.

Procedure
Define the Form Task
Define the single-step task as a form task [Seite 2248]:
Create a single-step task for the form application that you want to execute from a workflow. You can specify any name you want for this task.

Enter the object type FORM.

Enter the Process method.

Choose YES to confirm that elements that do not exist in the task container are to be copied from the method parameter container of the object method.

The system automatically creates the four parameters R3FForm (form name), R3FFormTyp (form type), Workitem of the referenced object method, and CompletionDocument as elements in the task container. A binding definition between the task container and the method container has also been created for these elements.

The three method parameters cannot be completely copied automatically to the task container. For this reason, make sure that you carry out the following steps:

Display the task container by choosing Goto → Container.

Delete the Workitem element and exit the screen.

Choose Goto → Binding → Object method to start the Binding Editor to define the binding between the task container and method container.

Enter the following binding:

Workitem <= &_Workitem&
Extending the Container Definition
You can add other container elements to a container as form-specific extensions. The contents of these elements can then be displayed and edited in the fields on the form. All of the data that is to be displayed to the form user must exist as container elements in the task container.

To edit the task container, choose Goto → Container.

To add elements to the task container, choose Edit → Create and follow the standard procedure:

Remember that only elements that reference fields can be transferred to the form application. Do not forget to identify the individual elements as import or export elements (or both), depending on how they are to be used.

Create a suitable container element for all of the fields you want to fill in the form.

Exit the Container Editor.

Agent Assignment
When you define a single-step task, you must specify its possible agents [Extern].

To do so, choose Tools → Agent assignment → Maintain.

Save your task definition in the development class that you entered in the Customizing activity for maintaining the namespace.

Assigning Initial Values
The single-step task must now be told the name of the form that is to be executed. For this reason, the form type and form must be assigned as initial values when you define the task.

To assign initial values, choose Goto → Assign initial values.

Assign appropriate values to the R3FFormTyp (form type) and R3FForm (form name) container elements.

If you are using the Visual Basic form [Seite 1730] APPROVE.EXE enter EXE as the form type and APPROVE as the form name.

If you are using the Outlook form template [Seite 1731] WFAPPROV2.OFT enter OFT as the form type and WFAPPROV2 as the form name.

Further Information
You can also enter a long-text description for your single-step task to inform the mail recipient that he or she can start the form application by executing the attachment.

If you do not enter your own long-text description, a standard text will be displayed.
**Adding tasks to a workflow definition**

**Electronic forms**

**Prerequisites**
You have defined a workflow. You have defined a form task [Seite 2248] for execution as a form. You have started the Workflow Builder [Seite 1002].

**Procedure**
The workflow definition for executing your form application is processed as follows:

Add a step in the Workflow Builder that represents an activity in which your form task is referenced.

Define the binding between the workflow container and the elements in the task container that you defined previously for your form application.

If the recipient is to be allowed to change or enter data in the form, you must also define a binding from the modified elements in the task container to the corresponding elements in the workflow container.

Delete all of the binding definitions that may have been created automatically and that refer to the element _WI_Object_ID.

**Further Notes**
If data was changed in the form application and copied to the workflow container, you must define the following steps to evaluate this information.
Maintaining Plug-Ins and Opening a Structure: Workflow

Prerequisites

You have created a form with Visual Basic.
You have either defined a single-step task to be executed as a form and copied this task to your workflow definition or have defined a single or multistep task that is to be started by a form.
You have launched the SAPforms Designer [Seite 1735] as a Visual Basic add-in and are in the main window. The form controls are shown in the table displayed in the bottom control.

The procedure described below is the same if you are using the Submit control [Seite 1747] or the stand-alone version of the SAPforms Designer.

Procedure

Maintain Plug-Ins

Choose Tools → Options.

The Options dialog box of the SAPforms Designer is then displayed.

Select the Plug-Ins tab page.

Enter SAPforms Data Plug-In for Workflow [Seite 1897] in the Active Plug-In field in the SAP Forms Data Plug-In box.

The value in the SAPforms Plug-In box is set by default according to the form development environment you are using. If you are using the stand-alone version of the SAPforms Designer, you must also maintain this plug-in.

You can enter one of the following values in the Active Plugin field in the SAP Transmit Plug-In box.

<table>
<thead>
<tr>
<th>Communication Plug-Ins</th>
</tr>
</thead>
<tbody>
<tr>
<td>SAPforms Transmit Plug-In for RFC [Seite 2125]</td>
</tr>
<tr>
<td>SAPforms Transmit Plug-In for MAPI [Seite 2146]</td>
</tr>
<tr>
<td>SAPforms Transmit Plug-In for SMTP via WinSock [Seite 2188]</td>
</tr>
</tbody>
</table>

Once it has been processed, the work item can communicate with the R/3 System either synchronously via RFC or asynchronously via mail. The same applies to starting a task. The decision is made by choosing the appropriate communication plug-in.

Note that certain default settings must be defined so that you can exchange data via mail. For more information, see SAPforms Administration [Seite 1727].

Exit the dialog box.

The Workflow Task Description entry has been added to the File menu in the main window of the SAPforms Designer.
Opening a Structure

Choose File → Workflow Task Description.

The SAPforms RFC Logon - System Selection dialog box is then displayed.

Choose the R/3 System you want to log on to from the System tab page.

You can find out which R/3 Systems you are already logged on to on the tab page Recent. You can also select an existing logon.

Choose Next and log on.

The Select (Workflow) Task dialog box appears.

Enter the number of the task in the Selected Task field and choose OK.

You can search for a workflow in R/3 by using the pushbutton Search in the Search area.

Enter the task code in the field Search Key.

Result

The container definition for this task is exported from R/3 and displayed as a tree in the top control of the main SAPforms Designer window. You can now define the binding [Seite 2256].
Defining and Generating a Binding: Workflow

Prerequisites
You have created a form with Visual Basic.
You have either defined a single-step task to be executed as a form and copied this task to your workflow definition or have defined a single or multistep task that is to be started by a form. You have maintained the plug-ins for the interface and the communication method.
You have launched the SAPforms Designer and are in the main window. The form controls are shown in the table displayed in the bottom control. The top control displays the container definition for your task as a tree structure.

Procedure

Defining and Generating a Binding
You must first assign the individual in the container structure to the fields in the current Visual Basic form. This assignment is stored as part of the form and is evaluated at runtime to transfer data from the form to the R/3 System.
Expand the container structure.
The elements are displayed.
Assign (using “drag and drop”) the parameter that you want to fill with the form to the form controls.
You can release the binding by selecting the required row in the table and choosing Binding → Release.

Choose File → Generate.
The Generate tab page on the SAPforms Options dialog box is then displayed.
Select Insert initialization code in Load()- method and button to trigger the insert action.
Choose OK.

⚠️  This is only recommended the first time you generate your form and if the form does not already have these buttons. Otherwise, you might receive an error when compiling the form because some of the declarations and event procedures are contained twice in the form. You must remove duplicate code from the form by hand.

Choose File → Exit to exit the SAPforms Designer.

Result
Your form is now complete. You can now start it in Visual Basic by choosing Run or compile an EXE file.

The procedure is the same if you are using the stand-alone version of the SAPforms Designer.
If you want to process a work item with the form using the Submit control [Seite 1747], you must insert the following line in the load event of the VB code.

SAPformsSubmit1.SAPform.Init Me, Command
Workflow Examples

The directory ...\SAPforms\Samples contains two examples (APPROVE.VBP, CUSTOMER.VBP) of approval workflows that illustrate the functions for starting workflows and processing work items. These workflows cover the scenarios: Process notification of absence (WS 30000161) and Example workflow for customer creation (WS 20000084).

Also compare the documentation to workflow WS 3000161 Demo Example: Edit Absence Notification as Form [Seite 72].

Scenarios

In both scenarios, an employee starts a form as an EXE file on a local PC. He or she completes the form and sends it to the R/3 System by clicking the Submit button. This example is designed for communication via RFC. In other words, an online connection to the R/3 System is required.

Example: Scenario WS 20000084 (Simplified)

The employee receives the message "Your request has been submitted successfully". The corresponding workflow is then started in the R/3 System. The employee responsible for approving the notification then receives a work item in his or her Workflow inbox [Seite 1408]. If he or she chooses Execute, the work item starts the form again. This time, however, only values are displayed. The form now also contains a Reject and an Accept button. Depending on the superior’s decision, the employee receives a rejection or an approval in his or her workflow inbox.

For the scenario example workflow for creating customers (WS 20000084) to function, the file CUSTOMER.TXT must also be located in the directory, in which the
executable EXE-file (the form) is located, as the form reads initial values from this file. The file is stored in the directory ...SAPforms\Forms.

It concerns the data for state, reference customer sales organization, division and distribution channel ("**"). Except for state this data is system-specific and must be determined separately. In order for the example to work, you must maintain the appropriate data for your system in the file CUSTOMER.TXT. See also: Determining a Reference Customer [Seite 2282].

**Process Flow**

To use these examples, you need to carry out the following steps:

1. Start Visual Basic and load the file APPROVE.VBP or CUSTOMER.VBP from the ...\SAPforms\Samples directory.
2. Start the SAPforms Designer as a Visual Basic add-in.
   
   The structure of the workflow WS 30000161 or WS 20000084 is displayed in the top control. The bottom control lists the form controls in a table. The binding is already defined and generated.

   The additions to the VB code required for this scenario have already been integrated and commented. These are, for example, the program lines (Private Sub Form_Load () ) that ensure that a Reject and Accept button is added to the form when it is processed as a work item. The result of this query must also be processed (Private Sub cmdAccept_Click(), Private Sub cmdReject_Click()). A certain value, therefore, is assigned to the container element CheckResult (APPROVE.VBP) and ProcessingState (CUSTOMER.VBP) and is interrogated later in the workflow.

