

# HP OpenView Select Identity

For the Red Hat Enterprise Linux and  
Windows 2003 Operating Systems

Software Version: 4.0

---

## Workflow Studio Guide

March 2006



## Legal Notices

### Warranty

The only warranties for HP products and services are set forth in the express warranty statements accompanying such products and services. Nothing herein should be construed as constituting an additional warranty. HP shall not be liable for technical or editorial errors or omissions contained herein.

The information contained herein is subject to change without notice.

### Restricted Rights Legend

Confidential computer software. Valid license from HP required for possession, use or copying. Consistent with FAR 12.211 and 12.212, Commercial Computer Software, Computer Software Documentation, and Technical Data for Commercial Items are licensed to the U.S. Government under vendor's standard commercial license.

### Copyright Notices

© Copyright 2006 Hewlett-Packard Development Company, L.P.

### Trademark Notices

AMD and the AMD logo are trademarks of Advanced Micro Devices, Inc.

Intel® and Pentium® are registered trademarks of Intel Corporation in the United States and other countries.

JAVA™ is a US trademark of Sun Microsystems, Inc.

Oracle® is a registered US trademark of Oracle Corporation, Redwood City, California

UNIX® is a registered trademark of The Open Group.

## Documentation Updates

This manual's title page contains the following identifying information:

- Software version number, which indicates the software version
- Document release date, which changes each time the document is updated
- Software release date, which indicates the release date of this version of the software

To check for recent updates, or to verify that you are using the most recent edition of a document, go to:

**[http://ovweb.external.hp.com/lpe/doc\\_serv/](http://ovweb.external.hp.com/lpe/doc_serv/)**

You will also receive updated or new editions if you subscribe to the appropriate product support service. Contact your HP sales representative for details.

The following table indicates changes made to this document since the last released edition.

## Support

Please visit the HP OpenView support web site at:

**<http://www.hp.com/managementsoftware/support>**

This web site provides contact information and details about the products, services, and support that HP OpenView offers.

HP OpenView online software support provides customer self-solve capabilities. It provides a fast and efficient way to access interactive technical support tools needed to manage your business. As a valuable support customer, you can benefit by using the support site to:

- Search for knowledge documents of interest
- Submit enhancement requests online
- Download software patches
- Submit and track progress on support cases
- Manage a support contract
- Look up HP support contacts
- Review information about available services
- Enter discussions with other software customers
- Research and register for software training

Most of the support areas require that you register as an HP Passport user and log in. Many also require a support contract.

To find more information about access levels, go to:

**[http://www.hp.com/managementsoftware/access\\_level](http://www.hp.com/managementsoftware/access_level)**

To register for an HP Passport ID, go to:

**<http://www.managementsoftware.hp.com/passport-registration.html>**

# Contents

1	Introduction to Workflow Studio	1
	Workflow Templates in Select Identity	2
	Concepts and Terms	3
	Behind the Scenes	4
2	Getting Started	7
	Overview of the Workflow Studio Interface	7
	Overview of Creating a Workflow Template	11
	Default Workflow Templates	12
	SI Provisioning Only	12
	SI Provisioning Only Bulk	14
	SI Provisioning Only With ExclusionRule	16
	SI Password Change Provisioning	17
	SIBulkOneStageApproval	18
	SI CertificateRequestProcess	20
	SI Email Only	24
	SI PasswordExpire Email	25
	UserAccountExpirationWF	25
	SI EmailVerifyAndApproval	26
	SI OneStageApproval	29
	SI TwoStageApproval	30
	SI ThreeStageApproval	32
	ReconciliationDefaultProcess	33
	ReconciliationDefaultProcessMove	35
	SI User Enable Disable Workflow	36
	SI Recon User Enable Disable Workflow	38
	Managing Workflow Templates	39
	Modifying Workflow Templates	39

Copying Workflow Templates . . . . .	39
Deleting Workflow Templates . . . . .	40
Creating a Provisioning Template Example . . . . .	40
<b>3 Using Files in the Workflow Studio . . . . .</b>	<b>49</b>
Using a Properties File to Reactivate a Pending Workflow . . . . .	49
Using the Application Definition File . . . . .	51
Defining a Java Application in the Application Definition File . . . . .	51
Defining a Web Service Application in the Application Definition File . . . . .	54
Web Service Metadata . . . . .	54
Importing the Application Definition File . . . . .	57
<b>4 Creating a Workflow Template . . . . .</b>	<b>59</b>
Overview of Template Actions . . . . .	61
Properties and Variables . . . . .	64
Using Properties . . . . .	65
System-Defined Properties . . . . .	66
Timeout Values . . . . .	72
Join Count Examples . . . . .	73
Using Variables . . . . .	74
Create or Changing a Variable . . . . .	74
Workflow Engine-Defined Variables . . . . .	76
Block Variable . . . . .	76
Activities and Blocks . . . . .	78
Setting Properties on the Activity Page . . . . .	81
Defining Actions . . . . .	82
Check Email Verification . . . . .	85
Save Email Notification . . . . .	86
Get Approvers By Role . . . . .	86
Notify Approvers . . . . .	87
Notify Approvers (Default Email Template) . . . . .	87
Create Workflow Approver Task . . . . .	88
Provisioning Task . . . . .	89
Email Notification . . . . .	90
Send an Email Notification . . . . .	91
Post Provision . . . . .	92

Stated Post Provision . . . . .	93
Post Provision in Reconciliation to Save Data . . . . .	94
Post Provision in Bulk Processing to Save Data . . . . .	94
External Call . . . . .	95
Approvers External Call . . . . .	95
Add a Set of Users to a List of Services . . . . .	97
Set Variable . . . . .	97
Log Message . . . . .	98
Throw Exception . . . . .	98
Run Script . . . . .	98
Call Subworkflow . . . . .	99
Add Item to List . . . . .	100
Add Item to Map . . . . .	100
Send Email . . . . .	101
Recover From Last Error . . . . .	102
XPath . . . . .	102
Set Block Variable . . . . .	103
Get Block Variable . . . . .	103
Transitions . . . . .	103
<b>5 Reporting . . . . .</b>	<b>105</b>
Viewing Workflow Status . . . . .	105
Report Templates . . . . .	110
Template Structure . . . . .	110
Instance-level Reporting . . . . .	111
Block-level Reporting . . . . .	114
<Text> Element . . . . .	115
<Table> . . . . .	118
Creating a Custom Report Template . . . . .	122
<b>6 Scenarios . . . . .</b>	<b>127</b>
Enforcing Entitlement Rules . . . . .	127
Adding Services to a User . . . . .	130
Parallel Approval Workflow Design . . . . .	132
Workflow Design . . . . .	132
Testing the Workflow . . . . .	135

A	Frequently Asked Questions (FAQ)	139
	General	139
	Activities	140
	Blocks	141
	Actions	143
	Reports	145
B	Verisign Certificate Management	147
	Glossary	151
	Acronyms	151
	Terms	158
	Index	181



---

# 1 Introduction to Workflow Studio

As described in the *HP OpenView Select Identity Administrator Guide*, a **workflow process** represents the process by which service-access requests are approved and provisioned by HP OpenView Select Identity (Select Identity). Provisioning includes adding, modifying, and removing user accounts.

The complexity of the workflow process can vary widely depending on your provisioning needs. You can simply provision a user by creating the user in Select Identity and then pushing the user account to the external resource. Or, provisioning can require approval from multiple administrators. The approval process can also rely on external calls to third-party systems or databases.

For example, when an employee is promoted to manager, the employee needs access to the company's HCM system to manage other employees. To support these newly-acquired responsibilities, the employee must be granted new entitlements and access privileges. Before giving the employee access to these systems, upper-level management needs to approve the access requests and the employee must be created in the supporting systems. Thus, the workflow process involves retrieving the names of managers, requesting their approval to add the employee to the HCM systems, provisioning the employee's account, and notifying the employee that authorization to manage others has been approved.

**Workflow Studio** enables you to create a workflow template that represents a provisioning process. A **workflow template** models this process in order to automate the actions that approvers and systems management software must perform. This guide describes how to use Workflow Studio to create workflow templates and the building blocks you will use.

# Workflow Templates in Select Identity

Using the Select Identity client, you can assign workflow templates to request events in a Service Role. A Service Role is a Select Identity abstraction defining how a logical grouping of users access a Select Identity Service. The Select Identity Service is a superset of all the identity management elements of a business service.

For example, you can assign a simple provisioning template to an add request for self-registration. This template might perform user provisioning and request a single approval. Thus, when a new user requests access to the service, the template is invoked and an administrator must approve the request before the user is added to the supporting systems.

The following tables list the request events to which you can assign a workflow template when creating Service Roles:

## Bulk Request Events

- Adding a new user
- Adding a service to a user

## Delegated-registration Request Events

- Adding a new user
- Adding a service to a user
- Deleting a service membership
- Disabling a service membership
- Enabling a service membership
- Modifying a user
- Moving a user
- Viewing a service membership

## Reconciliation Request Events

- Adding a service to a user
- Deleting a service membership

## Self-service Request Events

- Adding a new user
- Adding a service to a user
- Modifying a profile
- Viewing a profile

## Service Change Reconciliation Event

- Modifying a user

As Select Identity invokes a template, it creates a workflow instance and performs activities as defined in the template. (“Workflow” refers to a workflow instance.) If you create a more complicated workflow, activities might include the following:

- Selecting a list of approvers by specifying a role created on the **Admin Roles** home page
- Sending email using one of the email templates created on the **Notifications** home page
- Calling an external system registered with Select Identity on the **External Calls** home page

You can generate reports to track the status of request events and the workflows supporting them. See [Reporting](#) for more information.

# Concepts and Terms

While building workflow templates in Workflow Studio, you will encounter the following terms:

## Activities

An activity represents a step in a process that may be traversed when a workflow template is executed. Activities are the core components of workflow templates. The actions defined in activities do the work. An activity can contain actions, which perform various operations such as, track approvals, start a subworkflow, send email, call external applications, and so on.

## Actions

An action invokes functions provided by the workflow engine or by external applications. For example, you can use actions to log information to a file, set a variable to be used later in the workflow, or call an external process to validate certificates.

## Blocks

A block consists of one or more activities. A block is used to group a set of related activities. A block serves two purposes: to define properties (block-level properties) to be shared by a subset of activities and to provide block-level reporting. To define block properties such as Timeout, Join Count, Escalation, and so on, a block must be used to handle these special properties. For example, you might define a block that submits an approval request, waits for the response, and continues execution when the required number of approvers take action.

## Transitions

A transition provides a link from one activity to another. You can define that one activity always follows another. You can define a condition that must be met before the workflow transitions from an activity to one or more other activities. For example, you can define a transition that only allows the workflow to progress if at least two administrators approve a request. If the request is not approved, the workflow can transition to an activity that sends an email notification to an administrator.

# Behind the Scenes

When you save a workflow template, it is saved in the Select Identity repository as an XML file. Its format is the XML Processing Description Language (XPDL) as defined by the Workflow Management Coalition (WfMC). Every time you click **Save**, a new version of the template is created (a template's XML is never overwritten). This assures that currently running workflow instances do not break because the workflow template changes. From Workflow Studio, you always see the latest version of the template. You cannot view or modify the older versions of the template.

The following diagram illustrates the workflow architecture:

Following is a description of the architecture:

- The Workflow Studio creates templates and stores them as XML files. The `Workflow.xsd` and `Action.xsd` files define the structure and the action data model as defined by WfMC.
- The workflow engine interprets the template using the Workflow Meta Data Model, which defines Select Identity's template structure.

- The Scripting Manager evaluates workflow properties and expressions, and the Workflow State Persistency Manager persists workflow properties and state data.
- The Action Handlers invoke predefined actions. The `Applications.xml` file defines applications that are referenced by actions in the template. The Workflow Common Services provide common services for use by the engine, and the Application Services interface into Select Identity.
- The Workflow API provides an interface to the query instance state and other information.



---

## 2 Getting Started

This chapter is provided to help you become comfortable using Workflow Studio and its controls.

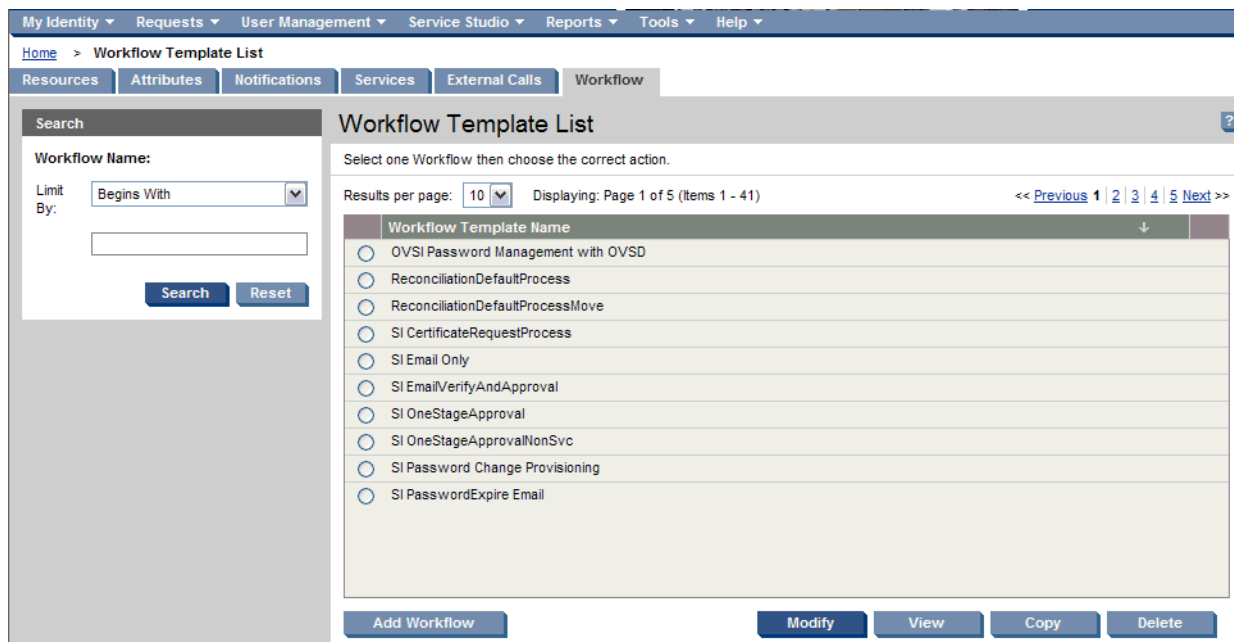
This chapter covers the following:

- Overview of the Workflow Studio Interface
- Overview of Creating a Workflow Template
- Default Workflow Templates
- Managing Workflow Templates
- Creating a Provisioning Template Example

### Overview of the Workflow Studio Interface

To launch Workflow Studio, select the **Service Studio** → **Workflow** menu option. The **Workflow Template List** opens.

**Figure 1 Workflow Template List**



The Workflow Template List contains the following:

- All existing workflow templates, including the default templates provided with Select Identity (see [Default Workflow Templates](#)).
- Number of items per page. You can select the number from the Results per Page list.
- Number of pages and total number of items.
- Search function to quickly find a template in a long list. You can enter letters to search for templates using:

**Begins with**  
**Ends with**  
**Contains**  
**Exact**










From the **Workflow Template List**, you can access the Workflow Studio to create a template or to view, modify, copy, or delete an existing template.








Click **Add Workflow**, **View**, or **Modify**, to open the Workflow Studio in a new window. [Figure 1](#) illustrates the main areas of the Workflow Studio interface.



Use the toolbar buttons to create, select, and move components in the template workspace. When you select a component, its properties are shown on the property tabs.

The following table describes the buttons on the toolbar:

<b>Button</b>	<b>Name and Shortcut</b>	<b>Description</b>
	<b>Save Workflow</b> (no shortcut)	Saves the workflow template.
	<b>Select (V)</b>	Selects an activity or transaction in the template workspace. Click this button then click in the template workspace, to view template properties.
	<b>Multiple Select (no shortcut)</b>	Selects multiple activities or transactions in the template workspace.
	<b>Move (X)</b>	Moves an activity to the clicked spot in the template workspace. (You cannot use this button to drag and drop an activity.)
	<b>Add Start Block (no shortcut)</b>	Adds a start block activity to the template workspace.
	<b>Add End Block (no shortcut)</b>	Adds an end block activity to the clicked location in the template workspace.
	<b>Block Activity (no shortcut)</b>	Adds a single activity that represents an entire block.
	<b>Add Provision (no shortcut)</b>	Adds a single activity representing an entire provisioning block. By default, the new activity's Block Type property is set to provisioning. (Block type is used to identify the rendering format in the report template.)
	<b>Add Post Provision (no shortcut)</b>	Adds a single activity representing an entire post-provisioning block. By default, the new activity's Block Type property is set to postprovisioning.

Button	Name and Shortcut	Description
	<b>Add Approval</b> (no shortcut)	Adds a single activity representing an entire approval block. By default, the new activity's Block Type property is set to <code>approval</code> and its Join Count property is set to 1.
	Add Activity (A)	Adds an activity to the clicked spot in the template workspace.
	Add Transition (C)	Adds a transition between the selected activities.
	Delete (D)	Deletes the selected activity or transition.
	Grid On/Off (no shortcut)	Toggles grid display in the template workspace.
	Refresh Image (no shortcut)	Refreshes the data shown in the Workflow Studio window.
	Help (no shortcut)	Opens the online help system.

The following graphics represent the building blocks of workflow templates:



An activity



A transition

# Overview of Creating a Workflow Template

This section provides an overview of the steps necessary to create a workflow template. Details about each step are provided in [Creating a Workflow Template](#) on [page 59](#): Also see [Creating a Provisioning Template Example](#) on [page 40](#) for detailed step-by-step instructions.

- 1 Select the **Service Studio** → **Workflow** → **Add New Template**. The **Workflow Studio**, an interface dedicated to creating workflow templates, opens.
- 2 Specify a name for the template and define its global attributes on the **Workflow Information** tab (on the right). See [Properties and Variables](#) on [page 64](#) for details.

- 3 Create **activities**.

Each activity represents a step in a workflow, such as obtaining approver email addresses or checking the status of a stage. For each activity, you can define properties and actions.

You must create the following activities in each workflow:

`begin` — marks the starting point of the workflow.

`end` — marks the ending point of the workflow.

`errorEnd` — terminates the workflow if an error occurs. When the workflow ends with the `errorEnd` activity, the “End with error” status is shown on the **Request Status** page.

You can also create an activity to handle exceptions.

- 4 Group activities by creating **blocks**.

A block consists of a group of related activities, or a single block activity that enables you to define information to be shared by the contained activities and to provide block-level reporting. For example, you may want to create a retry block for a provisioning activity. If provisioning fails, the retry block can wait for a specified amount of time then retry provisioning. Since `timeout` is a block property, the block activity must be used for the retry operation.

A block can consist of start and end blocks with many activities in between, or a block can be a single block activity that represents a block. Block activities can be added to a workflow template through the **Add Block Activity**, **Add Approval**, **Add Provision**, or **Add Post Provision Block** buttons on the toolbar. See [Activities and Blocks](#) on page 78 for details.

- 5 Create **transitions** between activities.

A transition instructs Select Identity how to proceed from one activity to the next. You can set a condition for the transition, so that the transition is not executed until the condition is met. To understand how to create transitions, see [Transitions](#) on page 103.

- 6 Repeat [Step 3](#) through [Step 5](#) as necessary. The creation process for workflow templates is iterative and may require that you rework the logic and flow of the template.
- 7 Save the template.

To view the status of workflows during execution, see [Reporting](#) on page 105 for details.

## Default Workflow Templates

Select Identity provides default workflow templates, each of which is an example of a common workflow process. You can assign the default templates to Service Roles as you become familiar with the system. Edit these templates or use them in their original form.

### SI Provisioning Only

This template provisions a user.

- 1 First, this template provisions the user in the external resource from the **Provision** block activity.

If this is unsuccessful, the workflow retries this provisioning step three times. If the provisioning fails more than three times, the template sends email notification of the failure and the workflow terminates.

- 2 If provisioning is successful, the workflow executes the **Post Provision** action to update the user in Select Identity.

If the user update fails, the template sends email notification of the failure and the workflow terminates.

- 3 If the user is successfully provisioned in Select Identity, the workflow transitions to the **Notify Target** block, which sends email notification using the **PostAddNotification** template. It also determines if the user's password was changed or reset and sets several variables.
- 4 If the user's password has changed, the workflow sends email notification using the **NewAccountPassword** template. If the password was reset, the **Reset Password** template is used for notification, and this notification passes the [USERDEF:ResetStatus] variable.

If an error occurs at any time during the workflow, the template catches the exception, sends an email notification, and exits. (If an exception occurs when sending the email, it logs that error.)

The **SI Provisioning Only** template contains the following blocks:

<b>Block</b>	<b>Activity Tab Fields</b>
<b>begin</b>	Wait Activity: No
<b>Provision</b>	Wait Activity: Yes Join Type: XOR Split Type:XOR
<b>RetryEB</b>	Wait Activity: No
<b>RetrySB</b>	Wait Activity: Yes
<b>Post Provision</b>	Wait Activity: No Split Type: XOR
<b>Notify Error</b>	Wait Activity: No Join Type: XOR
<b>catchException</b>	Wait Activity: No Split Type: XOR
<b>LogException</b>	Wait Activity: No Join Type:

<b>Block</b>	<b>Activity Tab Fields</b>
<b>errorEnd</b>	Wait Activity: No Join Type: XOR
<b>Notify Target</b>	Wait Activity: No Split Type: XOR
<b>Send Password</b>	Wait Activity: No
<b>Send Reset Password</b>	Wait Activity: No
<b>end</b>	Wait Activity: No Join Type: XOR

## SI Provisioning Only Bulk

This template is similar to the **SI Provisioning Only** template, but it is only for use with bulk events. It does not retry activities in the event of failure.

- 1 First, the workflow provisions the user in an external resource.
- 2 If provisioning is successful, the workflow transitions to the **PostProvision** block, which provisions the user in Select Identity using the action “Post Provision in Bulk Processing to save data.”
- 3 If the user is successfully provisioned, the workflow transitions to the **Notify Target** block, which sends email using the **PostAddNotificationBulk** template. It also determines if the user’s password has changed and sets several variables.
- 4 If the user’s password has changed, the workflow sends email using the **New Account Password Bulk** template.
- 5 If an error occurs at any time during the workflow, the template catches the exception, sends an email notification, and exits. (If an exception occurs when sending the email, it logs that error.)

The **SI Provisioning Only Bulk** template contains the following blocks:

<b>Block</b>	<b>Activity Tab Fields</b>
<b>begin</b>	Wait Activity: No
<b>Provision</b>	Wait Activity: Yes Split Type: XOR
<b>Post Provision</b>	Wait Activity: No Split Type: XOR
<b>Notify Error</b>	Wait Activity: No Join Type: XOR
<b>catchException</b>	Wait Activity: No Split Type: XOR
<b>LogException</b>	Wait Activity: No Join Type
<b>errorEnd</b>	Wait Activity: No Join Type: XOR
<b>Notify Target</b>	Wait Activity: No Split Type: XOR
<b>Send Password</b>	Wait Activity: No
<b>end</b>	Wait Activity: No Join Type: XOR

## SI Provisioning Only With ExclusionRule

This template is similar to **SI Provisioning Only** with one entry added before the first (provisioning) step; it provisions the user after checking for possible exclusions.

This template also evaluates the exclusion rule for entitlement or service conflicts. If conflicts exist, a rejection email is sent and the workflow terminates.

The **SI Provisioning Only With ExclusionRule** template contains the following blocks:

<b>Block</b>	<b>Activity Tab Fields</b>
<b>begin</b>	Wait Activity: No
<b>ExclusionRuleCall</b>	Wait Activity: No Split Type: XOR
<b>Notify Rejected</b>	Wait Activity: No
<b>errorEnd</b>	Wait Activity: No Join Type
<b>catchException</b>	Wait Activity: No Split Type: XOR
<b>LogException</b>	Wait Activity: No Join Type
<b>Provision</b>	Wait Activity: Yes Join Type: XOR Split Type: XOR
<b>RetryEB</b>	Wait Activity: No
<b>RetrySB</b>	Wait Activity: Yes
<b>Post Provision</b>	Wait Activity: No Split Type: XOR



<b>Block</b>	<b>Activity Tab Fields</b>
<b>Notify Error</b>	Wait Activity: No Join Type: XOR
<b>Notify Target</b>	Wait Activity: No Split Type: OXR
<b>Send Password</b>	Wait Activity: No
<b>Send Reset Password</b>	Wait Activity: No
<b>end</b>	Wait Activity: No Join Type: OXR

## SI Password Change Provisioning

This template is similar to **SI Provisioning Only** except there is no retry block for provisioning. This template is used for password resets.

The **SI Password Change Provisioning** template contains the following blocks:

<b>Block</b>	<b>Activity Tab Fields</b>
<b>begin</b>	Wait Activity: No
<b>Provision</b>	Wait Activity: Yes Split Type: XOR
<b>Post Provision</b>	Wait Activity: No Split Type: XOR
<b>Notify Error</b>	Wait Activity: No Join Type: XOR
<b>catchException</b>	Wait Activity: No Split Type: XOR
<b>LogException</b>	Wait Activity: No Join Type

<b>Block</b>	<b>Activity Tab Fields</b>
<b>errorEnd</b>	Wait Activity: No Join Type
<b>Notify Target</b>	Wait Activity: No Split Type: OXR
<b>Send Password</b>	Wait Activity: No
<b>Send Reset Password</b>	Wait Activity: No
<b>end</b>	Wait Activity: No Join Type: OXR

## SIBulkOneStageApproval

This template is similar to the **SI Provisioning Only Bulk** template with the addition of an approval step prior to provisioning.

- 1 First, it requests approval for provisioning. If approved, the workflow provisions the user in an external resource.
- 2 If provisioning is successful, the workflow transitions to the PostProvision block, which provisions the user in Select Identity using the action “Post Provision in Bulk Processing to save data.”
- 3 If the user is successfully provisioned in Select Identity, the workflow transitions to the Notify Target block, which sends email using the **PostAddNotification Bulk** template. It also determines if the user’s password has changed and sets several variables.
- 4 If the user’s password has changed, the workflow sends email using the **NewAccountPassword Bulk** template.
- 5 If an error occurs at any time during the workflow, the template catches the exception, sends an email, and exits. (If an exception occurs when sending the email, it logs that error.)

The **SIBulkOneStageApproval** template contains the following blocks:

<b>Block</b>	<b>Activity Tab Fields</b>
<b>begin</b>	Wait Activity: No
<b>Approval1</b>	Wait Activity: Yes Split Type:
<b>Notify Rejected</b>	Wait Activity: No
<b>errorEnd</b>	Wait Activity: No Join Type
<b>catchException</b>	Wait Activity: No Split Type: XOR
<b>LogException</b>	Wait Activity: No Join Type
<b>Provision</b>	Wait Activity: Yes Join Type: XOR Split Type: XOR
<b>RetryEB</b>	Wait Activity: No
<b>RetrySB</b>	Wait Activity: Yes
<b>Post Provision</b>	Wait Activity: No Split Type: XOR
<b>Notify Error</b>	Wait Activity: No Join Type: XOR
<b>Notify Target</b>	Wait Activity: No Split Type: OXR
<b>Send Password</b>	Wait Activity: No
<b>end</b>	Wait Activity: No Join Type: OXR

## SI CertificateRequestProcess

This template validates a digital certificate request and generates a certificate during the user-provisioning process. This workflow consists of the following parts:

- Approval: Provisioning requests are either approved or rejected.
- Certificate Request: Validates and generates the digital certificate.
- Provisioning: The user is provisioned in Select Identity if the above Certificate Request procedure is successful.



Before using this template, which is configured to manage Verisign certificates, perform the steps in [Verisign Certificate Management](#) . To use this template to manage certificates provided by another Certificate Authority (CA), you must change the values passed to the parameters (see below) and implement the Java class that is specific to that CA. See the *HP OpenView Select Identity External Call Developer Guide* for more information.

Following is the **SI CertificateRequestProcess** process flow:

- 1 Once the user is approved for provisioning, the certificate request function is called in the CertRequest activity block of the workflow, as shown in the default **SI CertificateRequestProcess** template.
- 2 The CertRequest activity block does the following:

- a Calls the WorkflowCertificateRequest external call and passes the following parameters, which are required by the external call. The WorkflowCertificateRequest external call is pre-registered in Select Identity for validating certificate requests. The following table lists the parameters of the WorkflowCertificateRequest external call:

<b>Field Name</b>	<b>Default Value</b>	<b>Description</b>
<b>DN_FieldName</b>	DN	Attribute name that stores the user's distinguished name (DN) from the certificate.
<b>CertificateFieldName</b>	CertRegPswd	Challenge password assigned at the time of user registration.
<b>EmailTemplateName</b>	Verisign Certificate Issue	Default email template from Select Identity, to send email to the user.
<b>CertificateProviderName</b>	Verisign	Certificate provider name. In the case of Verisign, the name must be "Verisign." In all other cases, the administrator can assign the name.
<b>ExternalCallName</b>	VerisignCertImpl	Name of the CA-specific Java class that implements validation and generation functions for the certificate.

- b A Certificate Service defines the attributes for DN\_FieldName and CertificateFieldName and associates them with a Service view. For example, from the table above, the Service would define the DN and CertRegPswd attributes and associate them with a view.
- 3 The validation request is delegated to a provider-specific external call based on the WorkflowCertificateRequest parameter ExternalCallName — in the case above, VerisignCertImpl.

- 4 Once the request is validated, the provider-specific external call returns one of the following responses, based on the `CertificateManagementInterface`:
  - **Reject the certificate request:**  
The user is not provisioned and is notified via email.
  - **Generate a new certificate:**
    - An email is sent to the user with a link to `Select Identity` to initiate the certificate request. The email template parameter, `EmailTemplateName`, in the `WorkflowCertificateRequest` external call identifies the appropriate email template to send to the user.
    - After this, the workflow transitions to a wait state in the `CertWT` activity. Once the user clicks on the link, `Select Identity` responds with a challenge password query.
    - If the password validation is successful, an external call is made to the provider-specific external call to generate a new certificate.
    - The user's DN is updated in the provisioning request and the workflow transitions to the `Provisioning` block to provision the user in `Select Identity`.
    - If the validation fails, the user is not provisioned and is notified through email.
  - **User already has a certificate:**  
The existing certificate is returned to the `WorkflowCertificateRequest` external call, which in turn updates the status in the workflow and the DN attribute for the request. The DN attribute name is stored in the external call parameter `DN`.

The **SI CertificateRequestProcess** template contains the following blocks:

Block	Activity Tab Fields
<b>begin</b>	Wait Activity: No
<b>Approval1SB</b>	Wait Activity: No
<b>Approval1WT</b>	Wait Activity: Yes
<b>Approval1EB</b>	Wait Activity: No Split Type: XOR

<b>Block</b>	<b>Activity Tab Fields</b>
<b>Approval2SB</b>	Wait Activity: No
<b>Approval2WT</b>	Wait Activity: Yes
<b>Approval2EB</b>	Wait Activity: No Split Type: XOR
<b>Notify Rejected</b>	Wait Activity: No Join Type: XOR
<b>catchException</b>	Wait Activity: No Split Type: XOR
<b>LogException</b>	Wait Activity: No
<b>errorEnd</b>	Wait Activity: No Join Type: XOR
<b>CertRequest</b>	Wait Activity: No
<b>CheckStatus</b>	Wait Activity: No Split Type: XOR
<b>Notify Error</b>	Wait Activity: No Join Type: XOR
<b>CertWT</b>	Wait Activity: Yes Join Type: XOR
<b>ProvisionSB</b>	Wait Activity: No Join Type: XOR
<b>ProvisionWT</b>	Wait Activity: Yes
<b>ProvisionEB</b>	Wait Activity: No Join Type: XOR
<b>RetrySB</b>	Wait Activity: No
<b>RetryWT</b>	Wait Activity: Yes
<b>RetryEB</b>	Wait Activity: No

<b>Block</b>	<b>Activity Tab Fields</b>
<b>Post ProvisionSB</b>	Wait Activity: No
<b>Post ProvisionEB</b>	Wait Activity: No Split Type: XOR
<b>Notify Target</b>	Wait Activity: No Split Type: OXR
<b>end</b>	Wait Activity: No Join Type: OXR

## SI Email Only

This template invokes the email notification action to send email using the **User Account Locked** template. If an error occurs during the workflow, the template catches the exception, sends an email notification, and exits. (If an exception occurs when sending the email, it logs that error.)

The **SI Email Only** template contains the following blocks:

<b>Block</b>	<b>Activity Tab Fields</b>
<b>begin</b>	Wait Activity: No
<b>Send Password</b>	Wait Activity: No
<b>end</b>	Wait Activity: No
<b>catchException</b>	Wait Activity: No Split Type: XOR
<b>LogException</b>	Wait Activity: No
<b>errorEnd</b>	Wait Activity: No Join Type: XOR



## SI PasswordExpire Email

This template sends email using the **Password Expire Notification** email template. If an error occurs during the workflow, the template catches the exception, sends an email, and exits. (If an exception occurs when sending the email, it logs that error.)

The **SI PasswordExpire Email** template contains the following blocks:

Block	Activity Tab Fields
<b>begin</b>	Wait Activity: No
<b>Send Password</b>	Wait Activity: No
<b>end</b>	Wait Activity: No
<b>catchException</b>	Wait Activity: No Split Type:
<b>LogException</b>	Wait Activity: No
<b>errorEnd</b>	Wait Activity: No Join Type:

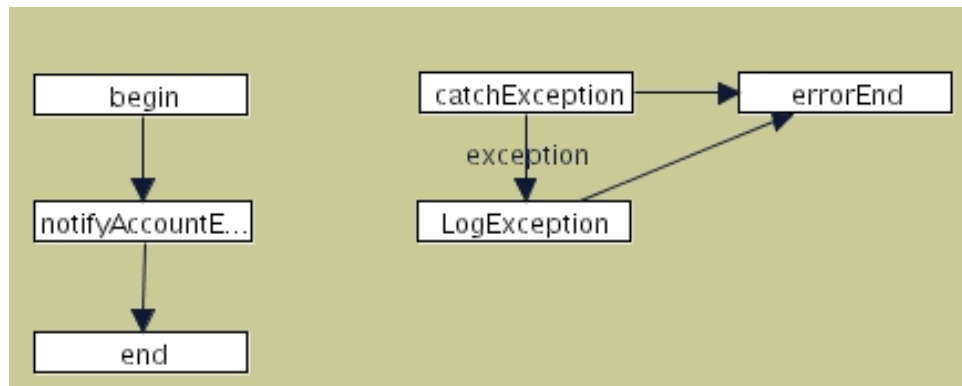
## UserAccountExpirationWF

This template sends email by invoking the Send an Email Notification action. It uses the **User Account Expiration Notice Email** template and passes the USERDEF variable to the template. This variable is also set, defining the email address of the user's manager. If an error occurs during the workflow, the template catches the exception, sends an email, and exits. (If an exception occurs when sending the email, it logs that error.)

The **UserAccountExpirationWF** template contains the following blocks:

Block	Activity Tab Fields
<b>begin</b>	Wait Activity: No
<b>notifyAccountExpiration</b>	Wait Activity: No
<b>end</b>	Wait Activity: No
<b>catchException</b>	Wait Activity: No Split Type: XOR
<b>LogException</b>	Wait Activity: No
<b>errorEnd</b>	Wait Activity: No Join Type: XOR

**Figure 2 UserAccountExpirationWF Template**



## SI EmailVerifyAndApproval

This template verifies the user's email address in Select Identity, waits for approval, then provisions the user.

The SI EmailVerifyAndApproval template contains the following blocks:

<b>Block</b>	<b>Activity Tab Fields</b>
<b>begin</b>	Wait Activity: No
<b>Check Email</b>	Wait Activity: No Split Type: XOR
<b>Verify Email</b>	Wait Activity: Yes
<b>Approval1</b>	Wait Activity: Yes Join Type: XOR Split Type; XOR
<b>Notify Rejected</b>	Wait Activity: No
<b>catchException</b>	Wait Activity: No Split Type: XOR
<b>LogException</b>	Wait Activity: No
<b>errorEnd</b>	Wait Activity: No Join Type: XOR
<b>Provision</b>	Wait Activity: Yes Join Type: XOR Split Type; XOR
<b>RetryEB</b>	Wait Activity: No
<b>RetrySB</b>	Wait Activity: Yes
<b>Post Provision</b>	Wait Activity: No Split Type: XOR
<b>Notify Target</b>	Wait Activity: No Split Type: XOR

<b>Block</b>	<b>Activity Tab Fields</b>
<b>Send Password</b>	Wait Activity: No
<b>Send Reset Password</b>	Wait Activity: No
<b>end</b>	Wait Activity: No Join Type: XOR

The **SI EmailVerifyAndApproval** workflow is executed as follows:

- 1 The Check email Verification action is invoked in the Check Email activity, which sets a property to false if the email address is already validated in Select Identity.
- 2 If the email address is not validated, the workflow transitions to the VerifyEmail block. This block invokes the Save Email Notification action, which saves an email notification to a batch table. When the batch process runs, the email is sent. The workflow continues when the address is validated by the user clicking on the link in the email.
- 3 The workflow transitions to the Approval block. The Get approvers by role action is invoked to retrieve the list of approvers for the user. The Notify Approvers action is invoked to send email to the list of approvers retrieved in the previous action. The Create workflow approver task action is invoked to create an approval task on the Request Worklist home page for all users in the approver list. This activity is a wait activity, thus the workflow will not transition until the approver accepts or rejects the approval request.  
  
If the request is approved, the workflow transitions to the Provision block. The Provisioning Task action is invoked to provision the user in the external system(s) as defined by the Service.
- 5 If the provision fails, the workflow transitions to the Retry block. The workflow will retry provisioning three times. If it fails after three retries, the workflow transitions to the Notify Error activity, which emails the failure. The workflow then terminates.

- 6 If the provision succeeds in the external system, the PostProvision block begins. This block provisions the user in Select Identity. If post-provisioning succeeds in Select Identity, the Notify Target activity sends email to the user and transitions to either the Send Password or Send Reset Password activity, which sends the password to the request target if the password changed.
- 7 If the post-provisioning fails, the workflow transitions to the Notify Error activity, which emails notification of the failure. The workflow then exists.  
  
If an error occurs at any time during the workflow, the template catches the exception, sends an email notification, and exits. (If an exception occurs when sending the email, it logs that error.)

## SI OneStageApproval

This template requests approval then provisions the user. Approvers are derived from the Concerto Sys Admin role by default. It is identical to **SI EmailVerifyAndApproval** except for the following:

- It does not verify the user's email address.
- If the password was reset, the template sends Reset Password.

The SI OneStageApproval template contains the following blocks:

Block	Activity Tab Fields
<b>begin</b>	Wait Activity: No
<b>Approval1</b>	Wait Activity: Yes Join Type: XOR
<b>Notify Rejected</b>	Wait Activity: No
<b>catchException</b>	Wait Activity: No Split Type: XOR
<b>LogException</b>	Wait Activity: No Join Type: XOR
<b>errorEnd</b>	Wait Activity: No Join Type: XOR

<b>Block</b>	<b>Activity Tab Fields</b>
<b>Provision</b>	Wait Activity: Yes Join Type: XOR Split Type; XOR
<b>RetryEB</b>	Wait Activity: No
<b>RetrySB</b>	Wait Activity: Yes
<b>Notify Error</b>	Wait Activity: No Join Type: XOR
<b>Post Provision</b>	Wait Activity: No Split Type: XOR
<b>Notify Target</b>	Wait Activity: No Split Type: XOR
<b>Send Password</b>	Wait Activity: No
<b>Send Reset Password</b>	Wait Activity: No
<b>end</b>	Wait Activity: No Join Type: XOR

## SI TwoStageApproval

This template adds another approval stage to the **SI OneStageApproval** template. Second-stage approvers are derived from the Workflow Approvers role by default.

The **SI TwoStageApproval** template contains the following blocks:

<b>Block</b>	<b>Activity Tab Fields</b>
<b>begin</b>	Wait Activity: No
<b>Approval1</b>	Wait Activity: Yes Split Type: XOR
<b>Approval2</b>	Wait Activity: Yes Split Type: XOR
<b>Notify Rejected</b>	Wait Activity: No Join Type: XOR
<b>catchException</b>	Wait Activity: No Split Type: XOR
<b>LogException</b>	Wait Activity: No Join Type: XOR
<b>errorEnd</b>	Wait Activity: No Join Type: XOR
<b>Provision</b>	Wait Activity: Yes Join Type: XOR Split Type; XOR
<b>RetryEB</b>	Wait Activity: No
<b>RetrySB</b>	Wait Activity: Yes
<b>Notify Error</b>	Wait Activity: No Join Type: XOR
<b>Post Provision</b>	Wait Activity: No Split Type: XOR
<b>Notify Target</b>	Wait Activity: No Split Type: XOR

<b>Block</b>	<b>Activity Tab Fields</b>
<b>Send Password</b>	Wait Activity: No
<b>Send Reset Password</b>	Wait Activity: No
<b>end</b>	Wait Activity: No Join Type: XOR

## SI ThreeStageApproval

This template adds another approval stage to the **SI TwoStageApproval** template. Third stage approval derives the approvers from the Concerro Sys Admin role by default.

The **SI ThreeStageApproval** template contains the following blocks:

<b>Block</b>	<b>Activity Tab Fields</b>
<b>begin</b>	Wait Activity: No
<b>Approval1</b>	Wait Activity: Yes Split Type: XOR
<b>Approval2</b>	Wait Activity: Yes Split Type: XOR
<b>Approval3</b>	Wait Activity: Yes Split Type: XOR
<b>Notify Rejected</b>	Wait Activity: No Join Type: XOR
<b>LogException</b>	Wait Activity: No Join Type: XOR
<b>catchException</b>	Wait Activity: No Split Type: XOR
<b>errorEnd</b>	Wait Activity: No Join Type: XOR



<b>Block</b>	<b>Activity Tab Fields</b>
<b>Provision</b>	Wait Activity: Yes Join Type: XOR Split Type; XOR
<b>Retry</b>	Wait Activity: Yes
<b>Notify Error</b>	Wait Activity: No Join Type: XOR
<b>Post Provision</b>	Wait Activity: No Split Type: XOR
<b>Notify Target</b>	Wait Activity: No Split Type: XOR
<b>Send Password</b>	Wait Activity: No
<b>Send Reset Password</b>	Wait Activity: No
<b>end</b>	Wait Activity: No Join Type: XOR

## ReconciliationDefaultProcess

This template reconciles account data with a resource. It is very similar to the **SI Provisioning Only** template; it provisions users in an external resource. Then, if the user is successfully provisioned, the template executes the Post Provision in Reconciliation action, updating the Select Identity database with data sent in the reconciliation request.

The **ReconciliationDefaultProcess** template contains the following blocks:

<b>Block</b>	<b>Activity Tab Fields</b>
<b>begin</b>	Wait Activity: No
<b>Provision</b>	Wait Activity: Yes Join Type: XOR Split Type: XOR
<b>RetryEB</b>	Wait Activity: No
<b>RetrySB</b>	Wait Activity: Yes
<b>Prov Error</b>	Wait Activity: No
<b>catchException</b>	Wait Activity: No Split Type: XOR
<b>LogException</b>	Wait Activity: No
<b>errorEnd</b>	Wait Activity: No Join Type: XOR
<b>Post Provision</b>	Wait Activity: No Split Type: XOR
<b>PProv Error</b>	Wait Activity: No
<b>end</b>	Wait Activity: No

- ▶ Failure notifications are sent automatically unless turned off. If processing thousands of requests at once, you may want to either turn off the block or remove it to prevent overloading the email server with thousands of failure notices.

## ReconciliationDefaultProcessMove

This template is similar to the **ReconciliationDefaultProcess** template. This workflow is used for Adding Services to a user based on a context change. After Post Provisioning this workflow includes another activity called Add2Service.

- 1 The Add2Service activity first calls the “LoadUserServices” workflow external call.
- 2 Then Add2Service calls the action “Add a set of users to a list of services.”

The **ReconciliationDefaultProcessMove** template contains the following blocks:

Block	Activity Tab Fields
<b>begin</b>	Wait Activity: No
<b>Provision</b>	Wait Activity: Yes Join Type: XOR Split Type; XOR
<b>RetryEB</b>	Wait Activity: No
<b>RetrySB</b>	Wait Activity: Yes
<b>Prov Error</b>	Wait Activity: No
<b>catchException</b>	Wait Activity: No Split Type: XOR
<b>LogException</b>	Wait Activity: No
<b>errorEnd</b>	Wait Activity: No Join Type: XOR
<b>Post Provision</b>	Wait Activity: No Split Type: XOR
<b>PProv Error</b>	Wait Activity: No

<b>Block</b>	<b>Activity Tab Fields</b>
<b>Add2Svcs</b>	Wait Activity: No Split Type: XOR
<b>Add2Svc Error</b>	Wait Activity: No
<b>end</b>	Wait Activity: No

- ▶ Failure notifications are sent automatically unless turned off. If processing thousands of requests at once, you may want to either turn off the block or remove it to prevent overloading the email server with thousands of failure notices.