3. Compile the APPROVE.EXE or CUSTOMER.EXE file and save it in the ...\SAPforms\Forms directory.
5. Start the APPROVE.EXE or CUSTOMER.EXE file.
   
   One of the scenarios described above is then started.
Calling a BAPI

Use

The SAPforms interface allows you to call any BAPI using an electronic form. As is the case with the workflow container, you can define a binding between fields in the electronic form (form controls) and the BAPI parameters.

You can also call function modules that are not BAPIs. In order to do so, however, the process type Remote Function Call Supported must be specified in the administration data of the function modules.

Prerequisites

BAPI Selection

You know the BAPI in the R/3 System that you want to call. You also know the obligatory parameters of this BAPI. If these parameters are not filled with data, the BAPI cannot be called. You can display the import and export parameters of a BAPI in the BAPI Browser (transaction BAPI).

Create a Form and Process it with the SAPforms Designer

After you have created the electronic form [Seite 2261], you must prepare it for communication with the R/3 System. This is the task of the SAPforms Designer [Seite 1735].

Features

The electronic form is filled out and sent to the R/3 System. The BAPI parameters are filled with data from the form fields after you have logged on to the system. This method will be used here to create a new customer (synchronously).

In many applications, some of the parameters are not to be filled from the inputs in the form fields, for example, because the BAPI requires input data that would only confuse the person filling out the form. In this case, the data can be filled in the background.

In the CALLBAPI.VBP model, which is stored in the ...\SAPforms\Samples directory, this function is carried out by the CUSTOMER.TXT file (\SAPforms Forms). The file must be in the same directory as the compiled EXE file for this purpose.

Example: Create customer master record [Seite 2264]
Designing an Electronic Form to Call a BAPI

Purpose
You want to use an electronic form to call a BAPI. This is necessary, for example, if the user does not have access to SAPGUI. In order to call a BAPI (e.g. to “create a new customer”), an online connection must be established to an R/3 System. If the BAPI is to be called offline from the notebook of an external employee, you must use an IDoc. This scenario is demonstrated in the example: Create customer via IDoc [Seite 2280]

Process Flow
In the process described here, it is assumed that you are using Visual Basic. This also applies to the individual sections in this chapter. However, you can also design forms with MS Outlook [Seite 1731]. To design an electronic form to call a BAPI, you must work through the following steps:

In the R/3 System:
- Choose the relevant BAPIs
- Determine the import parameters for the BAPI
  Calling a BAPI [Seite 2260]

In Visual Basic:
- Design the electronic form
- Call the SAPforms Designer as an add-in or
- Use the Submit control.
  Designing forms with Visual Basic [Seite 1730]

In the SAPforms Designer:
- Maintain the plug-ins.
- Open the BAPI structure
  Maintaining Plug-Ins and Opening a Structure: BAPI [Seite 2262]
- Define the binding
- Generate
  Defining and Generating a Binding: BAPI [Seite 2263]

Result
Once your form has been designed and processed by the SAPforms Designer [Seite 1735], you can make the form accessible to the end users.
Maintaining Plug-Ins and Opening a Structure: BAPIs

Prerequisites
You have created a form in Visual Basic and have launched the SAPforms Designer as an add-in and are in the main window. The form controls are shown in the table displayed in the bottom control.

The procedure described below is the same if you are using the Submit control or the stand-alone version of the SAPforms Designer.

Procedure
Choose File → Open SAP Structures → and then the entry for BAPIs.

The SAPforms RFC Logon - System Selection dialog box is then displayed.

Choose the R/3 System you want to log on to from the System tab page.

You can find out which R/3 Systems you are already logged on to on the tab page Recent. You can also select an existing logon.

Choose Next and log on.

Enter the name of the BAPI you want to call.

The interface description of the BAPI is exported from R/3 and displayed as a tree in the top control of the main SAPforms Designer window.

You do not need to maintain the plug-ins on a separate screen for this form application. You have already declared the Plug-In for the Function Module Interface with the Open command. A communication plug-in is not required.

The value in the SAPforms Plug-In box is set by default according to the form development environment you are using. If you are using the stand-alone version of the SAPforms Designer, you must also maintain this plug-in.

Result
You can now define the binding.
Defining and Generating a Binding: BAPIs

Prerequisites
You have created a form in Visual Basic and have launched the SAPforms Designer [Seite 1735] as an add-in and are in the main window. The form controls are shown in the table displayed in the bottom control. You have selected the BAPI that you want to call. The interface description of the BAPI is displayed as a tree in the top control of the main SAPforms Designer window.

Procedure
You must now assign the individual parameters of the BAPI to the fields in the current Visual Basic form (“binding definition”). This assignment is stored as part of the form and is used at runtime to transfer data from the form to the R/3 System.

1. Explode the BAPI structure to display the parameters.
2. Assign (using “drag and drop”) the parameter that you want to fill with the form to the form controls.
   - You can release the binding by selecting the required row in the table and choosing Binding → Release.
3. Choose File → Generate.
   - The General tab on the SAPforms Options screen is then displayed.
4. Select Insert initialization code in Load()- method and button to trigger the insert action.
5. Choose OK.

⚠️ This is only recommended the first time you generate your form and if the form does not already have these buttons. Otherwise, you might receive an error when compiling the form because some of the declarations and event procedures are contained twice in the form. You must remove duplicate code from the form by hand.

Choose File → Exit to exit the SAPforms Designer.

Result
Your form is now complete. You can now start it in Visual Basic by choosing Run or compile an EXE file.

The procedure is the same if you are using the stand-alone version of the SAPforms Designer.
Example: Creating a Customer Master Record

A model VB project (CALLBAPI.VBP) for calling the BAPI_Customer_CreateFromData BAPI of the object type Customer (KNA1) is stored in the ...SAPforms\Samples directory. This method is used to create a customer master record in the R/3 System.

Process Flow

To use this example, you need to carry out the following steps:

Start Visual Basic and load the CALLBAPI.VBP project from the SAPforms directory.

Start the SAPforms Designer as a Visual Basic add-in.

The BAPI structure is displayed in the top control. The form controls are shown in the table displayed in the bottom control. The binding is already defined and generated. The steps for Maintaining Plug-Ins and Opening a Structure [Seite 2262] and Defining and Generating a Binding [Seite 2263], therefore, have been completed.

All of the parameters required to execute the BAPI are listed in the table below:

<table>
<thead>
<tr>
<th>Name</th>
<th>Parameter</th>
<th>Values</th>
</tr>
</thead>
<tbody>
<tr>
<td>Name</td>
<td>PI ADDRESS.Name</td>
<td>Any</td>
</tr>
<tr>
<td>First name</td>
<td>PI ADDRESS.First_Name</td>
<td>Any</td>
</tr>
<tr>
<td>Street</td>
<td>PI ADDRESS.Street</td>
<td>Any</td>
</tr>
<tr>
<td>Postal code</td>
<td>PI ADDRESS.Postl_Code</td>
<td>5-digit number</td>
</tr>
<tr>
<td>City</td>
<td>PI ADDRESS.City</td>
<td>Any</td>
</tr>
<tr>
<td>Reference customer</td>
<td>PI_COPYREFERENCE.Ref_Custmr</td>
<td>&quot;*&quot;</td>
</tr>
<tr>
<td>Country</td>
<td>PI_ADDRESS.Country</td>
<td>US&quot;*&quot;</td>
</tr>
<tr>
<td>Language</td>
<td>PI_ADDRESS.Langu</td>
<td>US</td>
</tr>
<tr>
<td>Sales organization</td>
<td>PI_COPYREFERENCE.Salesorg</td>
<td>&quot;*&quot;</td>
</tr>
<tr>
<td>Division</td>
<td>PI_COPYREFERENCE.Division</td>
<td>&quot;*&quot;</td>
</tr>
<tr>
<td>Distribution channel</td>
<td>PI_COPYREFERENCE.Distr Chan</td>
<td>&quot;*&quot;</td>
</tr>
</tbody>
</table>

Not all attributes are filled by the form. Instead, they are filled with initial values in the background by an import routine in VB code. These values are stored in the CUSTOMER.TXT file in the directories ..\SAPforms\Samples or ..\SAPforms\Forms. It is important that the file is in the same directory as the EXE file read for this purpose.

It concerns the data for state, reference customer sales organization, division and distribution channel ("*"). Except for state this data is system-specific and must be determined separately. In order for the example to work, you must maintain the appropriate data for your system in the file CUSTOMER.TXT. See also: Determining a Reference Customer [Seite 2282].

Test your form by choosing Run in Visual Basic or generate and start an EXE file.

Enter meaningful data (see above) and choose Start. You are then logged on to your R/3 System automatically. After a short period of time, the message callbapi "The BAPI has been called successfully" is displayed.

This means that a customer master record has been successfully created in your R/3 System.
Example: Creating a Customer Master Record
Determining a Reference Customer

You must know the (or a) reference customer before you can execute the
`BAPI_CUSTOMER_CREATEFROMDATA` BAPI. This BAPI is used in the following examples provided in
the SAPforms documentation:

Create customer master record [Seite 2264]

Create customer master record via IDoc [Seite 2280]

The reference customers used in your company (and therefore the other sales data) are normally known. If
this is not the case, you can determine the reference customers from the internal number ranges. You can
only do so, however, if the reference customer was created with the internal number assignment function.
In other words, there are two ways of finding a suitable reference customer:

Determining an existing reference customer

Creating a new reference customer

From a business perspective, it makes more sense to create a separate reference customer for the form
application. To do so, use the transaction `V-09`.

Procedure

Determine the Internal Number Ranges

Start transaction `SPRO` and choose `Basic functions → Enterprise IMG → Display`. The `Structure`
screen is then displayed: `Enterprise IMG`.