## SI User Enable Disable Workflow

This template is similar to the **SI Provisioning Only** template. The only difference is that the post-provisioning block calls an external call, `UserEnableDisableWFExtCall`. This external call enables or disables the provisioned user depending on the value of the specified attribute by initiating a new request via the web service.

To use this new workflow, you must configure the external call in advance. `UserEnableDisableWFExtCall` has six parameters:

<b>Parameter</b>	<b>Parameter Value Description</b>
<b>AttributeName</b>	Attribute name for which the value is checked
<b>EnableValue</b>	If the value of the attribute of the user matches the <code>EnableValue</code> , then enable the user if the user is disabled
<b>DisableValue</b>	If the value of the attribute of the user matches the <code>DisableValue</code> , then disable the user if the user is enabled

<b>Parameter</b>	<b>Parameter Value Description</b>
<b>UserName</b>	Admin with authority to modify users that will use this external call
<b>Password</b>	Admin password
<b>url</b>	Webservices URL

The **SI User Enable Disable Workflow** template contains the following blocks:

<b>Block</b>	<b>Activity Tab Fields</b>
<b>begin</b>	Wait Activity: No
<b>Provision</b>	Wait Activity: Yes Join Type: XOR Split Type; XOR
<b>Retry</b>	Wait Activity: Yes
<b>Post Provision</b>	Wait Activity: No Split Type: XOR
<b>Notify Error</b>	Wait Activity: No Join Type: XOR
<b>catchException</b>	Wait Activity: No Split Type: XOR
<b>LogException</b>	Wait Activity: No Join Type:
<b>errorEnd</b>	Wait Activity: No Join Type: XOR
<b>Notify Target</b>	Wait Activity: No Split Type: XOR

<b>Block</b>	<b>Activity Tab Fields</b>
<b>Send Password</b>	Wait Activity: No
<b>Send Reset Password</b>	Wait Activity: No
<b>end</b>	Wait Activity: No Join Typs: XOR

## SI Recon User Enable Disable Workflow

This template is similar to the **ReconciliationDefaultProcess** template. The only difference is that the post-provisioning block calls an external call, `UserEnableDisableWFExtCall`. This external call is also used in the **SI User Enable Disable Workflow** template. See [SI User Enable Disable Workflow](#) on page 36 for a description of the `UserEnableDisableWFExtCall` external call and its parameters.

The **SI Recon User Enable Disable Workflow** template contains the following blocks:

<b>Block</b>	<b>Activity Tab Fields</b>
<b>begin</b>	Wait Activity: No
<b>Provision</b>	Wait Activity: Yes
<b>Post Provision</b>	Wait Activity: No
<b>end</b>	Wait Activity: No
<b>catchException</b>	Wait Activity: No Split Type: XOR
<b>LogException</b>	Wait Activity: No
<b>errorEnd</b>	Wait Activity: No Join Type: XOR

# Managing Workflow Templates


You can create templates from scratch, or copy an existing template and then modify it. Select Identity provides several default workflow templates. See [Default Workflow Templates](#) for details.

This section covers the following topics:

- [Modifying Workflow Templates](#)
- [Copying Workflow Templates](#)
- [Deleting Workflow Templates](#)

## Modifying Workflow Templates

Perform the following steps to modify an existing workflow template:

- 1 Select the **Service Studio** → **Workflow** menu option.  
The **Workflow Template List** opens.
- 2 Locate and select a template and click **Modify**.  
The **Modify Workflow Template** page opens with the template to be modified.
- 3 Make your changes and save the template by clicking **Save** .  
See [Creating a Provisioning Template Example](#) for step-by-step instructions on creating a workflow template.

## Copying Workflow Templates

Perform the following steps to copy an existing workflow template:

- 1 Select the **Service Studio** → **Workflow** menu option.  
The **Workflow Template Search** page opens.
- 2 Locate and select a template and click **Copy**.  
The **Copy Workflow Template** page opens.
- 3 Enter a unique template name.
- 4 Click **OK**.  
Copies the template with the new name and adds it to the list on the Workflow Template List.

- 5 If you want to modify this template, select it and click **Modify**. Then follow the instructions in [Modifying Workflow Templates](#).

## Deleting Workflow Templates




Perform the following steps to delete an existing workflow template:


- 1 Select the **Service Studio** → **Workflow** menu option.  
The **Workflow Template List** opens.
- 2 Locate and select the template you want to delete and click **Delete**.
- 3 When prompted to confirm the action, click **OK** to delete the template.  
Disables the workflow template in the database, making it inaccessible from the Select Identity interface.


## Creating a Provisioning Template Example


The **SI Provisioning Only** template provides a workflow for provisioning users. It is a good example of a workflow template, and the steps below walk you through the procedure of creating a similar template. This example shows you how to create a workflow using single block activities.



Instead of using the **Add Start Block** () , **Add Activity** () , and **Add End Block** () buttons to construct a block of activities, you can add a single block activity that represents a block. Select one of the following block buttons and put the block in the template workspace:

Block Activity ()

Add Provision ()

Add Post Provision ()


Add Approval ()



Follow this procedure to practice using the Workflow Studio, to become comfortable with its controls and properties.



As you specify properties for the template, you must click **Apply** to save the information. If you do not click **Apply** after entering data, your changes will be lost.

You can save your new or modified workflow template after you name it at any point by clicking **Save** . You can return to complete your workflow later.


Create the provisioning template by completing the following tasks:

- Task 1: Create a new workflow
- Task 2: Create the begin activity
- Task 3: Create the provisioning block
- Task 4: Optionally define the Provision block properties
- Task 5: Define the Actions for the Provision block
- Task 6: Add a transition from the begin activity to the Provision activity
- Task 7: Create the post-provisioning block
- Task 8: Add a transition from the Provision activity to the PostProvision activity
- Task 9: Add an end activity that runs if provisioning is successful
- Task 10: Add a transition from the Post-provision activity to the End activity
- Task 11: Define an activity that logs an error if provisioning is unsuccessful
- c Define an activity that logs an error if provisioning is unsuccessful
- Task 12: Add a transition from the Notify Error activity to the error end activity
- Task 13: Save the template

### Task 1: Create a new workflow

- 1 Select the **Service Studio** → **Workflow** menu option in the Select Identity client.  
The **Workflow Template List** opens.
- 2 Click **Add Workflow**.  
The **Add Workflow Template** page opens.
- 3 Enter `provision_workflow` in the **Workflow Name** field and click **Apply**.  
The general template properties display on the **Workflow Information** tab.

### Task 2: Create the begin activity


- 1 Click **Add Activity** () then click in the template workspace.  
Adds an activity called **Activity\_1** to the template.
- 2 Rename **Activity\_1** to `begin` in the **ID** field on the **Activity** tab.
- 3 Click **Apply** to save this information.






Activity names (IDs) must be unique in a workflow template. When assigning a name to an activity, ensure that it is unique. Also, the first activity of a template must be named **begin**.

### Task 3: Create the provisioning block

Add the Provision single block activity. In this case, you do not use the start and end block activities.

- 1 Click **Add Provision** ()  
This block adds a single provision activity representing the whole provisioning block.
- 2 Click below the begin activity in the template workspace.  
Adds an activity called **Activity\_2**.



If you wish to move an activity after adding it to the template, click **Move** () and click the activity you wish to move. Then click the point on the screen where you want to place the upper left corner of the activity. To help you organize activities and place them more precisely in the template, click **Grid On/Off** ()  
Click **Select** () when the move is complete.

- 3 Enter `Provision` in the **ID** field and select the **Wait Activity** check box.  
This activity must wait for the action to complete before it transitions to the next activity.
- 4 Click **Apply** to save these settings.

#### Task 4: Optionally define the Provision block properties

Define the Provision block properties in the **Properties** page by clicking the **Properties** tab.



Each block is identified by a block ID. By default, a block ID is the same as activity ID if the block is a single activity block. (It appears as `defaultBlockId` in the Properties screen Block Id field and the Workflow engine internally translates block ID into the activity ID.)

However, you can change the block ID in the Block ID property field to any name as long as it is a unique block ID within the template. The block Id is referenced by the report template to render the block information. It is also used to identify a block to access the block data.



#### Task 5: Define the Actions for the Provision block

- 1 Click the **Actions** tab.
- 2 Define a map variable for later use in the case of an error or exception occurring.  
The value of this variable is used in the Notify Error activity, which sends the Provision Failed email. This email states where in the workflow the error or exception occurred.
  - a Click **Add**.
  - b Select **Add Item to Map** from the **Name** list.
  - c Complete the fields as follows:

<b>New Collection:</b>	Check the box
<b>Map Variable Name:</b>	<code>\$ProvisionMap</code>
<b>Element Name Variable:</b>	<code>"errorEvent"</code>
<b>Element Value Variable:</b>	<code>"Provisioning"</code>
  - d Click **Apply** to save these settings.
  - e Click **Return** to add another action.
- 3 Define the provisioning action for the Provision block.


- a Click **Add**.
- b Select **Application Invocation** from the **Name** list.
- c Select **Provisioning** in the **Application Name** field.
- d Set **PR** as the provision status result code in the **Return Variable Name** field.
- e Click **Apply** to save these settings.

Task 6: Add a transition from the begin activity to the Provision activity

- 1 Click **Select** .
- 2 Click the **begin** activity then click **Add Transition** .
- 3 Click the **Provision** activity.  
An arrow appears between the two activities.

Task 7: Create the post-provisioning block

Perform the following steps to define the post-provisioning block that executes if the provisioning block is successful.



- 1 Click the **Add PostProvision Block** icon , then click below the Provision activity in the template workspace.  
Adds **Activity\_3**.
- 2 Specify properties for this activity.
  - a Enter **PostProvision** in the **ID** field.
  - b Click **Apply** to save this settings.
- 3 Create a map variable designating the current event, for the case of an error occurring.
  - a Click the **Actions** tab then click **Add**.
  - b Select **Add Item to Map** from the **Name** list
  - c Fill in the fields as follows:
 

<b>New Collection:</b>	<b>Check</b>
<b>Map Variable name:</b>	<b>\$ProvisionMap</b>
<b>Element Name Variable:</b>	<b>“errorEvent”</b>
<b>Element Value Variable:</b>	<b>“PostProvisioning”</b>


- d Click **Apply** to save these settings.
  - e Click **Return** to add another action.
- 4 Add an action that provisions the user in Select Identity.
- a Click the **Actions** tab then click **Add**.
  - b Select **Application Invocation** from the **Name** list
  - c Select **Post Provision** from the **Application Name** list.
  - d Enter **PPC** in the **Return Variable Name** field.
  - e Click **Apply** to save these settings.

**Task 8:** Add a transition from the Provision activity to the PostProvision activity



This transition defines the condition that the provisioning block must be successful for the workflow to execute the post-provisioning block.

- 1 Click the **Select** icon , then click the **Provision** activity.
- 2 Click the **Add Transition** icon , and click the **PostProvision** activity.
- 3 Enter **equal(PR,102)** in the **Condition** field on the **Transit** tab.
- 4 Click **Apply** to save these settings.




**Task 9:** Add an end activity that runs if provisioning is successful

- 1 Click the **Add Activity** icon , then click below the **PostProvision** activity. Adds **Activity\_4**.
- 2 Specify properties for this activity.
- 3 Enter **end** in the **ID** field and click **Apply**.


**Task 10:** Add a transition from the Post-provision activity to the End activity

- 1 Click **Select** .
- 2 Click the **Post Provision** activity then click **Add Transition** .
- 3 Click the **End** activity.  
An arrow appears between the two activities.



### Task 11: Define an activity that logs an error if provisioning is unsuccessful

- 1 Click the **Add Activity** icon , then click to the right of the **Provision** activity.  
**Activity\_5** appears in the template.
- 2 Specify properties for this activity.
  - a Enter **Notify Error** in the **ID** field and click **Apply**.
  - b Click the **Actions** tab, then click **Add**.
  - c Select **Application Invocation** from the **Name** list.
  - d Select **Send an Email Notification** from the **Application Name** list.
  - e Complete the following fields:  
**Email Template:** "Provisioning Failed"  
**Name-value Map Variable:** \$ProvisionMap
  - f Click **Apply** to save these settings.
- 3 Add a transition from the **Provision** activity to the **Notify Error** activity.
  - a Click the **Select** icon  and click the **Provision** activity.
  - b Click the **Transition** icon , and click the **Notify Error** activity.
  - c Click **Apply** to save your settings. Add an error end activity if an error occurs and provisioning was unsuccessful

When the workflow ends with the `errorEnd` activity, the “End with error” status shows on the **Request Status** page of the Select Identity client.


- 1 Click the **Add Activity** icon , then click above the **Notify Error** activity.  
Adds **Activity\_6**.
- 2 Enter **errorEnd** in the Id field and click **Apply**.

### Task 12: Add a transition from the Notify Error activity to the error end activity

- 1 Click **Select** .
- 2 Click the **Notify Error** activity then click **Add Transition** .

- d Click the **Error End** activity.  
An arrow appears between the two activities.

### Task 13: Save the template

Save the template by clicking **Save** .

A message appears in the property tabs panel indicating whether the template was saved successfully.





# 3 Using Files in the Workflow Studio

This chapter describes how to use files with workflows.

This chapter covers the following:

- Using a Properties File to Reactivate a Pending Workflow
- Using the Application Definition File

## Using a Properties File to Reactivate a Pending Workflow

You can reactivate a pending workflow in one of two ways:

- Call a workflow API. See the *HP OpenView Select Identity External Call Developer Guide* for details.
- Use a properties file, described in this section. This provides a means to start or reactivate a workflow instance without writing Java code.

You can create a properties file the workflow engine picks up to reactivate the pending workflow. You can give the properties file any name and place it in a location specified below. Once the workflow is reactivated, the workflow engine deletes the file.

Following are the requirements for this properties file:

- Specify the file path name in the `workflow.listening.folder.path` system property.

You define this system property in the `TruAccess.properties` file, which is located in the `%InstallDir%\sysArchive` directory. See “Configuring TruAccess.properties” in the *HP OpenView Select Identity Installation Guide* for more information.


- The properties consist of a workflow identifier and a list of workflow variables passed to the workflow.
- For an existing workflow instance reactivation, the identifier can be defined in two ways:
  - Use the `_instId_` and `_activityId_` properties together to identify the reactivation point.

```
_instId_ =
_activityId_ =
```

For example:

```
_activityId_=ApprovalOneWait
_instId_=1242
```

The workflow waits to be reactivated at the `ApprovalOneWait` activity for the instance `1242`.

 You need the instance Id. It is required to invoke an existing workflow. You may find the instance id from the request status.

- Define a list of workflow variables below the identifier properties in a format of name-value pair:

```
name1=value1
name2=value2
```

Set the rest of the workflow variables to be new workflow variables when the workflow is reactivated.

Following is a simple example of a properties file (`filexxx.properties`) used for reactivation:

```
_instId=10996
_activityId=Approval1
approverAction=approved
```

# Using the Application Definition File

In the Workflow Studio, the **Actions** → **Application Invocation** → **Application Name** list, contains all the registered workflow applications that a workflow can invoke. See [Define the Actions for the Provision block](#) for instructions on accessing the Application Name list.

Select Identity is shipped with a default Application Definition XML file, which lists some commonly used applications.

You can add additional applications in a separate XML file, which defines how the applications are invoked. You then import the file as a new workflow Application Definition file.

After the new Application Definition file is imported, all the defined applications from various Application Definition XML files are combined into a single application list in the **Application Name** list.

You can define a Java application and/or web service application in the Application Definition file. The following sections show you how to define the application and import the Application Definition file to the Workflow Studio:

- [Defining a Java Application in the Application Definition File](#)
- [Defining a Web Service Application in the Application Definition File](#)
- [Importing the Application Definition File](#)

## Defining a Java Application in the Application Definition File

The following application definition XML file shows how a Java application is defined.

```
<ActionTemplates Id="External Applications" Name="External
Application Defines" xmlns="http://www.wfmc.org/XPDL1.0"
xmlns:xpdl="http://www.wfmc.org/XPDL1.0">
  <ActionDefine Id="CheckEmailVerification"
  DisplayName="Checkemail
  Verification" Name="External Application Defines" >
    <FixedParams>
      <Param Name="type" Value="java">
        <Param Name="classpath" Value="c:/test/classes" />
      </Param>
```

```

        <Param Name="class"
Value="com.hp.ovsi.util.EmailUtil"/>
        <Param Name="method" Value="checkEmail"/>
    </FixedParams>
    <Params>
        <Param Name="id" Value="toInt($RequestId) "
Type="integer"
        DisplayName="request" Hide="true"/>
        <Param Name="addr" Value="addr" Type="string"
        DisplayName="Email Address" Hide="false"/>
    </Params>
</ActionDefine>
...
</ActionTemplates>

```

The Application Definition XML file can contain multiple applications. Each registered application is enclosed within the `<ActionDefinition>` tags. The `<Fixed Params>` tag describes the application general attributes as follows:

- `<Param Name="type" Value="java">`  
The value can be Java or web service. If it is Java, you need to further specify the top folder name (classpath) where the Java package starts if the Java class is not in Select Identity's recognized classpath, as in the example below:
- `<Param Name="classpath" Value="c:/test/classes" />`
- `<Param Name="class" Value="com.hp.ovsi.util.EmailUtil"/>`  
Java class name if type=java.
- `<Param Name="method" Value="checkEmail"/>`  
Java method name if type=java
- `<Params>`  
This tag lists the parameters in the same order as in the Java class method.
- `<Param Name="id" Value="toInt($RequestId) " Type="integer" DisplayName="request" Hide="true"/>`
  - Param/@Name: Unique string to identify the parameter name.
  - Param/@Value: Workflow variable or expression to be passed as an input parameter.

- Param/@Type: Type of parameter. Supported types are:

- string
  - integer
  - date
  - object

The type parameters are mapped to string, integer, date or any other Java data types respectively. The type parameters are used only for validation purposes.

- Param/@DisplayName: Text displayed on the screen as a label for this input parameter.
- Param/@Hide: If true, this parameter is not displayed in the **Action/Application** screen. The default is false.

This Email application is mapped to the following Java method:

```
package com.hp.ovsi.util;
public class EmailUtil {
public boolean checkEmail( int requested, String addr )
throws XXXException {
    ....
}
}
```



The return value is not specified here. It is defined as a workflow variable in the **Action/Application** screen.

Use the default Application Definition XML file shipped with Select Identity as an example, to see how an application is mapped to the Select Identity Workflow Studio screen.

You can also export the Application Definition XML file through the Import/Export Configurations function, modify it, and import it into the Workflow Studio. See the HP OpenView Select Identity Administrator Guide for details on using the Import/Export Configurations function and [Step 3: Import the XML file into Select Identity](#) for instructions on importing the Application Definition XML file.

## Defining a Web Service Application in the Application Definition File

You can define a Web Service application in the Application Definition file. Like the Java method invocation, all meta information about the Web Service invocation is defined in the Application Definition file. The type of invocation (Java or Web Service) is transparent to the workflow studio end user.

Typically, in order to invoke a remote web service, the service provider must publish the following web service meta data:

- 1 Request format. It is typically SOAP request XML
- 2 Service URL
- 3 SOAP Action string
- 4 Response format. It is typically SOAP response XML

The provider can either use these items with workflow template designer in a free text format or include them through a Web Service Definition Language (WSDL) file. In the case of WSDL, you can use WSDL enabled, commercially-available XML tools to extract these items.

## Web Service Metadata

SI currently requires you to configure the web service metadata manually in the Application Definition file. That is, you need to export the Workflow Application Definition file first and create or edit the application action items to configure the web service metadata and then import it back to the SI.

Here are the steps to configure web service metadata in an application action in Application Definition file:

Add the following <Param> tags under `FixedParams`

<b>Name</b>	<b>Value</b>	<b>Required</b>	
type	webService	yes	

Name	Value	Required	
url	Service URL	yes	
body	Body part of SOAP request	no	Use either body or envelope <Param> to specify body.
respXPath			To retrieve the desired data from SOAP response message and return it as workflow variable.

After the web service metadata is defined, you can further define input parameters under <Params>. Any <Param> defined under <Params> appears as input fields in the Action panel of the workflow studio, allowing the workflow template designer to enter the input parameter value expression. This maps the user input to the variable parameters in the request body.

Web service parameters in the XML body definition should be enclosed in double curly braces and referenced as <Param>.

respXPath is used to extract the return data from the Web service response. Alternatively, If respXPath is not specified, the entire SOAP response XML string is returned as workflow variable (specified in Action panel of workflow studio) and you can then subsequently use XPath actions to read individual data (see [Example 1 - Create getGeographicalInfo Web Service](#) below).

### Example 1 - Create getGeographicalInfo Web Service

Many web service returns multiple values in the response message. This example shows how to return two values, city and state names, from a response message. It also demonstrates how to use the request header to specify the login information.

```
<ActionDefine Id="GetGeoInfo" DisplayName="get geographical
info" >
    <FixedParams>
        <Param Name="type" Value="webService" />
        <Param Name="url" Value="http://ws.cdyne.com/
ziptogeo/zip2geo.asmx" />
        <Param Name="soapAction" Value="http://ws.cdyne.com/
GetLatLong" />
```

```

        <Param Name="header" ><![CDATA[
            <m:Header xmlns:m="http://www.xignite.com/
services/">
                <m:Username>abc</m:Username>
                <m>Password>abc</m>Password>
                <m:Tracer>abc</m:Tracer>
            </m:Header>
        ]]></Param>
        <Param Name="body" ><![CDATA[
            <m:GetLatLong xmlns:m="http://ws.cdyne.com">
                <m:zipcode>{{zipcode}}</m:zipcode>
                <m:LicenseKey>{{licenseKey}}</m:LicenseKey>
            </m:GetLatLong>
        ]]></Param>
    </FixedParams>
    <Params>
        <Param Name="zipcode" Value="75093" Type="OBJECT"
DisplayName="zip code" />
        <Param Name="licenseKey" Value="abc" Type="OBJECT"
DisplayName="License Key" />
    </Params>
</ActionDefine>

```

In this example, the header is used to specify the login information for the web getLatLong web service. respXPath is not used since the application intends to return multiple values to workflow instance.

In workflow studio, you add the get geographical info application action as below:

geoXML is specified as a return variable name

To retrieve two values out of response message string stored in geoXML, you need to add two XPath actions:

- "//City"
- "//StateAbbreve"

You can then test whether the city and state data has been assigned to the workflow variables geoCity and geoSt.



If the web service must be invoked through a proxy server, the proxy properties must be set in the `TruAaccess.properties` file as follows:

**`http.proxyHost=web-proxy.rose.hp.com` (proxy server IP or host name)**

**`http.proxyPort=8080` (proxy server port)**

## Importing the Application Definition File

After you create an Application Definition file, you can import the file using a different name than the Select Identity file name. You import the file from the Configurations page on the Select Identity client.

Typically, you will follow these steps to import an Application Definition file:

- 1 Download (export) an existing Application Definition XML file. See [Creating a Custom Report Template](#) for instructions.
- 2 Replace the `<ns1:Body>` part of the file with your new Application Definition.
- 3 Change the value attribute in the `<ns1:key>` tag to a different application definition name.
- 4 Save the file to disk.
- 5 Import the saved file to Select Identity. See [Creating a Custom Report Template](#) for instructions.

If the web service must be invoked through a proxy server, the proxy properties must be set in the `TruAaccess.properties` as follows:

`http.proxyHost=web-proxy.rose.hp.com` (proxy server IP or host name)

`http.proxyPort=8080` (proxy server port)

The method invocation can also be defined through the External Call page. (See the *HP OpenView Select Identity Administrator Guide* for details on using the external call function on the External Call page.)

Following is a comparison of registering a Java method using the Application Definition XML file vs. registering a Java method using an external call:

<b>Java method registered via Application Definition XML</b>	<b>External call Java method registered via External call page</b>
--	--

<p>Any Java class/method can be invoked. It can be a preexisting one that was not created for the purpose of being invoked by a workflow. The Java method is workflow independent.</p>	<p>The Java class must implement certain Select Identity-defined interface parameters. The required method signature (name and parameters) must be followed. The Java method is dependent on the Select Identity external call, and cannot be reused for another purpose.</p>
<p>Input parameters are directly passed into the registered Java method. Output parameters are directly returned by the Java method. You need only write regular Java code.</p>	<p>Workflow variables cannot be directly passed to the external call as input parameters. Input is read in an external call from a Select Identity-supplied API. Output is passed to a workflow by calling a Select Identity-supplied workflow API. The code is workflow dependent and cannot be reused in a non-workflow environment.</p>
<p>The Select Identity graphical user interface is not available. You need to export and edit the XML file and then import it to Select Identity.</p>	<p>The Select Identity interface is available. You edit the external call information directly through the Select Identity user interface using the <b>Service Studio</b> → <b>External Calls</b> menu option.</p>
<p>Web Service is supported.</p>	<p>Only Java invocation is supported.</p>
<p>Application input and output parameters are displayed and can be entered in the Workflow Studio.</p>	<p>Input and output parameters are hidden from end users.</p>

## 4 Creating a Workflow Template

Workflow templates provide several features that support robust logic, giving you great flexibility when creating the flow of workflows. When creating workflow templates, consider the following:

- **Task logic** — Which workflow activities can be handled by simple activities, which should be represented by blocks, and when should you invoke a subworkflow? Blocks provide a way to group activities.

Besides reporting capability, block activities are capable of modeling many business activities such as timeout, AND and XOR split and join, alert, escalation, and so on. Simple activities, on the other hand, can only perform actions executions. However, execution of block activities are time consuming compared with a simple activity. From a performance perspective, consider using a simple activity if it can fulfill your requirements.

- **External applications** — Are there external systems or scripts that will perform tasks during the workflow? If so, you can register external applications with Select Identity so that you can call on them from the Workflow Studio.
- **Branching** — How should one activity progress to the next? Is there a direct path or is the progression based on some condition?
- **Auditing and escalation** — What information do you need to track throughout the life of the workflow? You can do the following to track information:
  - Set information for a single activity.
  - Track information in blocks.
  - Store information to be used throughout the life of the workflow.

Also, before creating a workflow template, you may need to create dependent information using the Select Identity client, as follows:

- If you plan to request approvals from or notify administrators of changes, you can obtain a list of approvers by specifying a role created from the Select Identity **Admin Roles List**. To access this page, select the **Tools** → **Admin Roles** → **Admin Role List** menu option. See “Administrative Roles” in the *HP OpenView Select Identity Administrator Guide* for details. For an example of creating a parallel approval workflow, see [Parallel Approval Workflow Design](#) on page 132.
- If you intend to send email as part of the workflow, you can reference email templates created on the **Notification Template List**. To access this page, select the **Service Studio** → **Notifications** menu option. See “Service Studio” in the *HP OpenView Select Identity Administrator Guide* for details.
- If you must perform provisioning actions on external systems, such as to enforce entitlement rules, you make calls to these systems using external calls.

You can register the applications in one of the following ways:

- Using an XML file based on the format described in [Using the Application Definition File](#) on page 51. You export the xml file first, add or modify the application definitions, and then import the file to Select Identity through the Select Identity **Import/Export Configurations** menu option. See [Step 3: Import the XML file into Select Identity](#) on page 124 for instructions on importing an xml file.
- Registering the external calls with Select Identity through the **Service Studio** → **External Calls** menu option. See [Task 2, Register the rule with Select Identity](#) on page 128.

Keep the following in mind when creating workflow templates:

- When specifying information for an activity, be sure to click the **Apply** button to save your changes before loading other pages. If you do not update the Workflow Studio with your changes, you may lose newly updated properties.
- If you have an existing workflow template open, do not open another workflow template. If you open a second workflow template without closing the first workflow template, the Workflow Studio loads the second template in the current window, overwriting unsaved changes.

It is recommended that you create a practice workflow template, to practice using the Workflow Studio controls. The Workflow Studio controls can be tricky to use at first. See [Getting Started](#) on [page 7](#) for examples to help you become familiar with the interface.

When you are familiar with the Workflow Studio, create a simple workflow template you can assign to a request event in a Service Role. Then, build upon the simple template or create a template that you can truly use in user provisioning.

## Overview of Template Actions

You can use workflow templates to perform an array of tasks in addition to provisioning users. Workflow Studio's flexibility enables you to create templates to process the most complex provisioning actions. In general, there are a specific number of actions you need to perform as part of the provisioning process.

The following table lists the tasks you can perform and the template action you must configure to represent the action in Workflow Studio.

<b>Tasks</b>	<b>Actions in the Workflow Studio</b>
<p><b>Variables:</b> Retrieve and set variables to pass data to activities in a workflow and to external processes</p>	<ul style="list-style-type: none"> <li>• <a href="#">Add Item to List on page 100</a></li> <li>• <a href="#">Add Item to Map on page 100</a></li> <li>• <a href="#">XPath on page 102</a></li> <li>• <a href="#">Set Variable on page 97</a></li> <li>• <a href="#">Set Block Variable on page 103</a></li> <li>• <a href="#">Get Block Variable on page 103</a></li> </ul>
<p><b>Approvers:</b> Obtain and notify lists of approvers</p>	<ul style="list-style-type: none"> <li>• <a href="#">Create Workflow Approver Task on page 88</a></li> <li>• <a href="#">Application Invocation → Get Approvers By Role on page 86</a></li> <li>• <a href="#">Application Invocation → Notify Approvers on page 87</a></li> <li>• <a href="#">Application Invocation → Notify Approvers (Default Email Template) on page 87</a></li> <li>• <a href="#">Application Invocation → Approvers External Call on page 95</a></li> </ul>
<p><b>Email:</b> Obtain email addresses and send email</p>	<ul style="list-style-type: none"> <li>• <a href="#">Application Invocation → Email Notification on page 90</a></li> <li>• <a href="#">Application Invocation → Send an Email Notification on page 91</a></li> <li>• <a href="#">Application Invocation → Check Email Verification on page 85</a></li> <li>• <a href="#">Send Email on page 101</a></li> </ul>

Tasks	Actions in the Workflow Studio
<p><b>Notifications:</b> Send notifications using the notification policy and templates</p>	<ul style="list-style-type: none"> <li>• Application Invocation → <a href="#">Check Email Verification on page 85</a></li> <li>• Application Invocation → <a href="#">Email Notification on page 90</a></li> <li>• Application Invocation → <a href="#">Send an Email Notification on page 91</a></li> <li>• Application Invocation → <a href="#">Notify Approvers on page 87</a></li> <li>• Application Invocation → <a href="#">Save Email Notification on page 86</a></li> </ul>

Tasks	Actions in the Workflow Studio
<p><b>Logging and errors:</b> Log messages to the Select Identity log handler and handle errors</p>	<ul style="list-style-type: none"> <li>• <a href="#">Log Message</a> on page 98</li> <li>• <a href="#">Throw Exception</a> on page 98</li> <li>• <a href="#">Recover From Last Error</a> on page 102</li> </ul>
<p><b>Others</b></p>	<ul style="list-style-type: none"> <li>• <a href="#">Run Script</a> on page 98</li> <li>• <a href="#">Call Subworkflow</a> on page 99</li> </ul>
<p><b>Provisioning:</b> Provision users in Select Identity and external resources, and invoke scripts and external calls</p>	<ul style="list-style-type: none"> <li>• <a href="#">Application Invocation</a> → <a href="#">Provisioning Task</a> on page 89</li> <li>• <a href="#">Application Invocation</a> → <a href="#">Post Provision</a> on page 92</li> <li>• <a href="#">Application Invocation</a> → <a href="#">Post Provision in Reconciliation to Save Data</a> on page 94</li> <li>• <a href="#">Application Invocation</a> → <a href="#">External Call</a> on page 95</li> <li>• <a href="#">Approvers External Call</a> on page 95</li> <li>• <a href="#">Application Invocation</a> → <a href="#">Add a Set of Users to a List of Services</a> on page 97</li> </ul> <p>Also, you may want to create a retry block for each provisioning activity or block. If provisioning fails, the retry block can wait for a specified amount of time then retry provisioning again. See the <a href="#">Default Workflow Templates</a> on page 12 for examples.</p>

## Properties and Variables

When creating a workflow template, you can define properties and variables. Both properties and variables can be referenced in the report template to display information on the Request Status page on the Select Identity client.



- Properties define constant data when the workflow template is created. Properties exist in the level of a workflow template, block and activity.
- Variables pass data to and from the workflow engine. Workflow variables exist in the level of a workflow instance and a block.

This section covers the following:

- [Using Properties](#)
- [Using Variables](#)

## Using Properties

A property is simply a name-value pair. The value is a string, which instructs the workflow how to operate. Since properties define constant data, the property values do not change at runtime.

A workflow property is defined at the workflow template level. A block property is defined at the block level. For a single block activity, it is defined in the block activity. For a block constructed with start and end block activities, the block is defined in the end block activity. An activity-level property is defined in activity. See [Activities and Blocks](#) on [page 78](#) for details.

The specified properties can be read by external applications using the Workflow API provided by Select Identity. The properties can also be referenced by a report template to show relevant information in a status report.

Some properties are defined by the workflow engine (see [System-Defined Properties](#) on [page 66](#)). Assign values to these properties when you create the workflow template to instruct the workflow how to operate.

For example, you can use the Join Count block level property value to specify how many times a block activity must be reactivated before transitioning to the next block activity. The Join Count property can be specified in an end block activity or single block activity. See [Join Count Examples](#) on [page 73](#) for examples.

## System-Defined Properties

The following table describes the system-defined properties:

<b>Property</b>	<b>Where Defined</b>	<b>Description</b>
Report TemplateId	Workflow Information tab (global attribute for the template)	Specifies the name of the report template to use. The engine will use this report template to render the workflow instance data. If you don't set this property, the engine uses the default report template. See <a href="#">Report Templates</a> on page 110 for more information.
Block ID	Block activity	For block activities (activities that represent an entire block), this property replaces the Start block ID and End Block ID properties used for multiple-activity blocks. If you use the Add Provision, Add Post Provision, or Add Approval buttons to create a block activity, this property is automatically set to activityId by default.
Start block ID	Start block activity	Sets the block ID in a start block activity. This value must be unique within the template and must match the value of End Block ID. The workflow engine uses this ID to identify a block. Any activities between the start and end block activities are included in the same block.
End Block ID	End block activity	Sets the block ID in the end block activity. The assigned value must match that of Start block ID.

<b>Property</b>	<b>Where Defined</b>	<b>Description</b>
Block Type	End block activity and block activity	<p>Specifies the block type, which is used by the report engine when rendering information in the status report. All blocks of the same type in a workflow template are rendered with the same format according to the report template.</p> <p>If Block Type is not specified, Select Identity will not render the report for this block. The following block types are defined in the default report template, in addition to the default ReportBlock:</p> <ul style="list-style-type: none"> <li>• For blocks that verify email, specify <code>emailVerify</code> to report on this block.</li> <li>• For approval blocks, specify <code>approval</code> to report on this block.</li> <li>• For the provisioning block, specify <code>provisioning</code> to report on this block.</li> <li>• For the post-provisioning block, specify <code>postprovisioning</code> to report on this block.</li> <li>• Specify <code>ReconciliationPostProvision</code> to report on reconciliation blocks.</li> </ul>

<b>Property</b>	<b>Where Defined</b>	<b>Description</b>
Join Count	End block activity and block activity	<p>Specifies the total number of anticipated joining transitions. When this number is equal to the joined count, the end block or AND join activity exits and the workflow transitions out of the current block.</p> <p>You can use Join Count to model the number of approvers who must take approval action before the workflow can continue.</p> <p>Internally, there is block variable called <code>joinedCount</code>. It is incremented each time the workflow is reentered, which reflects approver action. When joined count reaches join Count, the workflow exits the block.</p> <p>Both join Count and joined count can be set at runtime with variable names to be <code>joinCount</code> and <code>joinedCount</code> respectively. See <a href="#">Join Count Examples</a> on page 73 for details.</p>
Role Name	End block activity and block activity	Specifies the role name (defined in Select Identity) that is used by the Get approvers by role action defined in the same activity.
Escalate To	End block activity and block activity	Specifies the email address of the user to whom the escalation message is sent. Specify this property in an approval block.
Escalation Handler	End block activity and block activity	Specifies how the escalation is delivered. Specify a string (but do not quote the string). Currently, you can only specify <b>email</b> as the value of this property. Specify this property in an approval block.

<b>Property</b>	<b>Where Defined</b>	<b>Description</b>
Escalation	End block activity and block activity	<p>Sends a reminder notification to the address specified by <b>escalateTo</b> when the workflow instance times out. This property uses the “New User account - Escalation” notification template when sending emails.</p> <p>Specify the timeout value in the following format:</p> <pre>[n day[s]] [ x hour[s]   hr[s]] [y minute[s]   min[s]] [z [second[s]   sec[s]]]</pre> <p>where [ ] indicates an optional parameter and   indicates OR. If you specify an integer, the workflow engine interprets the time as seconds.</p> <p>The following are examples:  7 days  1 day 12 hrs 30 mins  45 seconds  3600</p> <p>Set this property in an approval block.</p>
Escalation timeout	End block activity	<p>Specify the timeout value in the following format:</p> <pre>[n day[s]] [ x hour[s]   hr[s]] [y minute[s]   min[s]] [z [second[s]   sec[s]]]</pre> <p>where [ ] indicates an optional parameter and   indicates OR. If you simply specify an integer, the workflow engine interprets the time as seconds.</p>
Escalation Repeat Count	End block activity and block activity	<p>Directs the workflow engine to repeat the escalation notification. Specify how many times to repeat the escalation (integer). Specify this property in an approval block.</p>

<b>Property</b>	<b>Where Defined</b>	<b>Description</b>
Alert Handler	End block activity and block activity	Specifies how the alert is delivered. Specify a string (but do not quote the string). Currently, you can only specify <b>email</b> as the value of this property. Specify this property in an approval block.
Alert Timeout	End block activity and block activity	Sends an alert to the requestor's address when the workflow instance times out. This property uses the "New User account - Alert" notification template when sending emails. Specify the timeout value in the following format: [n day[s]] [ x hour[s]   hr[s]] [y minute[s]   min[s]] [z [second[s]   sec[s]] where [] indicates an optional parameter and   indicates OR. If you simply specify an integer, the workflow engine interprets the time as seconds. The following are examples: 1 day 5 hrs 20 mins 12 hours 2400 Set this property in an approval block.
Alert Repeat Count	End block activity and block activity	Directs the workflow engine to repeat the alert. Specify how many times to repeat the alert (integer). Specify this property in an approval block.
Reminder Handler	End block activity and block activity	Specifies how the reminder is delivered. Specify a string (but do not quote the string). Currently, you can only specify <b>email</b> as the value of this property. Specify this property in an approval block.

<b>Property</b>	<b>Where Defined</b>	<b>Description</b>
Reminder Timeout	End block activity and block activity	<p>Sends a reminder notification to all the approvers assigned this approval request when the workflow instance times out. This property uses the “New User account - Reminder” notification template when sending emails.</p> <p>Specify the timeout value in the following format:</p> <pre>[n day[s]] [ x hour[s]   hr[s]] [y minute[s]   min[s]] [z [second[s]   sec[s]]</pre> <p>where [] indicates an optional parameter and   indicates OR. If you simply specify an integer, the workflow engine interprets the time as seconds.</p> <p>The following are examples:  2 days 5 hours 30 minutes  1 day  3600</p> <p>Set this property in an approval block.</p>

<b>Property</b>	<b>Where Defined</b>	<b>Description</b>
Reminder Repeat Count	End block activity and block activity	Directs the workflow engine to repeat the reminder notification. Specify an integer representing the number of times to repeat the reminder. Specify this property in an approval block.
Timeout At	End block activity or block activity	Transitions out of the block when the timeout occurs. Specify the timeout value in the following format: 01/05/2006 14:00:00
Timeout After	End block activity or block activity	Transitions out of the block when the workflow instance times out. Specify the timeout value in the following format: [n day[s]] [ x hour[s]   hr[s]] [y minute[s]   min[s]] [z [second[s]   sec[s]] where [ ] indicates an optional parameter and   indicates OR. If you simply specify an integer, the workflow engine interprets the time as seconds. The following are examples: 1 day 12 hrs 30 minutes 1 hour 60.

## Timeout Values

Timeout values can also be set at runtime by invoking the Set Block Variable action. For example, to set an absolute timeout value in a single activity block, select the activity, select Set Block Variable from the actions drop down list and enter the values as follows:

```
Name                : SetBlockData
Block Variable      : timeoutLength
Name                :
Block Variable      : maxApprovalTimeout
timeoutLength is the block variable name
```



`maxApprovalTimeout` is a variable set somewhere before the Set Block Variable is invoked

For timeout at a specific time, set the following parameters in the Set Block Variable action.

```
Name                : SetBlockData
Block Variable      : timeoutTime
Name                :
Block Variable      : terminateAt
```

`timeoutTime` is the block variable name for absolute timeout block variable

`terminateAt` is a workflow variable set before this action is invoked

For blocks with start and end block activities, SetBlockData must be invoked in the start block activity so that the block variable can be passed to the workflow engine for timeout handling.

## Join Count Examples

Figure 3 shows how to use the Join Count property for a block with start and end block activities, or a single block activity. These different block implementations are functionally equivalent. In both cases, the Join Count property specifies that the block activity must be reactivated three times before transitioning to the next activity. To leave the block, the workflow must be reentered three times by approvers who perform an approval action, or by one of the approvers who performs a reject action.

Figure 4 shows an AND split workflow in which the Join Count property specifies the number of transitions the EndBlockActivity must receive to transition to NextActivity. The transitions can also take place when approvers reject the request. This example shows two possible scenarios:

- Both `condition1` and `condition2` are satisfied, and the workflow then waits for approval actions in `wait1` and `wait2` activities. When `wait1` is reactivated once and `wait2` is reactivated twice. This makes the total reactivation equal to three. As a result, the workflow transitions to `NextActivity`.
- Only `condition2` is satisfied. `wait2` must be reactivated three times to transition from the `EndBlockActivity`.

## Using Variables

A variable defines dynamic data that is set and changed while the workflow instance is running. Like properties, a variable is a name-value pair. Variables can be created or changed at run-time in a workflow instance through Actions, a Workflow API call, or returned by an Application Invocation.

This section covers the following:

- [Create or Changing a Variable](#)
- [Block Variable](#)

### Create or Changing a Variable

To create or change a variable in a workflow instance, use the Set Variable action. See [Set Variable](#) on [page 97](#) for details.

Assign a qualified variable string to the name of the variable. In addition, keep these guidelines in mind when naming a variable:

- The name cannot include spaces.
- The first character of the name cannot be a number
- The only special characters allowed in variable names are underscores ( `_` ) and dollar signs ( `$` ). Otherwise, use only alpha-numeric characters.

You can assign any of the following to the value of a variable:

- A constant (string, integer, and so on)  
If you assign a string constant, quote the string by surrounding it with double quotes. Here is an example: `Var1 = "this is a string"`. When `Var1` is evaluated, it resolves to `This is a string`.

If you assign an integer to a variable, do not quote the value, as in this example: `Var2 = 4`

- Another variable  
Simply assign the name of the variable as the value of your variable. If a variable called `Data` is assigned the string `"This is data"`, then assigning the `Data` variable to `Var2` (`Var2 = Data`) resolves as follows:

```
Var2 = "This is data"
```

- **An expression**  
An expression can contain a mixture of variables and constant values. For example, if a workflow variable called `Status` is assigned the value `OK`, the following expression will resolve as `The status is OK`:

```
"The status is" + Status
```

This illustrates an expression that appends the value of a workflow variable to a string.

A set of workflow macros are defined for use in expressions. For example, in a transition condition, you can use the `equal()` macro to compare two objects, as in this example: `equal (approved, "approved")`. The workflow engine will evaluate it and make a decision based on the returned result.

Following are the macros that can be used in workflow templates:

`equal (a, b)` — if `a` equals `b`, the macro returns `true`, otherwise `false`

`notEqual (a,b)` — if `a` is not equal to `b`, the macro returns `true`, otherwise `false`



You cannot change the variable's type after the variable is initialized.

Workflow Studio provides the following categories for variables:

- **Persisted** — Variable names of persisted variables begin with **\$** and are stored in the database, even when a workflow instance is passivated. You can access these variables at any time once the workflow instance is created.
- **Non-persisted** — This type of variable is temporary and available in the workflow until a wait activity begins (the workflow is passivated) or until the workflow completes (the workflow terminates).

You can reference workflow variables in workflow templates, report templates, and in Workflow API calls.

## Workflow Engine-Defined Variables

Workflow engine internally defines the following variables. These variables can be read by templates as normal workflow variables but cannot be modified.