Expand the tree structure by choosing `Financial Accounting → Accounts Receivable and
Accounts Payable → Customer Accounts → Master Records → Preparations for Creating
Customer Master Records Create` and choose `number ranges for customer accounts`. The
`Customer Number Ranges` screen is then displayed.

Choose `Display intervals`. The `Display Number Range Intervals` screen is then displayed. All of
the number ranges that have been defined for customers are listed here. Number ranges
with internal number assignment are those where the `Ext` box is not selected. Make a note of
the internal number ranges.

Determine the Reference Customer

Choose the transaction `VD03`. The `Display Customer: Sales Area Data Initial Screen` is then
displayed. `Sales Area Initial Screen`.

Click the F4 Help in the `Customer` field and choose `Customers per sales group` on the window
that then appears. This dialog box is displayed as standard the first time you click the
possible entries button. Your selection is stored, which means that this dialog box (in this
case `Customers per sales group`) will be displayed in future whenever you choose the
possible entries help.

Choose the `multiple selection` function for the `Customer` field on this dialog box. Enter the number
range intervals here.

Exit the `multiple selection` by choosing `Copy`. Then choose `Continue`. A list of customers and
associated sales data is then displayed. Sales organization, division and distribution channel.

Choose a customer for which all of the sales data has been entered. You can now use this
customer as a reference customer.
It is important that the sales data and company code data have been maintained for this customer. You can maintain this data, if necessary, with the transaction XD03. This applies in particular to the reconciliation account in the company code data. You can find this account under Account info in this transaction.

**Create a Reference Customer**

To create a reference customer start transaction V-09.

From the Create Customer: Initial Screen, select the Sales areas button.

Double-click a sales area on the following dialog box.

Enter a company code. The company code must be assigned to the selected sales organization. These assignments are listed in the Enterprise IMG (transaction SPRO) under Enterprise Structure → Structure maintenance → Assignment → Sales and Distribution → Assign sales organization to company code.

Enter an account group. Use the possible entries button for this purpose (F4). Choose an account group that uses internal number assignment (see above).

Choose Enter. Maintain the data for your reference customer on the screens that follow.
Using IDocs

Use
The SAPforms interface allows you to fill an IDoc (Intermediate Document) using an electronic form. The EDI-IDoc interface in R/3 is used for this purpose.
The IDoc interface supports electronic data communication between various computers and/or systems. In our case, communication takes place between the SAPforms interface (via an electronic form) and an R/3 System.
The use of IDocs is demonstrated here using a BAPI call as an example ([Example: Create customer master record via IDoc [Seite 2280]]). This is just one way to use this function. Naturally, the entire functionality of the IDoc interface [Extern] can be leveraged with electronic forms. Reference will be made to the corresponding online documentation where necessary.

Using IDocs

Integration
IDocs are complex hierarchical data structures.
One IDoc type (structure) consists of several segment types (naming convention: E1... or Z1...), that are arranged in a hierarchy. Each segment type, in turn, consists of segment attributes or fields that play a central role when the binding is defined at a later stage.
IDoc instances are instantiations of an IDoc type. They can be compared to text files that contain rows of concrete information. At the “meta level”, these instances are divided into a control record (EDI_DC40), data record (EDI_DD40), and a status record (EDI_DS40). The control record contains administrative information, such as the recipient, sender, client, port, or message type. The status records describe the current status of the IDoc (for example “processed”). The data records contain the actual useful data. This useful data is structured by the segment types.
Prerequisites

This technology is used in the following scenario to call a BAPI in the R/3 System using an IDoc. This will be carried out using the ALE-IDoc interface, which enables IDoc types to be generated on the basis of BAPIs. Using an appropriate partner profile and port definition, the BAPI can be called from an incoming IDoc.

Generating an ALE-IDoc Interface

BAPIs are used as a basis for generating an ALE-IDoc interface. This means that the BAPI’s transfer parameters are mapped to segments in the IDoc type. When an IDoc is received, the call parameters of the BAPI are filled with the contents of the corresponding segments. Return values issued by the BAPI are stored in the status records of the IDoc.

To send IDocs to the R/3 System, you must first configure the inbound processing function in R/3. All of the generated interfaces use the BAPI process code. This code refers to the BAPI_IDOC_INPUT1 function module. This function module calls the BAPI (as a standard interface) that is to be started by the electronic form.

Defining a Partner Profile

A partner profile is an entry in which the communication parameters are defined. The values entered in the profile are primarily information that is required in the IDoc control record. The recipient (in our case, an R/3 System) must have a corresponding inbound partner profile for these values so that an assignment can be made.

See also: General Partner Profile [Extern]

Configuring a Port

IDocs can be exchanged with external systems across different paths or ports. In our case, a sender port must be defined in the control record of the IDoc and then made known to the relevant R/3 System.

See also: Configuring a Port [Extern]

Creating a Form and Processing it with the SAPforms Designer

After you have created the electronic form [Seite 2271], you must prepare it for communication with the R/3 System. This is the task of the SAPforms Designer [Seite 1735].
SAP AG
Using IDocs

Features
Processing Inbound IDocs
IDoc

Port

Partner Profile

Not OK

OK
Determine Processing
from Process Code

IDoc contains
error

Process
IDocs

After you have started and filled out your Visual Basic form, for example, an IDoc instance is generated
according to the form structure and is filled with data from the form. Following this, the IDoc is sent by
SAPforms to the R/3 System by a synchronous RFC where it is processed in accordance with the inbound
processing settings.
In many applications, some of the segment fields are not to be filled from the inputs in the form fields, for
example, because the BAPI requires input data that would only confuse the person filling out the form. In
this case, the data can be filled in the background. In the SENDIDOC.VBP model, which is stored in the
...\SAPforms\Samples directory, this function is carried out by the CUSTOMER.TXT file in the same
directory and by additional VB code.
The control record must also be filled with data. To do so, you must add VB code to the electronic form.

April 2001

2270


Designing an Electronic Form for Use as an IDoc

Purpose
You want to use an electronic form to call a BAPI. This is necessary, for example, if the user does not have access to SAPGUI. As the user has no technical know-how at all (and perhaps do not even know that they have called an R/3 System), a potential error message should not go back to the user, as would be the case with direct calling of a BAPI [Seite 2260].
The use of IDocs would be suitable here as a recipient can be entered. In contrast to direct calling, it is then the recipient (for example an administrator) who has to correct the error.
This scenario is just one example of how IDocs can be used in conjunction with SAPforms. Naturally, the entire functionality of the IDoc interface [Extern] can be leveraged with electronic forms.

In order to use IDocs, an online connection to the R/3 System is required.

Process Flow
In the process described here, it is assumed that you are using Visual Basic. This also applies to the individual sections in this chapter. However, you can also design forms with MS Outlook [Seite 1731].
To call a BAPI from an electronic form using an IDoc, you must carry out the following steps:

In R/3:
Generate the IDoc using a BAPI
Download the IDoc and its segments
  Generating an IDoc [Seite 2273]
  Defining a partner profile
Defining a port
  Defining a Partner Profile and Port [Seite 2275]

In Visual Basic:
Design the electronic form
  Call the SAPforms Designer as an add-in or
  Use the Submit control.
  Designing forms with Visual Basic [Seite 1730]

In the SAPforms Designer:
  Maintain the plug-ins.
  Open the IDoc structure
  Maintaining Plug-Ins and Opening a Structure: IDoc [Seite 2277]
  Define the binding
  Modify the VB code
  Generate
  Defining and Generating a Binding: IDoc [Seite 2278]
Result
Once your form has been designed and processed by the SAP Forms Designer, you can make the form accessible to the end users.
Generating an IDoc

Prerequisites
You know the BAPI that you want to call with an IDoc.

Procedure

Generating the IDoc
Choose Tools → Business Framework → BAPI Development → ALE Development → BAPI → Maintain ALE Interface
The ALE Interface for BAPI screen is then displayed.
Enter the object type of your BAPI in the BOR object field.
Enter the method you want to execute in the Method field.
Choose Create.
The Message Type dialog box appears.
Enter a name for your message type and choose Continue.
The Enter interface parameters dialog box appears. You must modify some of the default values displayed here.
Enter the abbreviation for your development class.
Delete the entries in the ALE outbound frame.
Choose Continue.
The Enter Change Request dialog box appears.
Enter the change request and choose Continue.
The ALE interface is then generated. The Results screen is then displayed for you to check the result.

Downloading the IDoc
You can export meta information of an IDoc to your PC. This allows you to access the IDoc description easily (if you are using the SAPforms Designer). This step, however, is not essential, because you can log on to the R/3 System directly from the SAPforms Designer and then choose your IDoc description.
Choose Tools → Business Communication → IDoc → IDoc Basis → Documentation → IDoc Type (Parser)
The Documentation IDoc Record Types and IDoc Types (Parser) screen is then displayed.
Enter the generated Basis IDoc type in the Basis types field.
Delete the entry “46B” from the SegmentRelease field.
If you are using R/3 Release 4.5A or higher, select Extended grammar. This provides better support for determining the segment names.
Leave the other default settings as they are.
Choose Execute.
A list containing the IDoc is then displayed.

Choose List → Download.

The Save List in File... dialog box is then displayed.

Leave the default selection as it is and confirm it.

The Transfer List to a Local File dialog box is then displayed.

Enter the path and file name (extension *.DES) and choose Transfer.
Defining a Partner Profile and Port

Prerequisites
You have successfully generated an IDoc interface. Generating an IDoc [Seite 2273]

Procedure
Defining a partner profile
Choose Tools → Business Communication → IDoc → IDoc Basis → IDoc → Partner Profile

The Partner profile screen is then displayed.

Choose Create.

Enter a name or a number in the partner number field.

This name must reference an existing object for each partner type. It must also be identical to the partner number in the control record of the IDoc.

Choose the partner type Logical System (LS).

Logical systems are maintained with the transaction SALE.

Set the partner status to active (A) (tab page Classification).

Maintain the Type, Language, and User fields (Tab page Postprocessing: permitted agent).

Choose Save.

Then choose the function Create Inbound Parameter in the Inbound parameter table.

The Partner profiles screen appears. Inbound parameter.

Enter your message type in the Message type field and BAPI in the Process code field.

Choose Save.

Defining a port
You must define a sender port so that the SAPforms interface and the R/3 System can communicate. This information must also appear later in the control record of the IDoc.


The Ports in IDoc Processing screen is then displayed.

Select File in the structure tree and choose Create.

Enter the name of your port in the Port field.

This name is essentially irrelevant. However, it must be identical to the port name in the control record of the IDoc.

Enter a name in the Description field.

Select IDoc record types SAP Release 4.x.

Choose Save.
Result
The partner profile and port are now defined.
Maintaining Plug-Ins and Opening a Structure: IDocs

Prerequisites
You have created a form with Visual Basic. You have launched the SAPforms Designer as a Visual Basic add-in and are in the main window. The form controls are shown in the table displayed in the bottom control.

Procedure

Maintain Plug-Ins
Choose Tools → Options.

The Options dialog box of the SAPforms Designer is then displayed.

Select the Plug-Ins tab page.

Enter SAPforms Data Plug-In for IDoc [Seite 2006] in the SAPData Plug-In.

The value in the SAPforms Plug-In box is set by default according to the form development environment you are using. If you are using the stand-alone version of the SAPforms Designer, you must also maintain this plug-in.

Enter SAPforms Transmit Plug-In for RFC [Seite 2125] in the SAP Transmit Plug-In.

Exit the dialog box.

Opening a Structure
Choose File → Open SAP Structure and then the entry for IDoc structures.

A dialog box is then displayed in which you can choose whether you want to retrieve an IDoc description from the R/3 System via RFC (From R/3) or import it from a local file (From File).

You can only import the description from a local file if you have already exported it from the R/3 System.

Accessing the local file is the faster option. For more information see generating IDocs [Seite 2273]

When you select this the top control is divided in two. You can now display an instance of your IDoc structure in the right-hand control.

Choose File → Open IDoc Instance.

Result
The segment description of the IDoc is exported from R/3 and displayed as a tree in the top left-hand control in the main SAPforms Designer window.
Defining and Generating a Binding: IDocs

Prerequisites
You have maintained the plug-ins. You are in the main window of the SAPforms Designer. The top level of the IDoc structure (control record, data record, status record, segment definitions) are displayed in the top control of the Designer. The form controls are shown in the table displayed in the bottom control.

Procedure
Defining and Generating a Binding
You must now assign the individual segment attributes of the IDoc type to the fields in the current Visual Basic form (“binding definition”). This assignment is stored as part of the form and is used at runtime to transfer data from the form to the R/3 System.

Expand the segment types (E1..., bzw. Z1...).

The segment attributes are displayed
Assign (using “drag and drop”) the segment attributes that you want to fill with the form to the form controls.

You can release the binding by selecting the required row in the table and choosing Binding → Release.

Choose File → Generate.

The General tab on the SAPforms Options screen is then displayed.

Select Insert initialization code in Load()- method and button to trigger the insert action.

Choose OK.

This is only recommended the first time you generate your form and if the form does not already have these buttons. Otherwise, you might receive an error when compiling the form because some of the declarations and event procedures are contained twice in the form. You must remove duplicate code from the form by hand.

Choose File → Exit to exit the SAPforms Designer.

Modify the VB code.

Modifying the VB code.
In addition to the values specified in the binding, the control record of the IDoc must also be filled with data. As already mentioned, the control record contains the header data of the IDoc. The necessary IDoc header data includes:

Message type
Sender partner number
Sender partner type
Sender port
Receiver partner number
Receiver partner type
Receiver port
These values can be set with the following VB code:

```vbnet
objSAPForm.SAPData.Value("EDI_DC.MESTYP") = "+" 'message type
objSAPForm.SAPData.Value("EDI_DC.SNDPRN") = "+" 'send partner
objSAPForm.SAPData.Value("EDI_DC.SNDPRT") = "LS" 'send partner type
objSAPForm.SAPData.Value("EDI_DC.SNDPOR") = "+" 'sender port
objSAPForm.SAPData.Value("EDI_DC.RCVPRT") = "LS" 'receiver partner
objSAPForm.SAPData.Value("EDI_DC.RCVPRN") = "*" 'receiver partner type
```

A value is not required for the receiver port. Make sure to use the names you have selected (for "*") for the message type, sender partner/receiver partner (partner number) and sender port (port).

Generating an IDoc [Seite 2273]
Defining a Partner Profile and Port [Seite 2275]

Result
Your form is now complete. You can now start it in Visual Basic by choosing Run or compile an EXE file.
Example: Creating a Customer Master Record via an IDoc

A model VB project SENDIDOC.VBP for calling the BAPI_Customer_CreateFromData BAPI of the object type Customer (KNA1) via an IDoc is stored in the ../SAPforms/Samples directory. This method is used to create a customer master record in the R/3 System.

Process Flow

To use this example, you need to carry out the following steps:

Generate an IDoc. Generating an IDoc [Seite 2273]

Enter KNA1 as the BOR object and the method Createfromdata.

Enter your message type: Customercreate

Define the partner profile and port. Defining a Partner Profile and Port [Seite 2275]

Enter Createcust in the Partner number field.

Name your port Portcucr.

Enter $Tmp (local object) as the development class.

Start Visual Basic and load the SENDIDO.VBP project from the ../SAPforms/Samples directory.

The IDoc structure is displayed in the top control. The form controls are shown in the table displayed in the bottom control. The binding is already defined and generated. The steps for Maintaining Plug-Ins and Opening a Structure [Seite 2277] and Defining and Generating a Binding [Seite 2278], therefore, have been completed.

All of the segment attributes required to execute the BAPI are listed in the table below:

<table>
<thead>
<tr>
<th>Name</th>
<th>Segment Attribute</th>
<th>Values</th>
</tr>
</thead>
<tbody>
<tr>
<td>Name</td>
<td>E1BPKNA101000.NAME</td>
<td>Any</td>
</tr>
<tr>
<td>First name</td>
<td>E1BPKNA101000.FIRST_NAME</td>
<td>Any</td>
</tr>
<tr>
<td>Street</td>
<td>E1BPKNA101000.STREET</td>
<td>Any</td>
</tr>
<tr>
<td>Postal code</td>
<td>E1BPKNA101000.POSTL_CODE</td>
<td>5-digit number</td>
</tr>
<tr>
<td>City</td>
<td>E1BPKNA101000.CITY</td>
<td>Any</td>
</tr>
<tr>
<td>Reference customer</td>
<td>E1BPKNA102000.REF_CUSTMR</td>
<td><strong>(*)</strong></td>
</tr>
<tr>
<td>Country</td>
<td>E1BPKNA101000.COUNTRY</td>
<td>US**(*)**</td>
</tr>
<tr>
<td>Language</td>
<td>E1BPKNA101000.LANGU</td>
<td>US</td>
</tr>
<tr>
<td>Sales organization</td>
<td>E1BPKNA102000.SALESORG</td>
<td><strong>(*)</strong></td>
</tr>
<tr>
<td>Division</td>
<td>E1BPKNA102000.DIVISION</td>
<td><strong>(*)</strong></td>
</tr>
<tr>
<td>Distribution channel</td>
<td>E1BPKNA102000.DISTR_CHAN</td>
<td><strong>(*)</strong></td>
</tr>
</tbody>
</table>

Not all attributes are filled by the form. Instead, they are filled with initial values in the background by an import routine in VB code. These values are stored in the CUSTOMER.TXT file in the directories ../SAPforms/Samples or ../SAPforms/Forms. It is important that the file is in the same directory as the EXE file read for this purpose.

It concerns the data for state, reference customer sales organization, division and distribution channel (**(*)**). Except for state this data is system-specific and must be determined separately. In order for the example to work, you must maintain the
Example: Creating a Customer Master Record via an IDoc

appropriate data for your system in the file CUSTOMER.TXT. See also: Determining a Reference Customer [Seite 2282].

Modify the VB code. Define and generate binding [Seite 2278]

This has already been done in our example.

Test your form by choosing Run in Visual Basic or compile and start an EXE file.

Enter meaningful data (see above) and choose Execute. You are then logged on to your R/3 System automatically. After a short period of time, the message sendidoc "The IDoc has been sent successfully" is displayed.

This means that the IDoc generated by the form was successfully sent to the R/3 System.
Determining a Reference Customer

You must know the (or a) reference customer before you can execute the BAPI Customer CreateFromData BAPI. This BAPI is used in the following examples provided in the SAPforms documentation:

Create customer master record [Seite 2264]
Create customer master record via IDoc [Seite 2280]

The reference customers used in your company (and therefore the other sales data) are normally known. If this is not the case, you can determine the reference customers from the internal number ranges. You can only do so, however, if the reference customer was created with the internal number assignment function. In other words, there are two ways of finding a suitable reference customer:

Determining an existing reference customer

Creating a new reference customer

From a business perspective, it makes more sense to create a separate reference customer for the form application. To do so, use the transaction V-09.

Procedure

Determine the Internal Number Ranges

Start transaction SPRO and choose Basic functions → Enterprise IMG → Display. The Structure screen is then displayed: Enterprise IMG.

Expand the tree structure by choosing Financial Accounting → Accounts Receivable and Accounts Payable → Customer Accounts → Master Records → Preparations for Creating Customer Master Records Create and choose number ranges for customer accounts. The Customer Number Ranges screen is then displayed.

Choose Display intervals. The Display Number Range Intervals screen is then displayed. All of the number ranges that have been defined for customers are listed here. Number ranges with internal number assignment are those where the Ext box is not selected. Make a note of the internal number ranges.