Name		Access Scope	Possible Values
<code>\$_instId</code>	Workflow instance ID	instance	
<code>\$_blockId</code>	Block ID	block	
<code>_activityId</code>	Activity ID	Activity	
<code>_joinCommand</code>	Command string set by external call when reactivating pending workflow	Template can access this variable after block activity exits	<ul style="list-style-type: none"><li>• <b>exit</b>: force block to exit regardless of join count property. Typically used in rejection at approval stage</li><li>• <b>exitAll</b>: force block and all its parent blocks to exit</li><li>• <b>timeout</b>: after block timeout. This can be checked to handle approval escalation.</li></ul>

## Block Variable

A block property defined in the end block activity (or single block activity) is a static parameter and cannot be changed once the workflow template is saved in the Workflow Studio. But, the block variable is a runtime dynamic variable that is evaluated during the workflow execution. The block variable is also persisted so that it can be accessed after the workflow instance is passivated.

Some workflow engine-defined block properties can be overridden by block variables at run-time. The following workflow engine-defined block properties can be overridden by block variables:

Property Name	Variable Name
JoinCount	joinCount
timeoutAfter	timeoutLength
timeoutAt	timeoutTime

The block variable allows you to change the block behavior at runtime according to running conditions. For example, you can dynamically change the join count or timeout value to alter the block behavior.

A block variable is identified by `blockId` and variable name. You can set the block variable in a Java application through the Select Identity API or in an activity within a block in the following ways:

- Using the Set Block Variable action in the Workflow Studio. See [Set Block Variable](#) on page 103 for details.
- Using the Java API — see the *HP OpenView Select Identity External Call Developer Guide* (`docs/api_help/external_calls / external_calls.pdf`) and locate information about the `clientintf.jar` file.

You can use the Get Block Variable action to read in the block variable. The Get Block Variable action copies the block variable to a workflow instance variable within the workflow, where specified. The Get Block Variable action is then executed either inside or outside a block variable. See [Get Block Variable](#) on page 103 for details.

The block variable can also be read in a Java application through a Select Identity API.

# Activities and Blocks

Activities define the tasks that must occur in the workflow. When assigning a name to an activity, ensure that it is unique; activity names (IDs) must be unique in a workflow template.

You can create a group of activities that comprise a block. A block has the following purposes:

- Allows the block scope variables to be accessed, because these variables are always persisted.
  - ▶ Also does the perform action execution functionality, which a simple activity also does.
- Enables control logic to be executed within the block level base on the block level property and variable settings.
- Provides block-level reporting.

A block can be created in either of two ways:

- Group of activities

This block appears in the Workflow Studio as one or more activities, which must be bounded by start and end block activities. If you choose to create this type of block, you must follow these guidelines:

- Create the start block activity and assign the Start block ID property to it.
- Add all activities and transitions beneath the start block activity. The start block activity can perform actions, though you may want to perform actions after the start block activity for simplicity.
- Create an end block activity and assign the End Block ID property to it. The End Block ID property *must* match that of the Start block ID property. Also, an end block activity cannot be a wait activity.
- The end block activity should assign the result of the block logic and should set any properties that must be passed back to the workflow. It can also log results to the log handler. Other than this, the end block activity should not perform other actions.

- Any activity within the block cannot jump to an activity outside of the block without going through the end block activity. Likewise, no activity can jump into the block without going through the start block activity.

- Single block activity

This block appears in the Workflow Studio as one activity, which represents a block, and performs all the actions that take place within the group of activities. In this case, you do not use the start and end block activities. You can use the following four types of single block activities:

- Approval
- Provisioning
- Post Provisioning
- (general) Block Activity.

For examples of single block activities, see the default templates: [SI OneStageApproval](#) on page 29 and [SI ThreeStageApproval](#) on page 32. A single block activity functions the same as a block composed of start and end block activities. Using a single block activity simplifies the block creation at template design time.

The workflow engine supports the following special activities:

- begin — If you intend to assign the workflow template to a request event, you must create a begin activity as the first activity in the template. Assign **begin** (using all lowercase letters) as the name of the activity. You must create a begin activity as the first in the template. The workflow engine launches a fault handler if it cannot determine where to begin.

If the template will not be referenced by a request event, creating a begin activity is not necessary. For example, a subworkflow invoked by another workflow does not require a begin activity. The parent workflow can explicitly specify the starting activity in the subworkflow (using the activity ID variable). Likewise, workflow templates can be invoked by external applications using the Workflow API. If the workflow is invoked by the API, the begin activity is not necessary and the API can specify the first activity with which to start.

- Fault handler (also called catch exception activity)— An activity that catches exceptions that are launched when the workflow executes.

The fault handlers can be defined to catch the exceptions at various levels, as listed in the following table:

Level	Exception Caught	Implementation
<b>Block</b>	<p>Each block can have its own fault handler.</p> <p>The associated block fault handler can catch exceptions that are launched from any activity in the block, if the fault handler is not caught by other activity-level fault handlers within the block.</p>	<ul style="list-style-type: none"> <li>• Add a property in the fault handler activity.</li> <li>• Specify the property name to be <code>faultHandlerId</code> and value to be <code>block ID</code> for the associated block.</li> </ul>
<b>Instance</b>	<p>Each instance can have its own fault handler.</p> <p>The associated instance fault handler can catch exceptions from activities in a workflow, if the exceptions are not caught by other fault handlers at the block or activity level.</p>	<p>Do one of the following:</p> <ul style="list-style-type: none"> <li>• Designate an activity for fault handling by assigning <code>catchException</code> as the name of the activity.</li> <li>• Alternatively, add a property in any activity used to handle an instance level fault. Specify its name to be <code>faultHandlerId</code> and value to be <code>InstFaultHandlerId</code>.</li> </ul> <p>Note that only one instance fault handler is allowed per workflow template.</p>

Once an exception occurs, it propagates starting from the inner-most level (activity) to the outer-most level (instance) until it is caught.

For example, when a provision block launches an exception, if there is a provision block fault handler, this handler activity is activated and flow continues from this activity forward. If no block activity is defined, the workflow engine tries the instance-level fault handler, and if found, this fault handler is activated. If the instance-level fault handler is not found, the exception propagates to the Select Identity application, which logs the error.

- `errorEnd` — You may create an activity named `errorEnd` that terminates the workflow if an error occurs. When the workflow ends with the `errorEnd` activity, the End with error status appears on the Request Status page of the Select Identity client. If the workflow ends with any other activity, the Request Status page shows the End status.



You can view the Request Status through the **Requests** → **Request Status List** menu option on the Select Identity client. See the *HP OpenView Select Identity Administrator Guide* for details.

## Setting Properties on the Activity Page

You can set the following properties on the **Activity** page for activities:

<b>Property</b>	<b>Description</b>
<b>ID</b>	The name or ID of the activity.
<b>Join Type</b>	<p>If more than one incoming transition is defined for the activity, this list is displayed on the Activity tab for the selected activity. It enables you to specify that the activity transitions to the next activity if all of the incoming transitions are received (AND) or if the activity transitions to the next activity after only one transition condition is made (XOR).</p> <p>Join type transitions are usually paired with split type transitions, and XOR is the default join type. Also, if no condition (join or split) is specified for a transition, the transition always occurs.</p>
<b>Wait Activity check box</b>	Defines this activity as a wait activity. Only persistent instance variables and block variables are persisted after a wait activity.
<b>Split Type</b>	<p>If more than one outgoing transition is defined for the activity, this list is displayed for the selected activity. It enables you to specify that subsequent activities execute if all transition conditions are met (AND) or if only one transition condition is met (XOR). XOR is the default split type.</p>

# Defining Actions

Activities contain actions to define the steps in the tasks. Actions actually do the work in the workflow.



Actions are executed in the order in which they appear in the Actions list.

The workflow engine provides a set of actions you can assign and configure for activities. These engine-defined actions provide common workflow data and control operations, such as the [Set Variable](#) action (see [Set Variable](#) on [page 97](#) or the [Log Message](#) action (see [Log Message](#) on [page 98](#)). Application-specific actions specific to Select Identity are also provided.

These actions are preregistered with the Workflow Studio to enable you to perform general and application-specific tasks within an activity.

To assign an action in the Workflow Studio:

- 1 Select a block activity in the workflow template.
- 2 Click the **Actions** tab in the property tabs area.
- 3 Click **Add** and select an action.

If you select an existing action from the list first, the new action will be inserted into the list before the selected action.

Enter the information for the action as described in the following sections:

- [Application Invocation](#)
- [Set Variable](#)
- [Log Message](#)
- [Throw Exception](#)
- [Run Script](#)
- [Call Subworkflow](#)
- [Add Item to List](#)
- [Add Item to Map](#)
- [Send Email](#)
- [Recover From Last Error](#)

- XPath
- Set Block Variable
- Get Block Variable
- Application Invocation

The Application Invocation action enables you to call an external application. Select Identity provides many applications you can use, as described in this section.

Action	Description
<b>Application Name list</b>	<p>The external application. Select Identity includes the following default applications in the <b>Application Name list</b>. These applications are included in the Application Definition file. You can add applications by creating and importing a new Application Definition file. See <a href="#">Using the Application Definition File on page 51</a> for details.</p> <p>Following are the available applications:</p> <ul style="list-style-type: none"> <li>• <a href="#">Check Email Verification</a></li> <li>• <a href="#">Save Email Notification</a></li> <li>• <a href="#">Get Approvers By Role</a></li> <li>• <a href="#">Notify Approvers</a></li> <li>• <a href="#">Notify Approvers (Default Email Template)</a></li> <li>• <a href="#">Provisioning Task</a></li> <li>• <a href="#">Email Notification</a></li> <li>• <a href="#">Send an Email Notification</a></li> <li>• <a href="#">Post Provision</a></li> <li>• <a href="#">Stated Post Provision</a></li> <li>• <a href="#">Post Provision in Reconciliation to Save Data</a></li> <li>• <a href="#">External Call</a></li> <li>• <a href="#">Approvers External Call</a></li> <li>• <a href="#">Add a Set of Users to a List of Services</a></li> <li>• <a href="#">Create work flow approver task</a></li> <li>• <a href="#">Provision and Post Provision in Reconciliation to save data</a></li> <li>• <a href="#">Update Post Provision Status in Reconciliation,</a></li> <li>• <a href="#">Resource Recon Post Provision</a></li> <li>• <a href="#">Post Provision in Bulk Processing to save data</a></li> </ul>

<b>Fields</b>	<b>Description</b>
<b>Asynchronous Invocation</b> check box	The invocation mode. For synchronous invocation, the workflow is blocked until one of the following occurs: <ul style="list-style-type: none"> <li>• The application completes.</li> <li>• The main workflow launches a fault handler.</li> </ul> check this box for this application.
<b>Return Variable Name</b> field	The name of a workflow variable that holds the returned value of this application invocation. Only applications that run in synchronous invocation mode can return a value. Otherwise, leave this field blank.

## Check Email Verification

This application determines whether an email address exists in Select Identity. The Check email Verification action returns a boolean indicating the result. Set the following fields for this application

<b>Fields</b>	<b>Description</b>
<b>Asynchronous Invocation</b>	check this box for this application.
<b>Return Variable Name</b>	Leave this field blank.

## Save Email Notification

The Save Email Notification application enables you to create a notification for a user indicating that the user must verify the specified email address. This application must be invoked from a wait activity. Set the following fields for this application:

Field	Description
Asynchronous Invocation	check this box for this application.
Return Variable Name	Leave this field blank.

## Get Approvers By Role

The Get approvers by role application enables you to retrieve a list of approvers based on their role. This list is typically passed to subsequent application calls (such as NotifyApprovers) for further processing. The role must exist within Select Identity. Set the following properties for this application:

Field	Description
Asynchronous Invocation	Do not check this box for this application.
Return Variable Name	For this application, specify the list of approvers who are assigned the specified role, for example: <code>apprvList</code> .

In addition, you must set the Role Name property for this application. Set this variable in the end block activity. Click **Properties** and enter a property representing the role. Assign a string to the property.



Like any other properties, the Role name property value is not quoted.

Example: **Role Name: Workflow Approver**

## Notify Approvers

This application notifies approvers of a pending workflow task using the notification mapping created in the Service's Service Role. The approvers are those in the list variable passed to the application. The list is usually created by calling Get Approvers By Role (see [Get Approvers By Role](#) on page 86).



This application sends email to all approvers in the list. Therefore, if an notification template is used, the **To** field is ignored.

Set the following fields for this application:

Field	Description
<b>Asynchronous Invocation</b>	Do not check this box if you wish to store the return value for this application.
<b>Return Variable Name</b>	In this case, specify the name of a variable that can store the status (boolean) of the application if you wish to store the status. It returns true if all of the approvers are sent email notifications and false if a problem occurs sending the email.
<b>Notification Action</b>	The name of the notification mapping in the Service. Example: Approve
<b>Approver List Variable</b>	The list of approvers. You can specify a variable. Example: <code>apprvList</code>

## Notify Approvers (Default Email Template)

This application is similar to the Notify Approvers application, except in this application, you can specify the notification template you want to use in the workflow. This is important if you want to use a workflow other than what is defined in the Service, or if the action is called while performing a reconciliation action that may not be tied to a Service.

Set the following fields for this application:

Field	Description
<b>Asynchronous Invocation</b>	Do not check this box if you wish to store the return value for this application.
<b>Return Variable Name</b>	In this case, specify the name of a variable that can store the status (boolean) of the application if you wish to store the status. It returns true if all of the approvers are sent email notifications and false if a problem occurs sending the email.
<b>Notification Action</b>	The name of the notification mapping in the Service. Example: Approve
<b>Approver List Variable</b>	The list of approvers. You can specify a variable. Example: apprVList
<b>Default Email Template</b>	The name of the email notification template you want to use.

## Create Workflow Approver Task

This application creates a task for all approvers in the specified list on the **Request Worklist** (select **Requests** → **Request Worklist** on the Select Identity interface). Call this application in wait activities only. The approver list is populated by the Get approvers by role action (see [Get Approvers By Role](#) on page 86).



Set the following properties for this application:

Field	Description
<b>Asynchronous Invocation</b>	Do not if you wish to store the return value for this application.
<b>Return Variable Name</b>	For Create Workflow approver task, specify a variable name to store the return value to indicate the status (a boolean) of the application invocation if you wish to store the status of this application. It returns true if all the relevant information regarding the workflow approval task persists in the database successfully, and false if not. Example: wftask

In addition, Create Workflow Approvers requires the following parameter:

Parameter	Description
<b>Approver List Variable</b>	The list of approvers, which was returned by the Get approvers by role action. Specify a variable, such as <code>apprvList</code> .

## Provisioning Task

This application enables you to provision users on target resources, which are defined by the Service. If provisioning fails on a resource, provisioning on previous resources is rolled back. You can call this application from within a wait activity only.

Set the following fields for this application:

Fields	Description
<b>Asynchronous Invocation</b>	Do not check this box for this application.
<b>Return Variable Name</b>	The name of a workflow variable that holds the returned value of this application invocation. Only applications that run in the synchronous invocation mode can return a value.

This application creates a variable called `provisioningResult`, which indicates the status of the `ProvisionTask` operation. You can use in transition conditions and other activities. The following status codes are returned:

- 0 — pending
- 101 — in progress
- 102 — success
- 103 — failure
- 104 — waiting for response
- 105 — reset password failure

This operation also creates a variable called `errorCode`, which indicates the error information for this operation:

- 0 — no error
- 1001 — no resource(s) to provision

## Email Notification

This application sends an email notification to the user being provisioned using a notification template to send the email.



Note that this application sends email to the request target. Therefore, if a notification template is used, the `To` field is ignored. To send an email to someone other than the request target, use the [Send an Email Notification](#) on [page 91](#).

Set the following fields for this application:

<b>Fields</b>	<b>Description</b>
<b>Asynchronous Invocation</b>	Do not check this box if you wish to store the return value for this application.
<b>Return Variable Name</b>	The name of a workflow variable that holds the returned value of this application invocation. Only applications that run in the synchronous invocation mode can return a value.
<b>Email Template</b>	The name of the Select Identity email template to use. Specific the workflow variable name or constant template name string, if it is a constant string. Quote (with double quotes) this string.

## Send an Email Notification

This application is used to notify anyone other than the requestor or user being provisioned. For example, you could send email to the user's manager who is not a user provisioned in Select Identity. This application takes the notification template as an argument but also requires another argument, a map object. You could, for example, create a map object using the [Add Item to Map](#) action to contain the addresses then pass that object to this application. The application then sends email to the address in the **To** field of the notification template.

Set the following fields for this application:

<b>Fields</b>	<b>Description</b>
<b>Asynchronous Invocation</b>	Do not check this box if you wish to store the return value for this application.

<b>Fields</b>	<b>Description</b>
<b>Return Variable Name</b>	In this case, specify a variable name to store the status (a boolean) if you wish to store the status of this operation. It returns true if the notification was sent to the email target, and false if an exception occurred in the process.
<b>Email Template</b>	The notification template to use when sending the email. Specify the name of the template in quotes.
<b>Name-value Map Variable</b>	A map object variable that contains a list of name-value pairs to be used to substitute matched parameters in the email template. Create a map by calling the Add Item to Map action (see <a href="#">Add Item to Map</a> on page 100) before this action.

## Post Provision

This application synchronizes the Select Identity database with the users that are provisioned (after all provisioning completes successfully). Set the following fields for this application:

<b>Fields</b>	<b>Description</b>
<b>Asynchronous Invocation</b>	Do not check this box if you wish to store the return value for this application.
<b>Return Variable Name</b>	In this case, specify a variable name to store the status (a boolean) if you wish to store the status of this operation. It returns true if post provisioning succeeds, and false if it fails.

## Stated Post Provision

This application is similar to the Post Provision application, except that this application provides more details in the Request Status report (access through the **Request** → **Request Status List** menu option on the Select Identity interface). Set the following properties for this application:

<b>Fields</b>	<b>Description</b>
<b>Asynchronous Invocation</b>	<p>The invocation mode. For synchronous invocation, the workflow is blocked until one of the following occurs:</p> <ul style="list-style-type: none"><li>• The application completes.</li><li>• The main workflow launches a fault handler.</li></ul> <p>Do not check this box if you wish to store the return value for this application.</p>
<b>Return Variable Name</b>	<p>The name of a workflow variable that holds the returned value of this application invocation. Only applications that run in the synchronous invocation mode can return a value.</p> <p>In this case, specify a variable name to store the status (a boolean) if you wish to store the status of this operation. It returns true if post provisioning succeeds, and false if it fails.</p>

## Post Provision in Reconciliation to Save Data

This application synchronizes the Select Identity database after reconciliation. This application handles all updates done during reconciliation, instead of instantiating a workflow instance for each update. Set the following fields for this application:

<b>Fields</b>	<b>Description</b>
<b>Asynchronous Invocation</b>	Do not check this box if you wish to store the return value for this application.
<b>Return Variable Name</b>	In this case, specify a variable name to store the status (a boolean) if you wish to store the status of this operation. It returns true if reconciliation provisioning succeeds, and false if it fails.

## Post Provision in Bulk Processing to Save Data

This application synchronizes the Select Identity database after bulk processing. This application handles all updates done during bulk processing. Set the following fields for this application:

<b>Fields</b>	<b>Description</b>
<b>Asynchronous Invocation</b>	Do not check this box if you wish to store the return value for this application.
<b>Return Variable Name</b>	In this case, specify a variable name to store the status (a boolean) if you wish to store the status of this operation. It returns true if bulk provisioning succeeds, and false if it fails.

## External Call

This application makes a call to an external system during workflow approval. The external call must be developed as described in the *HP OpenView Select Identity External Call Developer Guide*, which is available on the Select Identity CD in the `docs/api_help/external_calls` directory. The external call must be registered in Select Identity before you can use this application.



If you need access to the user attributes or the request object in a workflow, you must create an external call. The RequestTarget class enables you to access this information; see the *HP Openview Select Identity External Call Developer Guide* for more information.

Set the following fields for this application:

Fields	Description
<b>Asynchronous Invocation</b>	Do not check this box if you wish to store the return value for this application.
<b>Return Variable Name</b>	In this case, specify a variable name to store the status (a boolean) if you wish to store the status of this operation. It returns true if the external call is successful and false if it fails.
<b>External Call Name</b>	The name of the external call. You can specify name of an Approver Selection or Workflow external call as registered on the External Calls page of the Select Identity client. This name can be a variable expression (name, constant string or combination), if it is a constant string. You must quote this name. Example: checkDB

## Approvers External Call

This application queries an external system for a list of users who can approve provisioning requests during a workflow. The external call must be developed as described in the *HP OpenView Select Identity External Call Developer Guide*, which is available on the Select Identity CD in the `docs/api_help/external_calls` directory. The external call must be registered in Select Identity before you can use this application.

Set the following fields for this application:

<b>Fields</b>	<b>Description</b>
<b>Asynchronous Invocation</b>	Do not check this box if you wish to store the return value for this application.
<b>Return Variable Name</b>	In this case, specify a variable name to store the status (a boolean) if you wish to store the status of this operation. It returns 1 if the external call is successful and 2 if it fails.
<b>External Call Name</b>	The name of the external call. You can specify name of an Approver Selection or Workflow external call as registered on the External Calls page of the Select Identity client. You must quote this name. Example: "checkDB"
<b>Default Approver</b>	A list of Select Identity user names (approvers); if more than one name is specified, separate them with commas. If no approvers are found in the external source, this list of approvers is returned.



## Add a Set of Users to a List of Services

This application adds the specified list of users to the Services provided in the map. Set the fields properties for this application:

<b>Fields</b>	<b>Description</b>
<b>Asynchronous Invocation</b>	Do not check this box if you wish to store the return value for this application.
<b>Return Variable Name</b>	In this case, specify a variable name to store the status (a boolean) if you wish to store the status of this operation. It returns 1 if the external call is successful and 2 if it fails.
<b>External Call Name</b>	The name of the external call. You can specify name of an Approver Selection or Workflow external call as registered on the External Calls page of the Select Identity client. You must quote this name. Example: "checkDB"
<b>Default Approver</b>	A list of Select Identity user names (approvers); if more than one name is specified, separate them with commas. If no approvers are found in the external source, this list of approvers is returned.

## Set Variable

This action assigns a value to a workflow variable. If the variable does not exist, this creates the variable and initializes its type.

The following parameters are provided for this action:

<b>Parameter</b>	<b>Description</b>
<b>Target Variable Name</b>	The name of the variable. Do not quote this name.
<b>Source Variable</b>	The value to assign to the target variable. Specify an expression such as a string constant, variable, or expression. Example: "Mark" + " " + IName

## Log Message

This action logs information to the console window or to a log file, as defined by the logging configuration for the Select Identity server. Log action can also be used to display log messages on the Request Status page as well as on the console. An example can be found in the LogException activity in the SIOneStageApproval template. Refer to the *HP OpenView Select Identity Installation Guide* for configuration details.