Determine the Reference Customer

Choose the transaction VD03. The Display Customer: Sales Area Data Initial Screen is then displayed. Sales Area Initial Screen.

Click the F4 Help in the Customer field and choose Customers per sales group on the window that then appears. This dialog box is displayed as standard the first time you click the possible entries button. Your selection is stored, which means that this dialog box (in this case Customers per sales group) will be displayed in future whenever you choose the possible entries help.

Choose the multiple selection function for the Customer field on this dialog box. Enter the number range intervals here.

Exit the multiple selection by choosing Copy. Then choose Continue. A list of customers and associated sales data is then displayed. Sales organization, division and distribution channel.

Choose a customer for which all of the sales data has been entered. You can now use this customer as a reference customer.
Determining a Reference Customer

It is important that the sales data and company code data have been maintained for this customer. You can maintain this data, if necessary, with the transaction XD03. This applies in particular to the reconciliation account in the company code data. You can find this account under Account info in this transaction.

Create a Reference Customer

To create a reference customer start transaction V-09.

From the Create Customer: Initial Screen, select the Sales areas button.

Double-click a sales area on the following dialog box.

Enter a company code. The company code must be assigned to the selected sales organization. These assignments are listed in the Enterprise IMG (transaction SPRO) under Enterprise Structure → Structure maintenance → Assignment → Sales and Distribution → Assign sales organization to company code.

Enter an account group. Use the possible entries button for this purpose (F4). Choose an account group that uses internal number assignment (see above).

Choose Enter. Maintain the data for your reference customer on the screens that follow.
Web Form Integration

Use
Web forms allow users to communicate with the R/3 System via the Internet using their Web browser.
Using the Web forms you can
Start workflows (Starting workflows with Web forms [Seite 2285])
Process work items (Processing Work Items as a Web Form [Seite 2297])

Prerequisites
To use this technology, you must
Configure an ITS (Internet Transaction Server [Extern])
Install the SAP@Web Studio

The SAP@Web Studio is part of the ITS installation and runs on NT4.0 or higher

Features
Unlike electronic forms, Web forms are not integrated using the SAPforms interface.
Instead, an ITS service is developed from a Web transaction generated previously from a task in the R/3 System. This ITS service consists of HTML templates (among other things) that can then be revised using suitable tools.
Starting Workflows with Web Forms

Use
A Web form is completed by an employee or customer on the Internet and sent to the R/3 System. The task (only workflow templates or standard tasks are supported) associated with this form is then started in the R/3 System. The corresponding import elements of the container for this task are then supplied with input data from the form.

Features

When the Web form is processed, a Web-enabled R/3 transaction (“Web transaction”) is called from a Web page. The transaction is actually called via the Internet Transaction Server (ITS). The ITS communicates with the R/3 System directly via the SAPGUI interface to support Web transactions.
Designing a Web Form to Start a Task

Purpose
You want to design a Web form to start a workflow or task in the R/3 System.

Prerequisites
A namespace template and a development class must be defined in Customizing. To do so, carry out the activity Name range maintenance for task-related transactions in Customizing for SAP Business Workflow.

Process Flow
You define the workflow or task that is to be started by a Web form. If elements in the task container are to be filled with values from the form, they must be identified as import elements

- Defined with a data type reference to an ABAP dictionary field.

- Multiple-line elements or elements that reference an object or structure are not supported at present.

You then generate a Web transaction in the system that allows users to fill the import elements of the task container and start the task.

A Web transaction is an R/3 transaction that must be specially designed so that it can be called from an HTML page. The name of this transaction is generated and consists of the namespace template that you defined in Customizing and the task ID.

Following this, you develop an ITS service from this transaction in the SAP@Web Studio.

An ITS service essentially comprises an ITS service description, HTML templates (one for every screen in the R/3 transaction) and language-support files. The Studio is a development tool linked to the R/3 System that runs on your PC. The Studio assumes that the ABAP transaction already exists.
Defining a Task to be Started with a Web Form

Prerequisites
You are familiar with the procedure for defining tasks.
You want to define the task (single or multistep task) that is to be started by the Web form.

The part of the task definition that is relevant to forms essentially entails extending the container definition to include the elements that are to appear as fields on your form.

Procedure
Defining a Task
To create the task, choose Tools → Business Workflow → Development → Definition tools → Tasks/Task groups → Create.
Enter the type of task you want to create in the Task type field.

Only workflow templates (task type WS) and standard tasks (task type TS) can be used with Web forms.

Then choose Task → Create.

Define the basic data of your task.

Extending the Container Definition
Now extend the container of your task to include the elements that are to be filled via the form.

To launch the container editor, choose Goto → Container.
The container already contains the standard system elements. To add elements to the container, choose Edit → Create.

You can only transfer single values (not object references) from the form. If you want to add container elements for this purpose, therefore, you must always create them as single-line elements with a data type reference to a table field in the dictionary.

Create a suitable container element for all of the fields you want to fill in the form. Define these elements as import elements.

When generating the HTML template in the SAP@WebStudio the container element name is also used as the name of the input field. It is therefore recommended that you choose “meaningful” names (although the HTML templates can of course be processed at any time).

Note that when later changes are made to the name of container elements, the web transaction must be regenerated.
Further Procedure

Complete the task definition:

Declare the possible agents [Extern] that are allowed to start the task by filling out and submitting the form.

Save your task definition in the development class that you entered in the Customizing activity for maintaining the namespace.

If the task is a multistep task, maintain the associated workflow definition.
Generating a Web Transaction

Prerequisites
You have defined and saved the task that is to be started or executed with the Web form.

Procedure
To generate a Web transaction from the task:
Choose Goto → Web transaction in the task definition.

You cannot generate a Web transaction from a task,
if you are in Display mode in the task definition
if you have created a new task but have not saved it yet.

The following dialog box is then displayed: Web Transaction for Task...: Change.
Choose Generate.

The system then creates a transaction including a module pool and screens.
Save your module pool in the development class that you entered in the Customizing activity for maintaining the namespace. You have also already saved the task in this development class.
Specify a change request.
The system activities (generating screens and module pool) are indicated by texts in the status line.
Confirm the messages.
The dialog box appears again. The name of the new transaction now appears on the screen.
Click Continue to exit this dialog box.

Result
A Web transaction with a module pool and four screens has now been created in the system. The various screens in this transaction are used to execute a single-step task and start a task.

If you change the task container after you have generated the Web transaction, you must delete the transaction in the Web Transaction for Task...: Change screen and regenerate it for the changes to take effect.

If you call the new transaction partly in the R/3 System, you will start a task.

Enter the name of the transaction in the OK code field in R/3.
Do not enter any data on the first screen. Choose Continue.
A screen is then displayed in which you can select the import parameters for the task container.
Choose a few entries here as a test.
Choose Transfer.
The task is now started and a work item is generated. The number of this work item is displayed on the next screen.
Background Info on the Generated Web Transactions

Namespace
Specify a namespace template in the Name range maintenance for task-related form transactions activity in Customizing for SAP Business Workflow. The namespace template must be no more than 2 characters long. The system then uses this information to create the transaction code prefix Y<namespace template> the module pool prefix SAPMY<namespace template> and reserves the corresponding namespaces.

Transaction and Module Pool Names
The Web transaction used to start a task or process a work item is given the name Y<namespace template><Task ID>H.
The name of the module pool for this transaction is SAPMY<namespace template><task ID>H.

<table>
<thead>
<tr>
<th>Namespace template</th>
<th>AB</th>
<th>Defined once in Customizing</th>
</tr>
</thead>
<tbody>
<tr>
<td>Task identifier</td>
<td>WS98700001</td>
<td>Assigned automatically by the system when the task is saved</td>
</tr>
<tr>
<td>Transaction</td>
<td>YABWS98700001H</td>
<td>Created with the Generate task function</td>
</tr>
<tr>
<td>Module pool</td>
<td>SAPMYABWS98700001H</td>
<td></td>
</tr>
<tr>
<td>Screens</td>
<td>50, 100, 150, 200</td>
<td></td>
</tr>
</tbody>
</table>

Module Pool Screens
The system creates four screens for the module pool. Matching HTML templates are then generated for these screens in the subsequent step Developing an ITS Service from a Transaction [Seite 2306].

Screen 050
This screen is filled with technical information and used internally.

Screen 100
This screen is only used if the transaction for executing a single-step task is executed. It contains the import elements of the container for the single-step task as display fields and the export elements of the task container as input fields.

Screen 150
This screen is only used if the transaction for starting a task is executed. It contains the import elements of the task container as input fields.

Screen 200
This screen contains final messages. If the procedure was successful, the number of the work item that was started is displayed here.
Developing an ITS Service from a Transaction

Prerequisites
You have created a Web transaction from a task, as described in the previous steps.

You know the task and the name of the transaction with the associated module pool. If you do not know the transaction and module pool, you can find out their names from the Repository Browser in the ABAP Workbench.

You have installed the SAP@WebStudio.

Procedure
The SAP@Web Studio contains separate documentation on using the Studio. For this reason, only information relevant to this scenario will be provided here.

Launch the SAP@Web Studio.

Create a project to store your ITS components.

Define a service for this project. Use the Service Wizard for this purpose.

Enter the name of the transaction as the Service Name.

Enter the name of the system in which the task is defined and other logon information.

Select Use Diag-Channel on the dialog box and enter the name of the transaction.

Every ITS service based on a service description, in which, for example, the R/3 System and the transaction to be used are determined. Every service description is in its own service file.

For each ITS there is also a global service description that contains the standard settings for all ITS services. These settings can be modified and/or overwritten by the individual service descriptions.

Each service description is in its own service file (ASCII file ending in srvc) in the file system of the ITS computer. The HTMLBusiness-templates that are needed for the Web transaction (HTML files with specific placeholders for R/3 data) are also located there.

The SAP@Web Studio allows the user to create and maintain service descriptions and HTMLBusiness-templates.

Create the HTML templates for the Web transaction. Use the Template Wizard for this purpose.

Enter the name of the module pool for the transaction in the Program field.

Enter 50;100;150;200 in the Screen number field.

Enter the name of the service created in Step 3 (name of the transaction) in the Service field.

Enter the 99 as the theme.
Developing an ITS Service from a Transaction

One Theme is a logical name for the optical appearance of a web transaction. The name consists of two characters (letters or numbers). In principle, as many Themes as you like can be used for a web transaction. The Theme should always be set to 99 for Web forms.

The Studio creates an HTML template file on your PC for every screen in the transaction. The name of this file appears in the object list for the service. To display the HTML source code, double-click the name of the template.

Revise the HTML templates. For more information, see Extending HTML templates [Seite 2310].

Define an ITS site for the system, if you have not already done so.

Transfer your ITS files to the ITS site (publish function).

Result

All of the files required by this service have been copied to your ITS server and, therefore, can be called when necessary.
Background Info on the Generated HTML Templates

An HTML template is created for each of the four screens. The SAP@Web Studio allows you to edit these templates so that you can design your own Web forms to match your corporate design.

The names of these HTML templates comprise the name of the module pool and the screen numbers:

```
SAPMY<namespace template><task identifier>H_<screen number>.HTML
```

The HTML templates are used as follows:

**SAPMY<namespace template><task identifier>H_50.HTML**

When you call the service from the Internet, as described in the section Starting a Task with a Web Form [Seite 2296], the fields in this template are filled but the template itself is not displayed.

Since this HTML template is never displayed, it is not edited.

**SAPMY<namespace template><task identifier>H_100.HTML**

This HTML template is the template for the form that processes a work item. As standard, the HTML template contains:

- display fields created by the import elements (that are not also export elements) of the task
- input fields created by the export elements of the task
- a button for completing processing and transferring the modified data to the task in the R/3 System.

**SAPMY<namespace template><task identifier>H_150.HTML**

This HTML template is the template for the form that starts a workflow template or a standard task. As standard, the HTML template contains:

- input fields created by the import elements of the task
- a button for completing processing and transferring the data to the task in the R/3 System.

**SAPMY<namespace template><task identifier>H_200.HTML**

This HTML template is the template for the form that acknowledges that the form has been successfully processed. As standard, the HTML template contains:

- three lines for message texts
- a button for acknowledging processing.

<table>
<thead>
<tr>
<th>Service</th>
<th>YABWS98700001H.SRVC</th>
<th>Must be created with the transaction name</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>HTML templates</strong></td>
<td></td>
<td></td>
</tr>
<tr>
<td>SAPMYABWS98700001H_50.HTML</td>
<td></td>
<td></td>
</tr>
<tr>
<td>SAPMYABWS98700001H_100.HTML</td>
<td></td>
<td></td>
</tr>
<tr>
<td>SAPMYABWS98700001H_150.HTML</td>
<td></td>
<td></td>
</tr>
<tr>
<td>SAPMYABWS98700001H_200.HTML</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Background Info on the Generated HTML Templates

April 2001
Starting a Task with a Web Form

Prerequisites
You have developed an ITS service in the SAP@Web Studio with the four HTML templates for starting tasks or processing work items, as described in the steps above. You have also edited the templates (if necessary) to design your own individual Web form.

Procedure
To start the task with the form:
Open your Internet browser.
Call the ITS service. The URL must be structured as follows:

http://<ITS-Server>/scripts/wgate/<webtransaction>/?WEB_FLAG=X~OkCode=CONT

<ITS Server> is the name of your Internet Transaction Server.

<webtransaction> is replaced with the name of your web transaction.

WEB_FLAG=X is the transaction that is executed using a Web form.

~OkCode=CONT - the initial screen (screen 50) is skipped
The form then appears with the fields for entering or displaying data.

Edit the form.
Transfer the form to the R/3 System using the function provided (usually by clicking a pushbutton).
A message is then displayed informing you of the work item number with which the task was started.
Processing Work Items as a Web Form

Use

Employees can also display their Business Workplace, with all the work items they have received, on the Internet (SAP Business Workplace Internet application component). While all of the work items displayed can also be executed in the R/3 System, only work items for tasks that fulfill certain technical conditions can be started over the Internet.

Work items for Web form tasks [Seite 2298] can always be executed on the Internet.

Features

If the work item representing the Web form task is listed in the Web-Workplace, the corresponding Web form can be processed. When the form is transferred to the R/3 System, the work item container of the relevant work item is updated. The work item is then terminated.

If the same work item would be listed in the Business Workplace in the R/3 System, it would be processed with the Web transaction that forms the basis of the Web form.
Web form tasks

Definition
Single-step task that executes the `HTMLProcess` method of the `FORM object type [Seite 74]`. The *Executable on the Internet* indicator must be selected in the properties of this task.

Use
When a Web form task is executed as a step in a workflow, a work item is displayed as standard in both the *Business Workplace* in the R/3 System and in the *Web-Workplace* of the recipients for this step. This work item can be executed from the Web-Workplace.

In this case, the work item container is processed with a Web form in the Internet browser.
Designing a Web Form to Process a Work Item

Purpose
You want to design a Web form to process a work item.

Prerequisites
A namespace template and a development class must be defined in Customizing. To do so, carry out the activity Name range maintenance for task-related transactions in Customizing for SAP Business Workflow.

Process Flow
You define a standard task that is to be executed as a Web form. This task must reference the HTMLProcess method of the FORM object type. The elements in the task container that are to appear and be processed on the form must be identified as export elements. Defined with a data type reference to a DDIC field. Multiple-line elements or elements that reference an object or structure are not supported at present. The elements in the task container that are only to be displayed on the form must be identified as import elements. Defined with a data type reference to a DDIC field. Multiple-line elements or elements that reference an object or structure are not supported at present.

You use this standard task in an activity in a workflow definition. Here, you must define the binding between the elements in the task container and the corresponding elements in the workflow container.

You can then generate a Web transaction from the task in order to execute the task, that is, that processes the export elements of the task container or displays the import elements. A Web transaction is an R/3 transaction that must be specially designed so that it can be called from an HTML page. The name of this transaction is generated and consists of the namespace template that you defined in Customizing and the task ID.

Following this, you develop an ITS service from this transaction in the SAP@WebStudio. An ITS service essentially comprises an ITS service description, HTML templates (one for every screen in the R/3 transaction) and language-support files.
Designing a Web Form to Process a Work Item

The Studio is a development tool linked to the R/3 System that runs on your PC. The Studio assumes that the ABAP transaction already exists and that you want to create (or change) the ITS service components.
Defining a Task to be Executed as a Web Form

Prerequisites
You have defined a workflow and now want to add a single-step task that can be processed with a Web form.

The part of the task definition that is relevant to forms essentially entails extending the container definition to include the elements that are to be processed as fields on your form.

Using the FORM.HTML Process object method.

Procedure

Defining the Form Task
To create the tasks, choose Tools → Business Workflow → Development → Definition tools → Tasks/Task groups → Create.

Enter the type TS for standard task in the Task type field.

Then choose Task → Create.

Define the basic data of your task.

Enter FORM as the object type and HTMLProcess as the object method.

Select the Executable on the Internet field in the task properties.

Extending the Container Definition
You can add other container elements to a container as form-specific extensions. The contents of these elements can then be displayed and edited in the fields on the form. All of the data that is to be displayed to the form user must exist as container elements in the task container.

To launch the container editor, choose Goto → Container.

The container already contains the standard system elements. To add elements to the container, choose Edit → Create.

You can only transfer single values (not object references) from the form. If you want to add container elements for this purpose, therefore, you must always create them with a data type reference to a table field in the dictionary.

For each field that is to be displayed on the form, create a corresponding container element and identify it as an import element.

that is to be entered on the form, create a corresponding container element and identify it as an export element.

that is to be displayed and overwritten on the form, create a corresponding container element and identify it as an import and export element.
Defining a Task to be Executed as a Web Form

When generating the HTML template in the SAP@WebStudio the container element name is also used as the name of the display-input field.

It is therefore recommended that you choose "meaningful" names (although the HTML templates can of course be processed at any time).

Note that when later changes are made to the name of container elements, the web transaction must be regenerated.

Further Procedure

Complete the task definition:

Declare the possible agents [Extern] that are allowed to start the task by filling out and submitting the form.

Save your task definition in the development class that you entered in the Customizing activity for maintaining the namespace.
Generating a Web Transaction

Prerequisites
You have defined and saved the task that is to be started or executed with the Web form.

Procedure
To generate a Web transaction from the task:
Choose Goto → Web transaction in the task definition.

You cannot generate a Web transaction from a task,
if you are in Display mode in the task definition
if you have created a new task but have not saved it yet.

The following dialog box is then displayed: Web Transaction for Task...: Change.

Choose Generate.

The system then creates a transaction including a module pool and screens.

Save your module pool in the development class that you entered in the Customizing activity for maintaining the namespace. You have also already saved the task in this development class.

Specify a change request.

The system activities (generating screens and module pool) are indicated by texts in the status line.

Confirm the messages.

The dialog box appears again. The name of the new transaction now appears on the screen.

Click Continue to exit this dialog box.

Result
A Web transaction with a module pool and four screens has now been created in the system. The various screens in this transaction are used to execute a single-step task and start a task.

If you change the task container after you have generated the Web transaction, you must delete the transaction in the Web Transaction for Task...: Change screen and regenerate it for the changes to take effect.

If you call the new transaction partly in the R/3 System, you will start a task.

Enter the name of the transaction in the OK code field in R/3.

Do not enter any data on the first screen. Choose Continue.

A screen is then displayed in which you can select the import parameters for the task container.

Choose a few entries here as a test.