For this action, you must set the **Message Variable** property. Specify the value to be logged; you can specify a variable, string, or expression.

Example: `"Current Name =" + nameVar`

## Throw Exception

This action launches a fault handler. For this action, you must set the **Exception Message Variable** property. You can specify a string, variable, or expression. Exceptions are handled by the catchException activity (see “Fault handler” in [Activities and Blocks](#) for more information).

Example: `"Error:" + fullName + " not found"`

## Run Script

This action, for advanced use only, enables you to run a script, such as to manipulate complex business logic for testing purposes. The script must be written in Beanshell (<http://www.beanshell.org>).

For this action, you must set the **Script Variable** property, which specifies the name of a variable that contains the script code. All the workflow variables can be referenced in this action. All the variables created by this action become workflow variables, which can be referenced anywhere in the workflow.

For example, after you assign `var1="this is a test"` from the Set variable action, you can assign this again in the Run Script action like this:

```
var2 = var1 + " completed successfully"
```

The `var2` variable becomes a normal workflow variable, which can then be referenced in the workflow.

## Call Subworkflow

This action calls another workflow template.



A subworkflow instance cannot be displayed on the **Request Status** page. Workflow templates that are executed as subworkflows are not associated with Service Views. Therefore, the subworkflow cannot be used to implement approval stages.

The following parameters are provided for this action:

<b>Parameter</b>	<b>Description</b>
<b>Template ID Variable</b>	The subworkflow's template name.
<b>Asynchronous</b>	The invocation mode. For synchronous invocation, the workflow is blocked until one of the following occurs: <ul style="list-style-type: none"><li>• The subworkflow activity ends.</li><li>• The subworkflow reaches the wait activity.</li><li>• The main workflow launches a fault handler.</li></ul>
<b>Activity ID Variable</b>	The name of the activity where the subworkflow should begin.
<b>Child Callback</b>	If selected, the subworkflow will resume the parent workflow when the subworkflow completes. This is typically used when the subworkflow contains a wait activity and this action is invoked from a wait activity in a parent workflow. The callback occurs when the subworkflow is reactivated and runs through the end. The callback from the subworkflow will then reactivate the parent workflow to model a call and return scenario in the workflow with wait activities.

## Add Item to List

This action enables you to create a list for passing multiple variables in a single workflow list variable. (A list is a collection of variables.) You must call this action for each variable you wish to add to the list.

<b>Parameter</b>	<b>Description</b>
<b>New Collection</b>	If you are creating a new list, check this box.
<b>List Variable Name</b>	The name of the variable to create or to which you are adding a variable. (Do not quote this name.)
<b>Element Value Variable</b>	The value (expression) of the list element. (Quote the value if this is a constant string.)

## Add Item to Map

This action enables you to create a map for passing multiple variables in a single workflow map variable. (A map is a collection of name-value pairs where the value is a workflow variable.) You must create an Add Item to Map action for each variable you need to set in the map.

<b>Parameter</b>	<b>Description</b>
<b>New Collection</b>	If creating a new map, check this box.
<b>Map Variable Name</b>	The name of the map to create or to which to add a variable. (Do not quote this name)
<b>Element Name Variable</b>	The name of the variable to set.
<b>Element Value Variable</b>	The value of the variable. (Quote the value if this is a constant string)



These User-Defined Variables added to the map can be later referenced in email templates. To access the substituted values of the variables in an email template, reference the variable in the form of [USERDEF:Element Name Variable]. For a complete list of predefined User-Defined variables, see the “Notification Variables” section listed in the *HP OpenView Select Identity Administrator Guide*.

## Send Email

This action sends email to anyone without using a notification template. For example, use this action to notify an administrator if an exception occurs.

Parameter	Description
<b>To Variable</b>	The email address where the workflow will send the message. You can specify a string, which must be quoted, or the name of another variable.
<b>From Variable</b>	The sender’s email address. You can specify a string, which must be quoted, or the name of another variable.
<b>CC Variable</b>	The email address of anyone you wish to copy on the message. You can specify a string, which must be quoted, or the name of another variable.
<b>Subject Variable</b>	The subject of the email. You can specify a string, which must be quoted, or the name of another variable.
<b>Content Variable</b>	The content of the email. You can specify a string, which must be quoted, or the name of another variable.

## Recover From Last Error

If a fault handler is launched, this action can be used to direct the workflow back to the activity where the exception occurred. Using this action implies that you are including logic to fix the problem that caused the exception.

The Recover from Last Error action is typically used with the `catchException` activity. The `catchException` activity starts when an exception is received in the course of a workflow instance execution. The `catchException` can then be followed by a wait activity that forwards an alert message to administrator who can take action to fix the error. The alert message contains an Exception ID that points to the latest exception object, `com.trulogical.truaccess.wfengine.data.WfManagedException`, stored in the `$_exceptionList` workflow variable. The ID is simply a list index that is used to identify the current exception object in the `$_exceptionList` variable.

After the error is fixed, the administrator can call back to the workflow with a null value to resume the workflow. After the workflow is reactivated, it executes the Recover from Last Error action with the ID transition to an activity that caused the exception.

## XPath

This action enables you to search for a string in an XML variable. You can set a variable containing multiple values.

Parameter	Description
XML Variable Name	The name of the variable that contains the XML string to be searched.
XPath Variable	The search path in xpath format.
Result Variable Name	The variable to contain the returned data.
Results List	check this box to return a complete list of the strings found.

## Set Block Variable

This action enables you to set a block variable in the current block where the action is invoked. If this is a single block activity, the call will set the variable to the current block. If this is a block consisting of start, end and wait activities, the call must set the variable before the wait activity (such as start block activity) if the set variable is intended to be read in the wait activity.

Parameter	Description
<b>Block Variable Name</b>	Name of the block variable.
<b>Block Variable Expression</b>	The value to assign to the target block variable. Specify a string constant, variable, or expression. Example: "Mark" + " " + IName

## Get Block Variable

This action enables you to copy a block variable to a workflow instance and execute the block variable.

Parameter	Description
<b>Block ID</b>	Property of the block variable.
<b>Block Variable Name</b>	Name of the block variable.
<b>Target Variable Name</b>	Workflow variable name.

## Transitions

Transitions enable you to connect activities and provide a path for progressing through the workflow. You can also define that certain conditions are met before the workflow progresses from one activity to the next. The Workflow Studio enables you to create transitions as follows:

- You can create a simple transition to connect one activity to the next.

- You can create multiple outgoing transitions. This is called a **split** transition. To indicate how outgoing transitions behave, specify the transition type in the activity that splits into multiple transitions.
  - Select the XOR split type for one of the outgoing transitions only. When any one of the transitions' conditions are met, the activity transitions to the next activity and all other transitions are ignored. If none of the conditions are satisfied, the transition that does not specify a condition performs the transaction.
  - ▶ In XOR transitions, do not specify conditions for all the transitions. Always have one transition without a condition to ensure that the workflow can continue if none of the conditions meets the condition logic.
  - Use the AND split type to transition from one activity to others. If a condition is specified in the transition, the transition occurs only when the condition is evaluated to true. Transitions that do not specify a conditions are always used.
- You can create multiple incoming transitions. This is called a **join** transition. As with split transitions, if you define conditions for each incoming transition, you can set a property on the activity that defines whether all conditions must be met (AND) or only one (XOR).

Set the join type in an end block activity. The join type must be paired with the previous split type. For an XOR join transition, the end block activity transitions when one of the incoming transitions is received. The XOR join transition is the default type.

For an AND join transition, if the Join Count property in the end block is not defined, the end block activity will transition to the next activity when all of the incoming transitions in the template are received. If Join Count is specified, the end block activity transitions when the number of transitions received is equal to Join Count.

- A transition can use an exception expression defined by the engine to indicate that the transition occurs only when an exception is launched by the incoming activity.



---

# 5 Reporting

The Select Identity client enables you to view the status of workflow instances. Once a workflow instance starts, you can view its status, whether it is running or complete. You can view the status of an entire instance or the blocks in a workflow.

The format of the status report is defined in a report template. The report template describes what information is displayed, how it is organized, and the display format, and a report template can be shared by many workflow templates. By default, reports for all workflow instances use this default template.

This chapter describes how to view a report for a workflow instance. It also describes the format of the report templates and how to create a custom report template for use by your workflow templates.

This chapter covers the following:

- [Viewing Workflow Status](#)
- [Report Templates](#)

## Viewing Workflow Status

Perform the following steps to view the status of a workflow:

- 1 Select the **Requests** → **Request Status List** menu option. The **Request Status List** opens.

**Figure 3 Request Status List**

The screenshot shows a web application interface for managing requests. At the top, there is a navigation bar with menus for 'My Identity', 'Requests', 'User Management', 'Service Studio', 'Reports', 'Tools', and 'Help'. Below this, a breadcrumb trail shows 'Home > Request Status'. On the left side, there is a 'Search' panel with several input fields: 'Request ID' (a dropdown menu), 'Begins With' (a text input with a dropdown arrow), 'Period:' with 'From:' and 'To:' text inputs and calendar icons, '(or)' text, 'Select:' with a dropdown menu set to 'Last 5 Days', and 'Detailed Status:' with a dropdown menu set to 'All'. There are 'Search' and 'Reset' buttons at the bottom of the search panel. The main area is titled 'Request Status List' and contains a table of request records. Above the table, there is a message 'Select a request record to view, then click on the appropriate action button.' and a pagination control showing 'Results per page: 10' and 'Displaying: Page 5 of 53 (Items 1 - 530)'. The table has columns for 'Request ID', 'Target User', 'Requestor User ID', 'Started', and 'Status (Time Elapsed/Ended)'. The records are as follows:

Request ID	Target User	Requestor User ID	Started	Status (Time Elapsed/Ended)
<input type="checkbox"/> 3276	ex2_jen (22EX2 - ajro55@hp.com)	sis	Jan 20, 2006 12:02 AM	Completed - Success (Jan 20, 2006 12:02 AM)
<input type="checkbox"/> 3274	Connors, Carol (WSu53093 - carol.connors@hp.com)	sis	Jan 19, 2006 10:01 PM	Completed - Success (Jan 19, 2006 10:01 PM)
<input type="checkbox"/> 3272	Bray, Bobbie (WSu53092 - carol.connors@hp.com)	sis	Jan 19, 2006 9:55 PM	Completed - Success (Jan 19, 2006 9:55 PM)
<input type="checkbox"/> 3270	Ellis, Eugene (WSu53095 - carol.connors@hp.com)	sis	Jan 19, 2006 9:55 PM	Completed - Success (Jan 19, 2006 9:55 PM)
<input type="checkbox"/> 3268	li,Pete (MyUser6 - pete.li@hp.com)	raja	Jan 19, 2006 5:36 PM	Terminated (Jan 19, 2006 5:52 PM)
<input checked="" type="checkbox"/> 3260	a, f (hw2 -	sis	Jan 20, 2006 1:14 PM	In Process (24 MM)

At the bottom of the table, there are three buttons: 'Terminate', 'View Request Status', and 'Retry Request'.

2 If you need to narrow the list of requests displayed, use the search options in the **Search** panel.



Change the number of items per page that you can view by selecting the appropriate number from the **Results Per Page** list.

3 Check the box beside the status record you want to view.

4 Click **View Request Status**.

The **Workflow Detail: [RequestID]** page opens.

**Figure 4 Workflow Detail: [Request ID] Page**

Workflow Detail: 3260 ?

Click on any Workflow block to see the request status details at the selected transition.

Request Details						
ID	Target User ID	Requestor ID	Service Name	Type	Action	Status
3261	hw2	sis	hwEsc1	Delegated Request	Add New User	In Process

[Refresh](#)

Workflow Instance : 2323

**Instance**

Name: Select Identity Instance Request Report

Start time: Jan 20, 2006 1:15 PM

Wait time: 0 days 0 hrs 25 mins

End time: N/A

Status: Wait

Remind List

Sequence	Time	Block Id
1	Jan 20, 2006 1:16 PM	Approval1
1	Jan 20, 2006 1:22 PM	Approval2

[Refresh](#) [Close](#)

5 Refresh the workflow displayed if necessary by clicking on the **Refresh** button.

The colors used in the diagram indicate the status of each activity and transition:

- Green activities are complete; they have been executed and have transitioned.
- Green transitions are complete and the workflow has transitioned past it.
- Yellow activities are in-progress activities (either waiting or being executed) in a block.

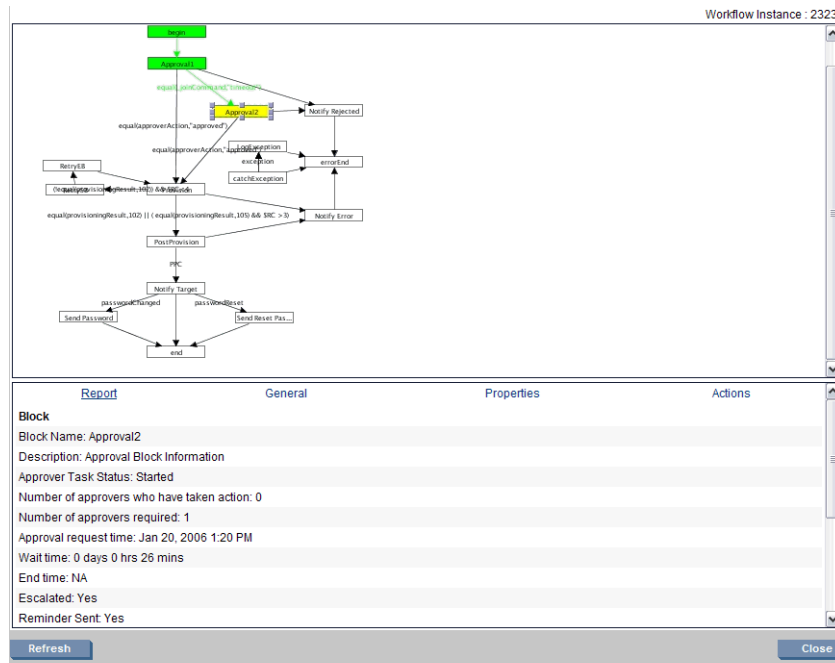
- White activities and black transitions have not been executed.

The tables below the diagram on the Workflow Detail page (see [Figure 4](#)) provide details about the entire workflow instance and its blocks. The content of these tables is determined by the default report template provided by Select Identity.

- 6 Click on any block in the workflow to view details about the block status for this request in the bottom panel.

[Figure 5](#) shows an example of a workflow block report of the selected block.

**Figure 5 Workflow Block Report**



- 7 Click the background of the template image to view the entire workflow instance status.

[Figure 6](#) shows an example of some of the workflow instance status showing information for the workflow in the figure above.

**Figure 6 Workflow Instance Status Examples**

<b>Instance</b>			
Name: Select Identity Instance Request Report			
Start time: Jan 20, 2006 1:15 PM			
Wait time: 0 days 0 hrs 35 mins			
End time: N/A			
Status: Wait			
<b>Remind List</b>			
Sequence	Time	Block Id	
1	Jan 20, 2006 1:16 PM	Approval1	
1	Jan 20, 2006 1:22 PM	Approval2	
<b>Escalation List</b>			
Sequence	Escalated To	Action Date	
1	weim@hp.com	Jan 20, 2006 1:18 PM	
1	weih@hp.com	Jan 20, 2006 1:24 PM	
2	weih@hp.com	Jan 20, 2006 1:27 PM	
<b>Log List</b>			
Activity	Message	Time	
Approval2	start	01-20-2006 01:20	
Approval2	get role [approvalid is 1105, username is thamis, firstName is Ted, lastName is Harris, email is sheryl.horn@hp.com, guid is 7E37A6BD-5F1F-8594-D36E-A6EEFF1D5AF, company is wc, department is , approvalid is 2244, username is KHW, firstName is Kirk, lastName is Husby, email is kirk.husby@hp.com, guid is 29281741-98E7-2788-A4EC-E453815D6849, company is HP, department is , approvalid is 2348, username is KHW2, firstName is Kirk, lastName is Husby, email is kirk.husby@hp.com, guid is 1814C9FC-3684-FEC3-2340-808C3C1D050E, company is HP, department is , approvalid is 1105, username is kth, firstName is Kirk, lastName is Husby, email is kirk.husby@hp.com, guid is 818CE10D-D938-6829-2011-2DC8B48B3EE, company is HP, department is , approvalid is 2245, username is KHW3, firstName is Kirk, lastName is Husby, email is kirk.husby@hp.com, guid is 617B1456-D446-ACB8-7D2F-8EEF319C091B, company is HP, department is , approvalid is 2, username is sisa, firstName is SelschIdentify, lastName is SysAdmin, email is selschidentify@hp.com, guid is GUID, company is , department is ]	01-20-2006 01:20	
Approval2	after create task	01-20-2006 01:20	
<b>Block</b>			
Block Name: Approval1			
Description: Approval Block Information			
Approver Task Status: Completed			
Number of approvers who have taken action: 0			
Number of approvers required: 1			
Approval request time: Jan 20, 2006 1:15 PM			
Wait time: 0 days 0 hrs 5 mins			
End time: Jan 20, 2006 1:20 PM			
Escalated: Yes			
Reminder Sent: Yes			
Alert Sent: Yes			
<b>Possible Approvers</b>			
User Name	Email	First Name	Last Name
dadm1	devi.krishnaswamy@hp.com	A	Abrams
kadmu1	devi.krishnaswamy@hp.com	A	Abrams
dkAd1	devi.krishnaswamy@hp.com	A	Abrams
jadm	nvo@hp.com	jen	admin
fn dao	dat.nguyen@hp.com	tung	dao
dadm4	devi.krishnaswamy@hp.com	D	DDDDD
dkAd4	devi.krishnaswamy@hp.com	D	DDDDD
kadmu4	devi.krishnaswamy@hp.com	D	DDDDD
MyUser6	raja.goli@hp.com	raja	goli
raja	raja.goli@hp.com	Raja	Goli
ra_test	raja.goli@hp.com	Raja	Goli
thamis	sheryl.horn@hp.com	Ted	Harris
kth3	kirk.husby@hp.com	Kirk	Husby
kth2	kirk.husby@hp.com	Kirk	Husby
kth	kirk.husby@hp.com	Kirk	Husby
slka	sri.katanguri@hp.com	Sri	Katanguri

8 Click **Close** when you are finished viewing the workflow instance details.

# Report Templates

A report template is associated with a workflow template using the `ReportTemplateId` global attribute, specified on the **Workflow Information** tab in Workflow Studio. Any workflow template whose report type matches the report template name uses the matching report template to render the status report. If the `ReportTemplateId` attribute is not defined in a workflow template, the default report template provided by Select Identity is used for that workflow template.

In addition to workflow-level reporting, the report template also describes how block status is reported. The workflow template uses the `Block Type` property to identify how a block is rendered. The block is rendered in a format given in the matching block section (identified by `Type` attribute in the `<Block>` element) of the associated report template.

In the default report template, six block types are defined that enable you to view block status:

- `emailVerify`
- `approval`
- `provisioning`
- `postprovisioning`
- `ReconciliationPostProvision`
- `UserServiceProvisioning`

If you assign one of these block types to the `Block Type` property in your workflow template, the report engine will render the information for the block according to the format defined in the default report template.

The following sections describe the format and contents of the report template file and provide steps for creating a new template based on the default.

## Template Structure

A report template is organized hierarchically. At the highest level, there is one workflow instance and many blocks. Here is an overview of the XML file:

```

<Page>
  <Instance>
    ...
  </Instance>
  <Blocks>
    <Block>
      ...
    </Block>
  </Blocks>
</Page>

```

The `<Instance>` tag defines properties for rendering workflow instance-level information. Each `<Block>` tag defines how to render the named block in the workflow instance. The following sections describe the properties defining how the instances and blocks are rendered in the report.

## Instance-level Reporting

The `<Instance>` tag in the template defines how Select Identity renders workflow instance-level data. Here is an example of the `<Instance>` tag from the default report template:

```

<Instance>
  <Row>
    <Text Name="Name" Value="Select Identity Instance Request
Report" />
  </Row>
  <Row>
    <Text Name="Start time" Value="StartTime" Scope="Eng" />
    <Text Name="Wait time" Value="WaitTime" Scope="Eng" />
    <Text Name="End time" Value="EndTime" Scope="Eng" />
  </Row>
  <Row>
    <Text Name="Status" Value="Status" Scope="Eng" />
  </Row>
  <Row>
    <Table Name="Remind List" Variable="remindList"
VariableType="List"
IgnoreNull="true">
      <Column ColLabel="Sequence" ColId="Attempts" />
      <Column ColLabel="Time" ColId="RemindingTime" />
    </Table>
  </Row>

```

```

        <Column ColLabel="Block Id" ColId="Block Id" />
    </Table>
</Row>
<Row>
    <Table Name="Escalation List" Variable="escalationList"
        VariableType="List" IgnoreNull="true">
        <Column ColLabel="Sequence" ColId="Attempts" />
        <Column ColLabel="Escalated To" ColId="EscalateTo" />
        <Column ColLabel="Action Date" ColId="ActionDate" />
    </Table>
</Row>
<Row>
    <Table Name="Exception List" Variable="exceptionInfoList"
        VariableType="List" IgnoreNull="true">
        <Column ColLabel="ID" ColId="Key" />
        <Column ColLabel="Activity" ColId="Activity" />
        <Column ColLabel="Exception" ColId="ExceptionClassName" />
        <Column ColLabel="Message" ColId="Message" />
        <Column ColLabel="Time" ColId="Time" />
    </Table>
</Row>
</Instance>

```

Figure 7 shows an example of instance-level data displayed in a report according to these settings in the default report template:

### Figure 7 Instance-Level Data Example

Instance		
Name: Select Identity Instance Request Report		
Start time: 07-26-2004 05:11		
Wait time: 0 days 0 hrs 1 mins		
End time: 07-26-2004 05:13		
Status: Workflow instance ends		
Remind List		
Sequence	Time	Block Id
1	2004-07-26 17:12:53.0	ApprovalBlock
Escalation List		
Sequence	Escalated To	Action Date
1	selectidentity@hp.com	???



The `<Instance>` tag in the report template defines five rows of text, as specified by the `<Text>` elements.



Value attributes are variables and are substituted by actual runtime data.

```
<Row>
  <Text Name="Name" Value="Select Identity Instance Request
Report" />
</Row>

<Row>
  <Text Name="Start time" Value="StartTime" Scope="Eng" />
  <Text Name="Wait time" Value="WaitTime" Scope="Eng" />
  <Text Name="End time" Value="EndTime" Scope="Eng" />
</Row>

<Row>
  <Text Name="Status" Value="Status" Scope="Eng" />
</Row>
```

The first row is named **Name** and its value is **Select Identity Instance Request Report**. You can see this text in the first row of the report and in the first `<Text>` element in the XML.

The report template also defines three tables — Remind List, Escalation List, and Exception List. For each table defined in the XML, columns are defined. For instance, the Remind List table has three columns — Sequence, Time, and Block ID:

```
<Table Name="Remind List" Variable="remindList"
VariableType="List"
IgnoreNull="true">
  <Column ColLabel="Sequence" ColId="Attempts" />
  <Column ColLabel="Time" ColId="RemindingTime" />
  <Column ColLabel="Block Id" ColId="Block Id" />
</Table>
```

You can see these columns in the Remind List table in the report shown above.



Only two tables are displayed in this report. No exceptions occurred in the workflow instance so no Exception List table is displayed.

See [<Text> Element](#) on page 115 and [<Table>](#) on page 118 for details about configured the data to be displayed in reports.

## Block-level Reporting

A report template can include multiple <Block> elements in the <Blocks> tag to indicate how to render data for blocks in the workflow. The Type attribute defined in the <Block> element identifies the block type. The Block Type property set in the workflow must match this name in order for the report engine to render information according to the definition.

Following is an example of a <Block> element from the default report template. This XML defines how to render information from the provisioning block:

```
<Block Type="provisioning">
  <Row>
    <Text Name="Block Name" Value="endBlockId" Scope="EA" />
    <Text Name="Description" Value="Provisioning Block
Information" />
  </Row>
  <Row>
    <Text Name="Block Status" Value="Status" Scope="Eng" />
  </Row>
  <Row>
    <Text Name="Provisioning start time" Value="StartTime"
Scope="Eng" />
    <Text Name="Wait time" Value="WaitTime" Scope="Eng" />
    <Text Name="Provision end time" Value="EndTime" Scope="Eng" /
>
  </Row>
  <Row>
    <Table Name="Provisioning History"
Variable="provisioningStatReport"
VariableType="List">
      <Column ColLabel="Request Id" ColId="RequestId" />
      <Column ColLabel="Resource Name" ColId="ResourceName" />
      <Column ColLabel="Resource Status" ColId="ResourceStatus" />
      <Column ColLabel="Res Last Update Time"
ColId="ResLastUpdateTime" />
      <Column ColLabel="Operation" ColId="OperationId" />
      <Column ColLabel="Operation Arg" ColId="OperationArg" />
      <Column ColLabel="Operation Status"
ColId="OperationStatus" />
    </Table>
  </Row>
</Block>
```

```

        <Column ColLabel="Op Last Update Time"
        ColId="OpLastUpdateTime" />
        <Column ColLabel="Rollback" ColId="ResRbFlag" />
        <Column ColLabel="Details" ColId="Details" />
    </Table>
</Row>
</Block>

```

Figure 8 shows an example of how the Select Identity client renders the provisioning block data in a workflow instance report:

**Figure 8 Provisioning Block Data in a Workflow Instance Report**

**Block**

Block Name: ProvisionBlock  
Description: Provisioning Block Information  
Block Status: Completed  
Provisioning start time: 07-26-2004 05:13  
Wait time: 0 days 0 hrs 0 mins  
Provision end time: 07-26-2004 05:13

**Provisioning History**

Request Id	Resource Name	Resource Status	Res Last Update Time	Operation	Operation Arg	Operation Status	Op Last Update Time	Rollback	Details
4152	LDAP71	SUCCESS	2004-07-26 17:13:48.081	ADD	NBKAAA2	SUCCESS	2004-07-26 17:13:47.425	false	
4152	LDAP71	SUCCESS	2004-07-26 17:13:48.081	LINK	South Texas	SUCCESS	2004-07-26 17:13:47.878	false	

As in the <Instance> element, the <Text> elements define the rows that are displayed in the report and the <Table> elements define the tables. See <Text> Element on page 115 and <Table> on page 118 for a description of the elements and attributes you can set for block-level reporting.

### <Text> Element

The <Text> element defines data to be displayed in a text row in the report. This element must be created within a <Row> element, and you can specify <Text> elements in the <Instance> and <Block> elements. Here are several examples from the default report template:

```

<Instance>
  <Row>
    <Text Name="Name" Value="Select Identity Instance Request Report" />
  </Row>

```

```

<Row>
  <Text Name="Start time" Value="StartTime" Scope="Eng"/>
  <Text Name="Wait time" Value="WaitTime" Scope="Eng"/>
  <Text Name="End time" Value="EndTime" Scope="Eng"/>
</Row>
...
</Instance>
<Blocks>
  <Block Type="emailVerify">
    <Row>
      <Text Name="Email verification Status" Value="Status"
Scope="Eng" />
    </Row>
    <Row>
      <Text Name="Block Name" Value="endBlockId" Scope="EA"/>
      <Text Name="Description" Value="Email Verification Block
Information"
Scope="Var" />
    </Row>
    ...

```

To understand how the report engine renders this element, see the explanation in [Instance-level Reporting](#) on page 111.

Here is a description of the attributes required by the <Text> element:

- **Name** — The text row label. For example, if you specify **Name="Name"**, a row named Name is rendered for instance-level data in the report.
- **Value** — The value assigned to the row. The value assigned to this attribute depends on the value assigned to the Scope attribute, as described below.
- **Scope** — The type of data displayed in this row. The following values are supported for this attribute:
  - **Text** (or if the Scope attribute is omitted from the <Text> element) — Shows static text. For example, if you specify the following element:

```

<Text Name="Name" Value="Select Identity Instance Request
Report" />

```

This text is displayed in the Name row of the report, as shown in the report example in [Instance-level Reporting](#) on page 111.

- **Var** — Shows the value of the specified variable. Variables are created in the workflow instance by the Set Variable action or when a variable name is assigned to the return value of an application invocation.

For example, if you created the `appList` variable in a block to contain the list of approvers returned by the Get approvers by role application invocation, you can specify the following in a `<Block>` element in the report template:

```
<Text Name="Approvers" Value="appList" Scope="Var" />
```

If the `<Text ... Scope="Var">` element is under `<Instance>`, the workflow instance variables are reported. If the `<Text ... Scope="Var">` element is under `<Block>`, the engine reports on variables in the current block first. If the variable does not exist in the current block, it tries to report on the same variable at the workflow instance level.

- **EA** — Extended Attributes. Shows the value of the specified property. Instance-level properties are set on the Workflow Information tab, which is available when you click in the background of a template. Block-level properties must be set in the end activity of the block. For a list of block-level properties that you can use for this attribute, see [System-Defined Properties](#) on page 66.

For example, if you wish to print the name of the block in the report, specify the following element:

```
<Text Name="Block Name" Value="endBlockId" Scope="EA" />
```

- **Eng** — Shows report engine-generated data; you must specify the name of the data. You can specify the following values for this attribute:
  - `StartTime` — The start time of the instance or block
  - `EndTime` — The end time of the instance or block
  - `WaitTime` — The total wait time of the instance or block (how long all wait activities took in the instance or block)
  - `Status` — The status of the instance or block
  - `joinedCount` — A block-level variable that specifies the number of provisioning actions that took place before the block exited

## <Table>

The <Table> tag specifies how to display a list of data objects in a table in the report. Each table displayed in the report template is mapped from a list. An object element in the list represents a row in the table and an attribute (field) in the object represents a column.

Within each <Table> element, <Column> elements define the content of the table. You can configure the report engine to render collection (list) variables created by the workflow engine or those defined by the Add Item to List action (see [Add Item to List](#) on page 100).

Here are several examples from the default report template:

```
<Instance>
...
<Row>
  <Table Name="Exception List" Variable="exceptionInfoList"
    VariableType="List" IgnoreNull="true">
    <Column ColLabel="ID" ColId="Key"/>
    <Column ColLabel="Activity" ColId="Activity"/>
    <Column ColLabel="Exception" ColId="ExceptionClassName"/>
    <Column ColLabel="Message" ColId="Message"/>
    <Column ColLabel="Time" ColId="Time"/>
  </Table>
</Row>
</Instance>

<Blocks>
...
  <Row>
    <Table Name="PostProvisioning Activities"
      Variable="ReconPostProvisionStatReport"
      VariableType="List">
      <Column ColLabel="Request Id" ColId="RequestId"/>
      <Column ColLabel="Resource Name" ColId="ResourceName"/>
      <Column ColLabel="Service Name" ColId="ServiceName"/>
      <Column ColLabel="User Name" ColId="ConceroUserId"/>
      <Column ColLabel="Operation" ColId="RequestType"/>
      <Column ColLabel="Operation Status" ColId="Status"/>
      <Column ColLabel="Message" ColId="Msg"/>
    </Table>
  </Row>
</Blocks>
```

```
</Row>
</Block>
</Blocks>
```

Each field in the object can be displayed as a column in the table.

### Attributes of the <Table> Element

The following describes the attributes of the <Table> element. Refer to the XML listed above for examples.

- **Name** — The title of the table.
- **Variable** — The variable whose values you wish to display in a table. Specify the name of the variable created by an external application or the Add Item to List action, or specify the name of a workflow engine-defined variable. If you wish to render an engine-defined variable, the following values are supported:
  - **remindList** — Lists information about notifications (reminders) sent during the workflow instance or block activities
  - **escalationList** — Lists information about escalations that occurred during the workflow instance or block activities
  - **exceptionInfoList** — Lists information about exceptions that occurred during the workflow instance or block activities
  - **alertList** — Lists information about alerts sent during the workflow instance or block activities
  - **pushList** — Lists approvers who have taken actions in the block; specify this value for tables in <Block> elements only
  - **provisioningStatReport** — Lists the status of each provisioning request; specify this value for tables in <Block> elements only
  - **ReconPostProvisionStatReport** — Lists the status of each reconciliation request; specify this value for tables in <Block> elements only
- **VariableType** — The type of variable. You must specify **List** as the value of this attribute; this is the only type supported.
- **IgnoreNull** — Whether to render the table if no values are available. Set this attribute to **true** if you do not want to render an empty table.

## Attributes of the <Column> Element

For each <Table> element, you must define one or more <Column> elements to configure the data that will be displayed in the table. The attributes of the <Column> element are as follows:

- **ColLabel** — The column name (the label of the column displayed in the report)
- **ColId** — The field name in the java bean object or the key name in the Map object. The ColId attribute corresponds to an attribute name of a data object element in the list variable if `variableType` is List. If you are rendering an engine-defined variable, the following fields are supported. Refer to the default report template for examples:
  - For the remindList variable:
    - **Attempts**
    - **RemindingTime**
    - **Block ID**
  - For the escalationList variable:
    - **Attempts**
    - **EscalateTo**
    - **ActionDate**
    - **ApproverComment**
  - For the exceptionInfoList variable:
    - **Key**
    - **Activity**
    - **ExceptionClassName**
    - **Message**
    - **Time**
  - For the alertList variable:
    - **Attempts**
    - **EscalateTo**
    - **ActionDate**
    - **ApproverComment**



- For the pushList variable:
  - ApproverName
  - Status
  - ActionDate
  - ApproverComment
- For the provisioningStatReport variable:
  - RequestId
  - ResourceName
  - ResourceStatus
  - ResLastUpdateTime
  - OperationId
  - OperationArg
  - OperationStatus
  - OpLastUpdateTime
  - ResRbFlag
  - Details
- For the ReconPostProvisionStatReport variable:
  - RequestId
  - ResourceName
  - ServiceName
  - ConceroUserId
  - RequestType
  - Status
  - Msg

For example, the following XML specifies to render data from the remindList variable:

```
<Row>
  <Table Name="Remind List" Variable="remindList"
  VariableType="List" IgnoreNull="true">
    <Column ColLabel="Sequence" ColId="Attempts"/>
    <Column ColLabel="Time" ColId="RemindingTime"/>
  </Table>
</Row>
```

```

    <Column ColLabel="Block Id" ColId="Block Id" />
  </Table>
</Row>

```

This XML is rendered as follows in the Workflow Instance report:

**Figure 9 Report of XML Data Rendered From remindList Variable**

RemindList		
Sequence	Time	Block Id
1	Jan 20, 2006 1:16 PM	Approval1
1	Jan 20, 2006 1:22 PM	Approval2

## Creating a Custom Report Template

A default report template is provided that describes how the report engine should render workflow data. You can export and modify the default template file from the **Tools** → **Import/Export Configurations** menu option. Or, you can create another report template.

You can use a default report template to create your own, by completing these three steps:

**Step 1: Export the default report template**

**Step 2: Edit the default report XML file**

**Step 3: Import the XML file into Select Identity**

To assign this report template to a workflow template, be sure to create the `ReportTemplateId` property as a global property for the template (on the template's Workflow Information tab, which you can display by clicking in the background of the template). Assign the name of the report template (such as `ReconciliationReport` in this example) as the value.

To render a specific block in the workflow template according to a format specified in `<Block>` in the report template, create the `Block Type` property in the end activity of the block. Assign the name of the `<Block>` element (as defined by the `Type` attribute of the `<Block>` element) as the value of the property. See [Block-level Reporting](#) on [page 114](#) for details.

Perform the following detailed steps to edit the default report template:

## Step 1: Export the default report template

- 1 Select the **Tools** → **Import/Export Configurations** → **Export Configuration** menu option.

The **Export Configuration** page opens.

**Figure 10 Export Configuration Page**

My Identity ▾ Requests ▾ User Management ▾ Service Studio ▾ Reports ▾ Tools ▾ Help ▾

Home > Export Configuration

### Export Configuration

Select the object you want to export, then select the item(s) from the list, Select the Export Configuration button to export the list of items. Save the resulting XML file with a new name from the browser menu.

Configuration Type: Request Instance Report ▾

Request Instance Report Name ▾ Begins with ▾ [ ] Search Reset

Results per page: 20 ▾ Displaying: Page 1 of 1 (Items 1 - 1)

Workflow Report Template Name
<input type="checkbox"/> DefaultReport

Select

Selected Workflow Report Templates
<input type="checkbox"/> DefaultReport

Remove

Export Configuration Cancel

- 2 Select **Request Instance Report** from the **Configuration Type** list.
- 3 Select **DefaultReport** in the **Workflow Report Template Name** panel and click **Select**.

The **DefaultReport** name appears in the selected panel.

➤ You can click **Remove** to delete any of the names in the selected panel.

- 4 Select **DefaultReport** in the **Selected Workflow Report Templates** panel and click the **Export Configuration** button.
- 5 You are prompted to choose one of the following options:
  - **Open** to view the XML data file.
  - **Save** to save the XML data file to your local machine.
  - **Cancel** to return to the Export Configuration page.

- 6 Click **Save** and save the file to any location.
- 7 When the file is downloaded, you are prompted to choose one of the following options:
  - **Open** the file to view it.
  - **Open Folder** to locate the folder in which the file resides.
  - **Close** the dialog box to return to the **Export Configuration** page.
- 8 If you choose **Open** to view the file, the file opens in a new window.
- 9 Close the XML window to return to the **Export Configuration** page.

### Step 2: Edit the default report XML file

- 1 Modify the elements in the <Instance> section to define how workflow instance-level data is to be rendered in the report. See [Instance-level Reporting](#) on [page 111](#) for details about the elements and attributes in this block.
- 2 If you are creating a new template file, you must change the Value attribute in the <ns1:Key> element. For example, if you wish to rename this template to **ReconciliationReport**, change the following:

```
<ns1:Key Value="DefaultReport"></ns1:Key>
```

TO:

```
<ns1:Key Value="ReconciliationReport"></ns1:Key>
```

- 3 Modify the existing <Block>, or add additional <Block> elements, to define parameters for reporting on a specific blocks in your workflow template.
- 4 Save the file with a different, unique name.

### Step 3: Import the XML file into Select Identity

- 1 Select the **Tools** → **Import/Export Configurations** → **Import Configuration** menu option.  
The **Import Configuration** page opens.

**Figure 11 Import Configuration page**

Import Configuration

Select the object you want to import into Select Identity and browse for the associated file name.

Configuration Type: Request Instance Report

File Name: nce Report\_ReconciliationReport.xml Browse...

Import Configuration Cancel

- 2 Select **Request Instance Report** from the **Configuration Type** list.
- 3 Click **Browse** to locate the XML data file you want to import.
- 4 Select **Import Configuration** to import the XML data file.  
A message shows on the **Import Configuration** page stating that the file's data was imported successfully.
- 5 Click **Cancel** to return to the Home page, or select a menu option.

The XML file is loaded into the Select Identity database when you import it. The newly-defined report template is available in the Configurations list.



---

## 6 Scenarios

This chapter provides the following scenarios that illustrate how to use external calls with workflows and create a parallel approval workflow:

- [Enforcing Entitlement Rules](#)
- [Adding Services to a User](#)
- [Parallel Approval Workflow Design](#)

### Enforcing Entitlement Rules

This scenario provides an example of how to use Select Identity to enforce business rules. Using an external call as a step in a workflow, you can enforce entitlement rules. Consider the following scenario:

A change to a user's department comes in through Reconciliation. As a result, Reconciliation runs a workflow, which checks the user's existing department and compares it to the department sent to Reconciliation. Based on the new department value, the old cost center and entitlement are removed from the user and the new values are added. (The resource is Active Directory.)

The following procedure illustrates how to register the external call that will evaluate whether the user's department has changed and assign new values, if necessary. The following is assumed about this scenario:

- Active Directory (AD) is the resource in which users are provisioned for this example, thus it is assumed that the AD connector, resource, and attributes exist.
- The Service exists and the ReconciliationDefaultProcess workflow template is associated with the Service views.

Complete the following Tasks to create the external call and associate it with a workflow template:

- Task 1, Code the external call that enforces the entitlement rules
- Task , The script used for this scenario is provided as an example in the *HP OpenView Select Identity External Call Developer Guide*, in the “Examples” chapter. Register the external call with Select Identity
- Task 2, Edit the default Reconciliation workflow template to call the Entitlement Rules external call.

**Task 1: Code the external call that enforces the entitlement rules**

Create a Java object that performs the task as part of a workflow. See the *HP OpenView Select Identity External Call Developer Guide* for detailed information on creating external calls.

The Java object enables you to integrate approval processes with external processes and systems. You must use the External Call API and Workflow API, which define Java-based interfaces for creating external callouts.

- In the script, the following codes are used:

Department	Entitlement	CostCenter
Sales	SA-505	101
Finance	FIN-505	205
HR	HR-101	308
Corporate	CORP-3	409

The script used for this scenario is provided as an example in the *HP OpenView Select Identity External Call Developer Guide*, in the “Examples” chapter. Register the external call with Select Identity

- 1 Select the **Service Studio** → **External Calls** menu option.  
The **External Call List** opens.
- 2 Click **Register External Call**.  
The **Register External Call** page opens.
- 3 Enter **Entitlement Rules** in the **External Call Name** field.
- 4 Enter **Enforces entitlement rules when a user’s department changes** in the **Description** text box.



- 5 Enter **com.trulogica.truaccess.wfenginesvcs.wfexternalcall.support.WfEntitlementChange** in the **Classname** field.



This is the name of the class that implements the Java interface. The class name shown here is for illustration purposes and may not be valid.


- 6 Select **Workflow External Call** from the **Call Type** list.
- 7 Click **Next**.  
The **Register External Call Details** page opens.

You can add parameters and their values to the external call. In this example, this script does not require input parameters.

- 8 Click **Finish** to register the new external call with Select Identity.

For more information about registering external calls, see the External Calls chapter in the *HP OpenView Select Identity Administrator Guide*.

**Task 2:** Edit the default Reconciliation workflow template to call the Entitlement Rules external call.

- 1 Select the **Service Studio** → **Workflow** menu option.  
The **Workflow Template List** opens.
- 2 Locate and select the **ReconciliationDefaultProcess** template.
- 3 Click **Modify**.  
The Workflow Studio opens and loads the **ReconciliationDefaultProcess** template.
- 4 Modify the **Provision** activity to call the Entitlement Rules external call.  
Click the **Select** icon  on the toolbar and click the **Provisioning** activity.  
The property tabs are displayed on the right.
- 5 Click the Application Invocation in the list, so that the new external call is executed before provisioning.
  - a Select **Application Invocation** from the **Name** list.
  - b Select **External Call** from the **Application Name** list.
  - c Enter “**Entitlement Rules**” in the **External Call Name** field.
  - d Click **Apply** to save the changes.

- 6 Save the template by clicking the **Save** icon  on the toolbar, and close Workflow Studio.

Now, Select Identity can enforce business rules. The external call is registered with Select Identity and the workflow will process the necessary entitlement changes based on changes to users.

For more information about creating workflows, see [Overview of Creating a Workflow Template](#).

## Adding Services to a User

Select Identity provides the LoadUserServices external call, which gives you the ability to add Services to a user based on a context change. Select Identity then provisions the user in the resources for the new Services. You can use this external call when moving a user (using the Select Identity client) or through reconciliation.

This scenario illustrates how you can use a workflow and the external call to automatically move users from one set of Services to another based on a user's company. The following is configured to support this:

- A rule is created that adds new Services to the users based on context. For each possible company, a list of Services is provided in the rule, as follows:

Company	Services
TI	<ul style="list-style-type: none"><li>• HRManagers</li><li>• DevManagers</li><li>• QAManagers</li><li>• AllManagers</li></ul>
HP	<ul style="list-style-type: none"><li>• ITManagers</li><li>• PchManagers</li><li>• FacManagers</li><li>• AllManagers</li></ul>
GE	<ul style="list-style-type: none"><li>• AcctManagers</li><li>• PayrollManagers</li><li>• AllManagers</li></ul>
SWB	<ul style="list-style-type: none"><li>• MedManagers</li><li>• SecManagers</li><li>• DocManagers</li><li>• AllManagers</li></ul>

- An external call is created that reads the rule and determines the Services available to a user based on a new context value. The external call returns the list of Services.
- A workflow template is created that removes the Services no longer valid for the user based on the context change. The workflow template then invokes the external call to retrieve the list of new Services. It evaluates each Service to determine if the user is qualified for assignment (based on required fields, constrained values, and so on). For each valid Service, a new request is created to provision the user on the applicable resource(s).