Choose Transfer.
The task is now started and a work item is generated. The number of this work item is displayed on the next screen.
Background Info on the Generated Web Transactions

Namespace
Specify a namespace template in the Name range maintenance for task-related form transactions activity in Customizing for SAP Business Workflow. The namespace template must be no more than 2 characters long.
The system then uses this information to create
the transaction code prefix \( Y<\text{namespace template}> \)
the module pool prefix \( \text{SAPMY}<\text{namespace template}> \)
and reserves the corresponding namespaces.

Transaction and Module Pool Names
The Web transaction used to start a task or process a work item is given the name \( Y<\text{name space template}><\text{Task ID}>H. \)
The name of the module pool for this transaction is \( \text{SAPMY}<\text{namespace template}><\text{task ID}>H. \)

<table>
<thead>
<tr>
<th>Namespace template</th>
<th>AB</th>
<th>Defined once in Customizing</th>
</tr>
</thead>
<tbody>
<tr>
<td>Task identifier</td>
<td>WS98700001</td>
<td>Assigned automatically by the system when the task is saved</td>
</tr>
<tr>
<td>Transaction</td>
<td>YABWS98700001H</td>
<td>Created with the Generate task function</td>
</tr>
<tr>
<td>Module pool</td>
<td>SAPMYABWS98700001H</td>
<td></td>
</tr>
<tr>
<td>Screens</td>
<td>50, 100, 150, 200</td>
<td></td>
</tr>
</tbody>
</table>

Module Pool Screens
The system creates four screens for the module pool. Matching HTML templates are then generated for these screens in the subsequent step Developing an ITS Service from a Transaction [Seite 2306].

Screen 050
This screen is filled with technical information and used internally.

Screen 100
This screen is only used if the transaction for executing a single-step task is executed. It contains the import elements of the container for the single-step task as display fields and the export elements of the task container as input fields.

Screen 150
This screen is only used if the transaction for starting a task is executed. It contains the import elements of the task container as input fields.

Screen 200
This screen contains final messages. If the procedure was successful, the number of the work item that was started is displayed here.
Developing an ITS Service from a Transaction

Prerequisites
You have created a Web transaction from a task, as described in the previous steps.
You know the task and the name of the transaction with the associated module pool. If you do not know the transaction and module pool, you can find out their names from the Repository Browser in the ABAP Workbench.
You have installed the SAP@WebStudio.

Procedure
The SAP@Web Studio contains separate documentation on using the Studio. For this reason, only information relevant to this scenario will be provided here.
Launch the SAP@Web Studio.
Create a project to store your ITS components.
Define a service for this project. Use the Service Wizard for this purpose.

- Enter the name of the transaction as the Service Name.
- Enter the name of the system in which the task is defined and other logon information.
- Select Use Diag-Channel on the dialog box and enter the name of the transaction.

Every ITS service based on a service description, in which, for example, the R/3 System and the transaction to be used are determined. Every service description is in its own service file.

For each ITS there is also a global service description that contains the standard settings for all ITS services. These settings can be modified and/or overwritten by the individual service descriptions.

Each service description is in its own service file (ASCII file ending in srvc) in the file system of the ITS computer. The HTMLBusiness-templates that are needed for the Web transaction (HTML files with specific placeholders for R/3 data) are also located there.

The SAP@Web Studio allows the user to create and maintain service descriptions and HTMLBusiness-templates.
Create the HTML templates for the Web transaction. Use the Template Wizard for this purpose.

- Enter the name of the module pool for the transaction in the Program field.
- Enter 50;100;150;200 in the Screen number field.
- Enter the name of the service created in Step 3 (name of the transaction) in the Service field.
- Enter the 99 as the theme.
One Theme is a logical name for the optical appearance of a web transaction. The name consists of two characters (letters or numbers). In principle, as many Themes as you like can be used for a web transaction. The Theme should always be set to 99 for Web forms.

The Studio creates an HTML template file on your PC for every screen in the transaction. The name of this file appears in the object list for the service. To display the HTML source code, double-click the name of the template.

Revise the HTML templates. For more information, see Extending HTML templates [Seite 2310].

Define an ITS site for the system, if you have not already done so.

Transfer your ITS files to the ITS site (publish function).

Result
All of the files required by this service have been copied to your ITS server and, therefore, can be called when necessary.
Background Info on the Generated HTML Templates

An HTML template is created for each of the four screens. The SAP@Web Studio allows you to edit these templates so that you can design your own Web forms to match your corporate design.

The names of these HTML templates comprise the name of the module pool and the screen numbers:

SAPMY<namespace template><task identifier>H_<screen number>.HTML

The HTML templates are used as follows:

SAPMY<namespace template><task identifier>H_50.HTML

When you call the service from the Internet, as described in the section Starting a Task with a Web Form [Seite 2296], the fields in this template are filled but the template itself is not displayed.

Since this HTML template is never displayed, it is not edited.

SAPMY<namespace template><task identifier>H_100.HTML

This HTML template is the template for the form that processes a work item. As standard, the HTML template contains:

- display fields created by the import elements (that are not also export elements) of the task
- input fields created by the export elements of the task
- a button for completing processing and transferring the modified data to the task in the R/3 System.

SAPMY<namespace template><task identifier>H_150.HTML

This HTML template is the template for the form that starts a workflow template or a standard task. As standard, the HTML template contains:

- input fields created by the import elements of the task
- a button for completing processing and transferring the data to the task in the R/3 System.

SAPMY<namespace template><task identifier>H_200.HTML

This HTML template is the template for the form that acknowledges that the form has been successfully processed. As standard, the HTML template contains:

- three lines for message texts
- a button for acknowledging processing.

<table>
<thead>
<tr>
<th>Service</th>
<th>YABWS98700001H.SRVC</th>
<th>Must be created with the transaction name</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>HTML templates</strong></td>
<td>SAPMYABWS98700001H_50.HTML</td>
<td></td>
</tr>
<tr>
<td></td>
<td>SAPMYABWS98700001H_100.HTML</td>
<td></td>
</tr>
<tr>
<td></td>
<td>SAPMYABWS98700001H_150.HTML</td>
<td></td>
</tr>
<tr>
<td></td>
<td>SAPMYABWS98700001H_200.HTML</td>
<td></td>
</tr>
</tbody>
</table>
Extending HTML Templates

HTML Business

The Internet Transaction Server [Extern] (ITS) requires an HTMLBusiness-template for each R/3 transaction screen. An HTMLBusiness-template is an HTML page in which HTMLBusiness-commands are embedded. The ITS generates a page of pure HTML for each template.

HTMLBusiness

Is not a programming language

Is an extension of HTML for inserting data from the R/3 System into HTML pages

Supports the transfer of data from web forms to the R/3 System

Does not require client extension

Properties

HTMLBusiness-commands are bracketed with ` (backtick) when mixed in with "normal" HTML.

There is no difference between upper and lower case in an HTMLBusiness-expression.

There must be at least one character between two consecutive HTMLBusiness-commands enclosed by ` (backtick).

The SAP@Web Studio shows the HTMLBusiness-orders in blue.

All names that occur in an HTMLBusiness-expression are interpreted as names of an R/3 screen field.

If an HTMLBusiness-expression consists of only one field name, this expression is replaced at runtime with the value of the R/3 screen field with the same name.

If a field with the addition .LABEL is used in an HTMLBusiness-expression, this expression is replaced in the R/3 screen with the field indicator of the same name.

As well as .LABEL, the following additions also exist, which can be used with every screen field that is ready for input.

<table>
<thead>
<tr>
<th>Properties</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>MAXSIZE</td>
<td>Maximum entry length of a screen field</td>
</tr>
<tr>
<td>VISSIZE</td>
<td>Maximum number of characters visible in a screen field</td>
</tr>
</tbody>
</table>

Programming Function Keys

If you want to design your form in such a way that inputs are not made directly in a field but with buttons, you must edit the HTML template as shown in the following example:

The standard task for approving an application contains the approval status element in its container. This element is marked as an import and export element and is defined with a data type reference to a Char1 field (for example SYST-INPUT).

The HTML template for screen 100, therefore, contains the Approval status input field and the Transfer button as standard.

The functionality you want to provide on the form will allow users to set the approval status to approved or rejected using one of two buttons and therefore end processing.

The Approval status input field and the Transfer pushbutton are to be removed.

The lines that generate this input field and button, therefore, must be deleted or commented out in the HTML template source code.

```
`approval status010100.label`
</INPUT TYPE="text" name="approval status010100" VALUE="`approval status010100`" maxlength="1" size="1">
</INPUT TYPE="submit" name="~OkCode(STRT)" value="`START.label`">
```
The following lines are then added to the coding:

```html
<form ACTION="\`wgateURL()\" METHOD="post">
  <INPUT TYPE="submit" name="~OKCode(STRT),approval status010100=A" value="approve">
  <INPUT TYPE="submit" name="~OKCode(STRT),approval status010100=R" value="approve">
</form>
```

This code creates the two buttons and labels them *Accept* and *Reject*. The value of the *approval status* container element is then set to *A* or *R*, depending on the button selected, and returned to the R/3 System. If the button text is to depend on the logon language, you must use language resources.

**Integration of Possible Entries**

By inserting a command it is now also possible in the web to use an input help for particular field types.

```html
<input type="text" size="10" maxlength="10"
  name="\`LASTDAY1_TEST015004.name`1"
  value="\`LASTDAY1_TEST015004.value`">\`assert(LASTDAY1_TEST015004.name)\`
  <INPUT TYPE="SUBMIT" NAME="~searchhelp(`LASTDAY1_TEST015004.name`)" VALUE="?">
```

**Simplification of Process Flow for Work Item Execution**

You can accelerate and simplify the process flow of execution by modifying the HTML template for the execution of a work item in the web described below. The Web Workplace is displayed directly, for example, if you cancel an activity.

```html
Template 50:
Instead of the old <form...after <body>>:
  <form action="\`wgateURL()\" method="post">
</form>
```

```html
Template 100:
After </title>, before </head>:
  <script language="JavaScript">
    function logoff()
    {
      hiddenConfirm = new Image();
      hiddenConfirm.src = "\`wgateURL(~OKCode="/NEX")\";
    }
    function return_bwsp()
    {
      parent.location.href="\`~ret_url`&BW02_1400-CHOICE=BAIN&~okcode=WORK&~confirm_url=`~confirm_url`&~dec_state=1";
    }
  </script>
At the start, directly after <body ...>: Replace <form method="post"
  action="\`wgateURL()\""> with:
  <FORM NAME="theform" METHOD=POST
  ACTION="\`wgateURL(~OKCode="WORK")`&~dec_state=0&~ret_url=`~ret_url`&~confirm_url="\`wgateURL()\"">
Before the cancel button: Instead of <input type=submit name="~okcode="/NEX"
  value="\`CANCEL.label\"">
</form>
```
Extending HTML Templates

Template 200:
Before </head>:
<script language="JavaScript">
function logoff()
{
    hiddenConfirm = new Image();
    hiddenConfirm.src = ":wgateURL(~OKCode="/NEX")";
}
function return_bwsp()
{
    parent.location.href="~ret_url`&BW02_1400-CHOICE=BAIN&~OKCode=WORK`&~confirm_url=`~confirm_url`&~dec_state=1`;
}
</script>
In the <form></form>-blocks delete the following line:
<input type=submit name="~okcode=/NEX" value="`SUBMIT.label`">
After </head>:
`if (~dec_state == "0")`
<body onload="return_bwsp()" BGCOLOR="#ffffff" LINK=BLUE VLINK=BLUE>
</body>
`elseif (~dec_state == "1")`
<body onload="logoff()" BGCOLOR="#ffffff" LINK=BLUE VLINK=BLUE>
Before </html>:
Adding tasks to a workflow definition

Prerequisites
You have defined a Web form task [Seite 2298], generated a Web transaction for it, and developed an ITS service in the SAP@WebStudio with the four HTML templates, as described in the steps above. You have also edited the templates (if necessary) to design your own individual Web form.

Procedure
The workflow definition for executing your form application is processed as follows:
Add a step that represents an activity in which your Web form task is referenced.
Define the binding between the workflow container and the import elements in the task container
   (Remember: These are the elements that are displayed on the form.)
Define the binding between the export elements in the task container and the workflow container.
   (Remember: These are the elements that can be entered in the form.)
Delete all of the binding definitions that may have been created automatically and that refer to
   the element _WI_Object_ID.

If data was changed in the form application and copied to the workflow container, you must define the following steps (condition, multiple condition) to evaluate this information.
Processing a Work Item on the Internet

Prerequisites
The workflow system has created a work item that represents the Web form task [Seite 2298]. This work item is displayed in the Business Workplace of its recipients.

Procedure
To process this work item as a Web form on the Internet:
Open your Internet browser.
Call the ITS service BWSP (Web-Workplace).

You can then process the work item from your Web-Workplace.

Your Business Workplace on the Internet is only a “mirror” of your Business Workplace in the R/3 System. In other words, you could also execute the work item in the R/3 System.

In this case, it is not processed with a Web form but with the Web transaction that was generated from the task.
Creating a Customer Master Record on the Internet (BC-BMT-WFM SAPforms)

Use
This Easy Web Transaction demonstrates how the workflow functionality is integrated in the Web using the SAPforms [Seite 1718] interface. This is illustrated by creating a customer master record. After a workflow has been started with a Web form (create customer master record), a work item is generated (check creation) which, in turn, is processed with a Web form. Unlike other Easy Web Transactions, two independent services are required here and must be developed separately. The first service starts a workflow, while the second executes a work item. Using both services in one workflow is simply one possibility here and is not mandatory.

You can use the Easy Web Transaction for training purposes or as a template to create your own Easy Web Transaction. This Easy Web Transaction only represents a simplified version of the Create customer master record functionality. First of all, the Web form that starts the workflow does not contain all of the options you can use to create a customer. Secondly, the flow logic of the workflow is extremely basic.

Type of Internet Application Component
Intranet application.

Prerequisites
To allow the workflow WS20000102 to be started on your system and the task TS20000256 to be executed, you should declare both as a general task. In this way, you can ensure that everyone can start the workflow. Since the (default) agent responsible for the task TS20000256 is the workflow system variable _WF_INITIATOR, the person who entered the data is automatically the person who checks it. This allows you to work through this scenario with only one user logged on to the system.

Standard Settings and Preassigned Data
When you work with the Web form for the workflow WS20000102, the following fields can be filled:

<table>
<thead>
<tr>
<th>Address Data</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Form of address</td>
<td>Selection field</td>
</tr>
<tr>
<td>First name</td>
<td>Optional</td>
</tr>
<tr>
<td>Last name</td>
<td>Required field</td>
</tr>
<tr>
<td>Street</td>
<td>Optional</td>
</tr>
<tr>
<td>Postal code</td>
<td>Required field</td>
</tr>
<tr>
<td>City</td>
<td>Required field</td>
</tr>
<tr>
<td>Country</td>
<td>Selection field</td>
</tr>
<tr>
<td>Language</td>
<td>Selection field</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Communication Data</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>1. Telephone number</td>
<td>Optional</td>
</tr>
<tr>
<td>2. Telephone number</td>
<td>Optional</td>
</tr>
<tr>
<td>Fax number</td>
<td>Optional</td>
</tr>
<tr>
<td>E-mail (administrator)</td>
<td>Optional</td>
</tr>
</tbody>
</table>
Creating a Customer Master Record on the Internet (BC-BMT-WFM SAPforms)

Organizational Data

<table>
<thead>
<tr>
<th>Field</th>
<th>Must be preassigned</th>
</tr>
</thead>
<tbody>
<tr>
<td>Sales organization</td>
<td></td>
</tr>
<tr>
<td>Distribution channel</td>
<td></td>
</tr>
<tr>
<td>Division</td>
<td></td>
</tr>
<tr>
<td>Reference customer</td>
<td></td>
</tr>
</tbody>
</table>

You must define a preassigned value for the sales organization, distribution channel, division, and reference customer fields. In order to do so, you must enter your company-specific data in the HTML template SWU5WS20000102H_150 in the SAP@WebStudio. The existing data in the template only serves as a placeholder.

To define default organizational data, open the HTML template SWU5WS20000102H_150 (service name TS_WS20000102H) in the SAP@WebStudio. The template coding (with comments) is displayed in the right-hand SAP@WebStudio window. The section of the source text in which you can change the organizational data starts after the comment CUSTOMIZING SECTION - BEGIN.

If your sales organization has the value “1234”, for example, enter this value in the template by changing the line after the first PLEASE CHANGE THE VALUE... as follows:

Before: ... "SALESORGANIZATION150150" value = "0001"...

After: ... "SALESORGANIZATION150150" value = "1234"...

You can change the distribution channel, division, and reference customer values in the same way by modifying the next three sections that start with PLEASE CHANGE THE VALUE... .

Features

Options

Firstly, the flow logic of the workflow can be extended. In this way, a rejected customer master record could be returned to its creator for resubmission. In this case, the existing workflow must be copied. Secondly, the Web form that starts the workflow can be extended to include the input options that are also available for creating a new customer master record but were not included in this Easy Web Transaction. For more information on controlling workflows with Web forms, please refer to Web form integration.

Service Name

The service names of this Internet Application Component are TS_WS20000102H and TS_TS20000256H. All of the associated files are stored under these service names in the SAP@Web Studio.

R/3 Development Objects

The following development objects are relevant to this Easy Web Transaction:

<table>
<thead>
<tr>
<th>Development class</th>
<th>Task</th>
</tr>
</thead>
<tbody>
<tr>
<td>Workflow</td>
<td>VSSF</td>
</tr>
<tr>
<td>Task identifier</td>
<td>WS20000102</td>
</tr>
<tr>
<td>Component</td>
<td>Value</td>
</tr>
<tr>
<td>--------------------</td>
<td>--------------------------------------------</td>
</tr>
<tr>
<td>Web transaction</td>
<td>TS_WS20000102H, TS_TS20000256H</td>
</tr>
<tr>
<td>Module pool</td>
<td>SWU5WS20000102H, SWU5TS20000256H</td>
</tr>
<tr>
<td>BAPIs</td>
<td>BAPI_CUSTOMER_CREATEFROMDATA</td>
</tr>
</tbody>
</table>
Process Flow (BC-BMT-WFM Creating a Customer Master Record)

This Easy Web Transaction demonstrates how the workflow functionality is integrated in the Web. This is illustrated by creating a customer master record.

Process Flow

The following process is executed when you use the *Create a customer master record (BC-BMT-WFM)* Easy Web Transaction:

The **New Customer** form is displayed when you log on to the ITS (Internet Transaction Server) on the corresponding R/3 System (service name TS_WS20000102H).

![New Customer form](image)

Complete all of the required fields in the form and choose **Submit data**. Choose **Reset** to delete the entries.

The next screen indicates the work item ID under which the task was started. Acknowledge that the task has started.

The employee responsible for checking the customer master record receives a work item in their Web workplace. With this Easy Web Transaction, this person is you (`WF_INITIATOR`).

Execute the work item.

The form you completed previously is then displayed for you to accept or reject. You can also cancel this procedure if necessary. If you do so, you can execute the work item again later.

The next screen informs you that the work item has been executed. Acknowledge the screen.

**Result**

If you accept the data, the workflow definition calls a BAPI in the background to create the master record. The master record is created if the data is consistent. Otherwise the BAPI will generate an error message and the master record will not be created.

In each case, a message appears in your Web inbox informing you of the result of the action.