Complete the following tasks to create the rule, external call, and workflow template to add services to a user:

- [Task 1, Create a rule called Services\\_ReconRule that adds new Services to a user based on context.](#)
- [Task 2, Register the rule with Select Identity](#)

**Task 1:** [Create a rule called Services\\_ReconRule that adds new Services to a user based on context.](#)

In this rule, the context is the user's company. If you wish to review the rule, refer to the `Services_ReconRule.xml` file in the `\SampleXML\Reconciliation\Rules\AddServiceWithExtCall` directory on the HP OpenView Select Identity CD.

**Task 2:** [Register the rule with Select Identity](#)

- 1 Select the **Tools** → **Rules** → **Add Rule** menu option. The **Add New Rule page** opens.
- 2 Click **Browse** to select the **Services\_ReconRule** file.
- 3 Click **OK** to make the file available within Select Identity. You are prompted that the changes you made will be saved.
- 4 Click **OK**. Returns to the **Rule List**.

# Parallel Approval Workflow Design

This scenario provides an overview of how to create a parallel approval workflow. In this scenario, the workflow is called **SI 3 Parallel Approvals**.

This scenario walks through the design and testing of the workflow using high-level instructions in the following sections:

- [Workflow Design](#)
- [Testing the Workflow](#)

## Workflow Design

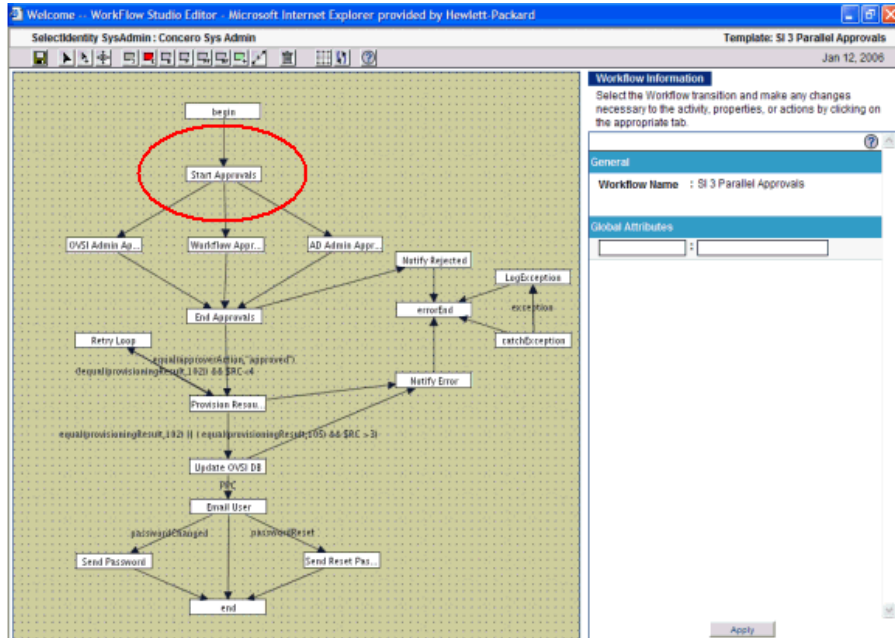
The key to parallel processing of any kind, including approvals in this scenario, is to follow these guidelines:

- Frame the steps with specific **Start** and **End** block activities. Between the Start/End block activities, the administrator can execute any series of workflow steps, including parallel, serial and any combination of these.
- Define the execution paths using the transitions.
- Define the **Split** type for the **Start Block** and the **Join Type** for the **End Block**.

Perform the following high-level steps to design the workflow:

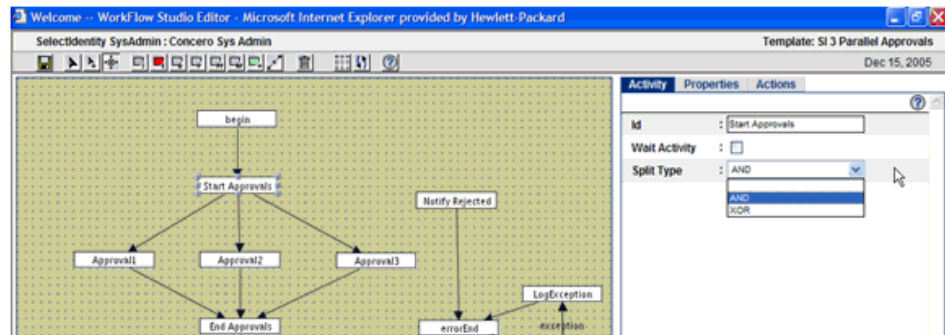
- 1 You must start and end the multi-approval stage with the **Start Block Activity** and **End Block**.

**Figure 12 Start Block Activity**



2 Use the logical AND to force all branches to run.

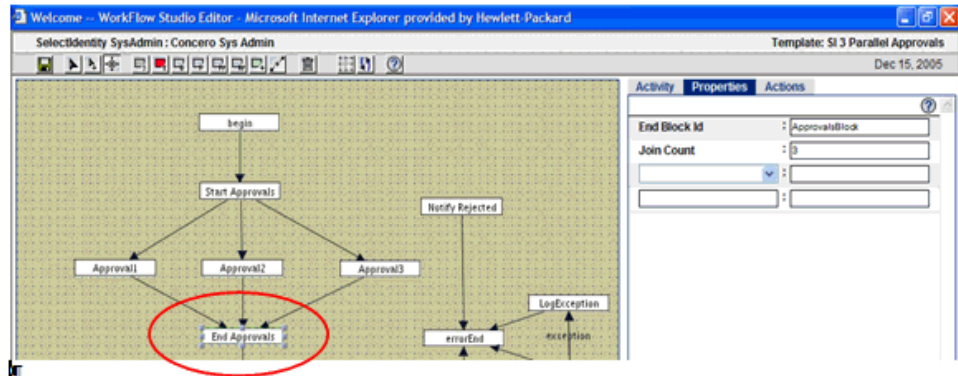
**Figure 13 Using the Logical AND**



➤ The three middle approval blocks can be simple Activity Blocks or Approval Blocks with their own block IDs for Request Status reporting.

- 3 You must use the End Block Activity to conclude the parallel processing.

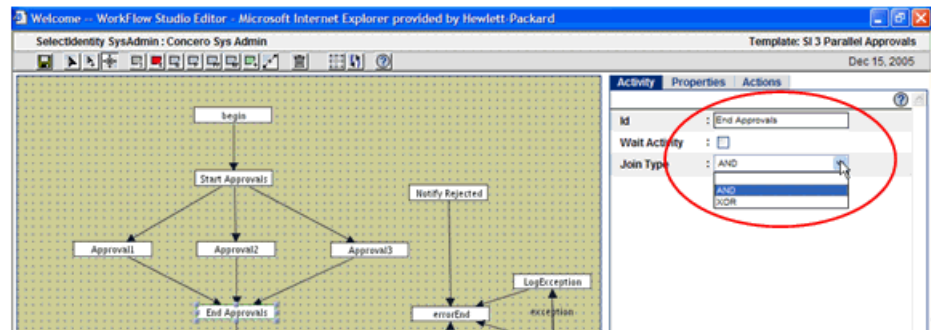
**Figure 14 End Block Activity**



- 4 Use the logical **AND** as the Join Type to force all three approvals to execute.
- 5 Use the **Join Count** property to determine how many approvals must complete before moving on.

The **End Approvals** activity as an end block activity ensures that the workflow does not continue to transition to the provision activity until it receives all three incoming transitions corresponding to three approvals.

**Figure 15 Join Count Property**



Reject action from the approver forces an exit from the approval block regardless of whether joined number of approvers has reached join count or not. Typically, the approval application sets workflow variable `_joinCommand` to exit when it reactivates the workflow so the workflow engine will force the block to exit.

You may want to force block to exit based on the business logic before the joinCount reaches joinedCount. This can happen when \_joinCommand variable is set to exit. You can set this variable in EndBlockActivity or additional activity between wait and EndBlockActivity. You may use Run Script action to evaluate some business logic before setting the variable. For example, when reject action is taken at approval stage, the approval application sets \_joinCommand to exit before it reactivates the workflow instance. \_joinCommand variable is also set to timeout when the block exists due to timeout without waiting for join. The subsequent transition may utilize this as condition to branch.

If the **approval1** activity is a block type, **approval1** will exit when a reject action is taken. To pass this exit variable to the end block activity of End Approvals, the \_joinCommand must be set in this activity as well to force the End Approvals block to exit. This can be done by adding the RunScript action in the End Approvals activity:

Set the following script text in the RunScript action, which is added to the Join activity (End Approvals) of the above workflow:

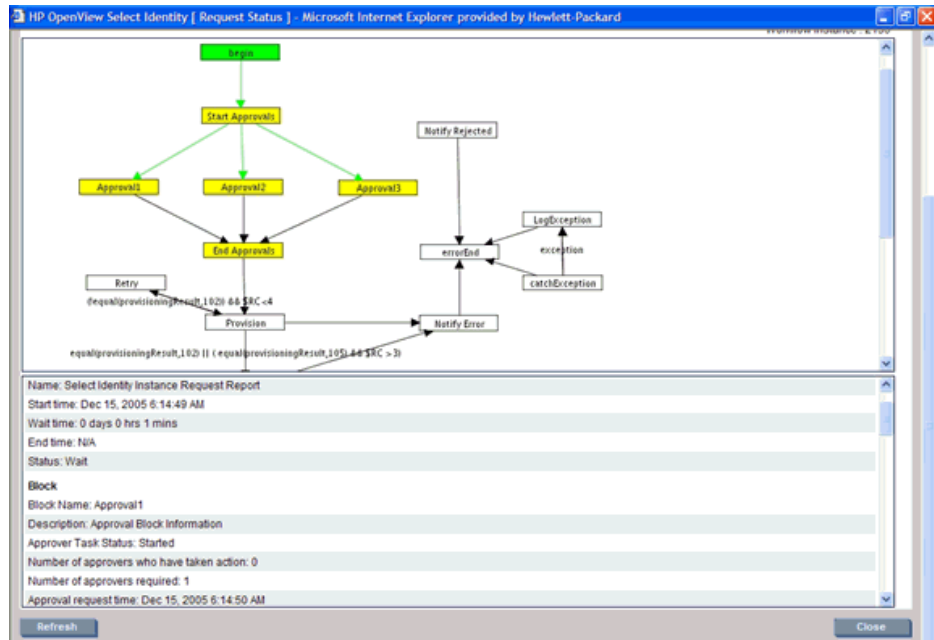
```
If( equal( approvalAction, "reject" ) )
    _joinCommand="exit";
```

## Testing the Workflow

This section shows a registration and subsequent workflow execution. The following steps describe how to test this workflow.

- 1 Select the **Requests** → **Request Status List** menu option to open the **Request Status List**.
- 2 Select the workflow name, **SI 3 Parallel Approvals**, and click **View Request Status**.

**Figure 16 Pending Approval Request Status**

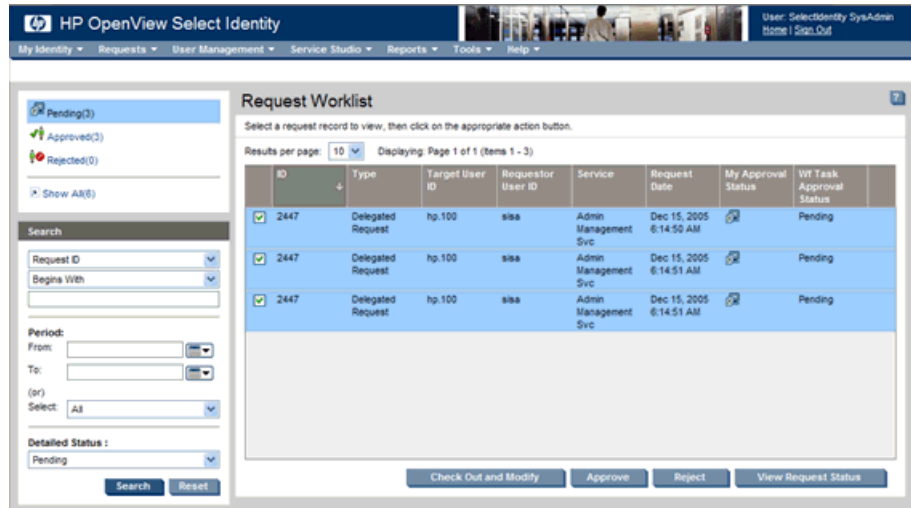


► Notice that all activities from Start to End blocks are yellow because of the pending approvals.

- 3 Go to the **Request Worklist** page (select **Requests** → **Request Worklist**) and select all approval requests to make multi-approval requests.

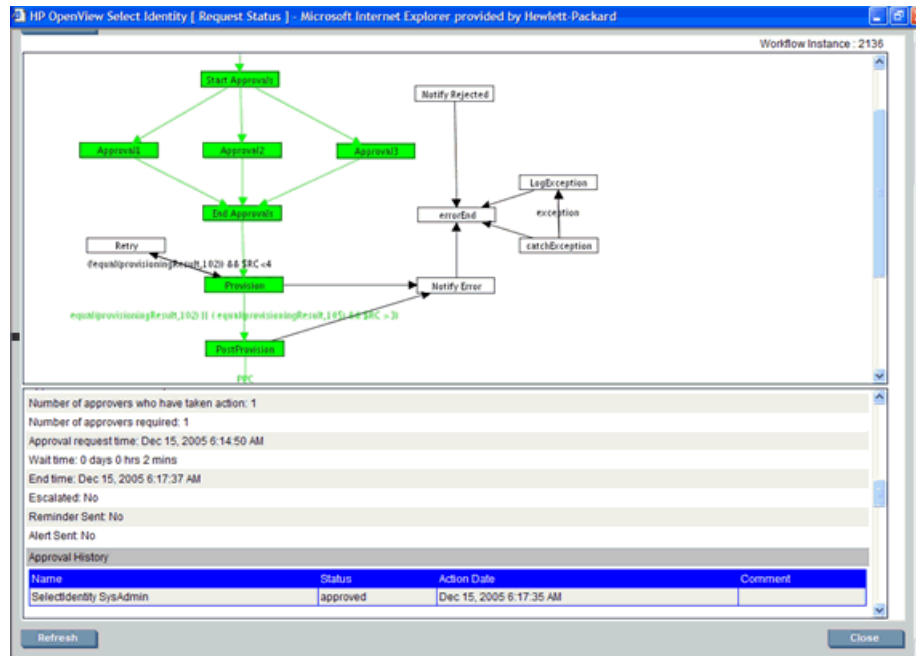


**Figure 17 Making Multi-Approval Requests**



- 4 Click **Approve** to execute the approvals.
- 5 Return to the Request Status List page and select the request if you want to view status.
- 6 Click **View Request Status** to ensure all pending activities were executed. The activities should be green.

**Figure 18 Post Approval Request Status**



---

# A Frequently Asked Questions (FAQ)

## General

What is the process for designing and creating a workflow template?

Follow these guidelines to design and create a workflow template:

- 1 Determine what tasks to create, such as for approval and provisioning.
- 2 Design a block for each task.
- 3 Determine what applications or actions must be invoked.
- 4 Decide how these applications are called (such as their order and the business rules that govern them) to layout activities and place actions.
- 5 List the input parameters required by the applications.
- 6 Determine what workflow variables are needed and how to create and pass them between activities.

What is a wait instance?

When a running workflow instance hits a wait activity, it is passivated until it is reactivated by an external source. The passivated workflow instance is called wait instance.

How is a workflow instance started or reactivated?

A workflow instance can be started or reactivated using the workflow API (see the *HP OpenView Select Identity External Call Developer Guide* for details), subworkflow action invocation, or properties file.

### Are macros defined for use in a workflow expression?

A workflow macro is a function defined by the workflow engine to be used as part of an expression. For example, in a transition condition, you can use the `equal()` macro to compare two objects, as in this example:

```
equal (approved, "approved" )
```

The workflow engine will evaluate it and make a decision based on the returned result. See [Using Variables](#) on page 74 for the list of macros provided for workflow templates.

### Can workflow variables be any java objects set in external call?

A workflow variable can be any Java object. However, if it is a persistent variable with its name preceded by \$, you may either use `HashMap` to contain object attributes or use the Java object marked as `Serializable`. If it is a Java object, you should also add following line in the Java class:

```
static final long serialVersionUID = uid;
```

where `uid` can be 0 for the newly created class or found by running JDK's `server` command.

### Where can I use workflow engine-defined variables?

The workflow engine internally defines a set of variables for advanced usage. These are low-level variables and only advanced users may access these variables. For example, when you register an application, you may need to pass the variables to application. See the *HP OpenView Select Identity External Call Developer Guide* for a list.

## Activities

### What is the purpose of a wait activity?

A wait activity is like normal activity except that it does not automatically transition to the next activity. After all of the actions are executed, this activity suspends the running workflow instance temporarily until it is resumed later by an external application, such as the Workflow API.

Typically, the `_instActivityId` variable that is defined by the workflow engine is passed to the external application, which uses this identifier to reactivate the workflow later. When the instance is reactivated, it resumes at a point where wait activity completes.

[Does the `catchException` activity catch any exceptions launched by any application?](#)

Yes. Any activities that receive errors stop executing that activity and the workflow transitions to the activity level fault handler activity, if it is defined for this fault activity.

If the fault handler is not defined for the activity, the activity transitions to the block level fault handler activity if it is defined for a block which contains fault activity.

If the fault handler is not defined for the block, the block transitions to the instance level fault handler activity.

[When should an exception expression be used for transition's condition?](#)

The `catchException` activity is used to globally catch an exception, while a transition with an exception condition enables the workflow to catch an exception at the activity level. See one of the default templates for an example. In the default templates, the `catchException` can catch an exception launched by an activity. It then transitions to the `Notify Error` activity. If this activity launches another exception, an infinite loop will occur. To avoid this, use a transition with an exception condition as shown in the template.

## Blocks

[How can I create a block in a workflow template?](#)

A block can be represented by a single block activity, or a start block activity, an end block activity, and other activities between these two activities. Start and end block activities are identified by the same ID, which is assigned to the `Start block ID` and `End Block ID` properties. For the single block activity, a block is identified by the `blockId` property. To define block properties such as `timeout`, `joincount`, `escalation`, and so on, a block *must* be used to handle these special properties.

### In what circumstances do I need to represent a block by start and end block activities?

Typically, you use a single block activity to represent a block, which is the simplest way to create a block in a workflow. You use blocks with start and end block activities when a block contains more activities. For example, when multiple concurrent approvers with different roles may require more than one approval activity in a block. Service level agreement (SLA) is another example in which you can use start and end block activities to surround a group of activities (or even all the activities between the begin and end activities) in the workflow to form a block. By defining a `timeoutAfter` block property, SLA is activated at the block level.

### How do I define block properties?

Block properties are defined in a single block activity, or the end block activity.

### How can I access block variables?

Block variables are accessed using Workflow API calls, or Get/Set Block Variable actions. A block variable is identified by name and block ID. A block variable can also be referenced in a report template to render the block information in the resulting report.

### What does the Join Count property do in a single block activity or an end block activity?

The workflow engine uses the Join Count property defined in an end block activity to determine whether the end activity (or block activity) can exit. Internally, it stores a variable that counts the number of times an end block activity is executed. If the Join Count value matches the joined count, the end activity transitions out of the block. If Join Count is not specified, the incoming transition count in the template for the end block is implicitly assumed to be the join count.

### How do I make activities run concurrently?

Activities that are split by a start block activity with the AND join type run concurrently.

### How do I synchronize concurrently running activities?

Activities that join at an end block activity with AND join type transitions are synchronized. The end block activity does not exit until all transitions are received.

### Is an end block activity a waiting activity?

An end block activity cannot be specified as a wait activity. However, an end block activity can suspend the instance if the total number of anticipated transitions are not received.



A single block activity can be a wait activity.

### Can a block time out?

A block can time out if it does not complete the block activities within the specified timeout property value. This timeout value is defined in the end block activity by the Timeout After property. You can also set the Timeout At property or a timeoutTime block variable property to time out at a specific time.

## Actions

### How does a parent workflow exchange variables with a subworkflow?

All persistent variables in the parent workflow are automatically passed to the subworkflow and all persistent variables from the subworkflow are passed back to the parent workflow when subworkflow returns. Engine-defined variables are not shared between workflow instances.

### How does the workflow behave when a subworkflow contains wait activities?

When a subworkflow is invoked synchronously, the parent workflow instance will not continue until the subworkflow returns. For a subworkflow with wait activities, the Call Subworkflow action should be invoked in a wait activity or by selecting the subworkflow wait check box for the activity. The Child Callback check box should be selected so that the waiting parent can be reactivated when the child subworkflow ends.

### Does a subworkflow run in the same instance as the parent workflow?

No, it runs in a separate instance. In the current version, the subworkflow instance status cannot be viewed in the Request Status window. Also, since the subworkflow cannot be associated with a service view in the current version, it cannot be used for an approval process.

### Does the Throw Exception action terminate a workflow instance?

If a catchException activity or other levels of the fault handler are implemented, the instance will start at the fault handler activity right after the exception is launched. Otherwise, the workflow instance propagates the exception back to the workflow client (where the workflow was invoked) and terminates itself.

### Should I implement a fault handler activity when the Throw Exception action is used?

Fault handler is optional. If it is not used, the launched exception will be propagated to the workflow invoker (such as the Select Identity server).

### How should the Recovery From Last Error action be used?

This action is typically used with a catchException activity. The catchException activity is usually followed by a wait activity that forwards an alert message containing the exception context to administrators who can fix the error. Once the error is fixed, the administrator can callback to the workflow using the exception context to resume the workflow which, in turn, uses the exception context to transition to activity that invokes the Recover From Last Error action. This action resumes the activity that threw the exception.

### How do I know when to pass variable names or expressions to an action?

In the Action panel, Variable Name (such as Return Variable Name) is a name, not an expression (left hand part of data assignment), so the Variable Name is not surrounded by double quotes. Variable, on the other hand, is an expression (right hand part of assignment). An expression can be a combination of constant data and variables. For example, 12, x, "result", x + 12; "result=" + x, and so on.

### How can I log an exception?

The workflow engine stores a list of exceptions in a workflow variable called `$_exceptionList`. The report template uses this variable to display a list of exceptions. Therefore, to log an exception in a workflow template, simply log this variable.

### Is the value returned by an application automatically assigned to a workflow variable?

Yes. The Return Variable Name field for each application invocation specifies the variable name that will contain the return value.



Does an application return a value when it is invoked asynchronously?

No, an asynchronous invocation returns to the caller before the invoked application returns a value.

## Reports

How do I edit a workflow report template?

You can download the default report template from the Configurations home page. After editing the template, upload the file. The uploaded report template takes effect immediately; you do not need to restart the server.

How can I render a variable or property in a report?

To render variable and property information in a report, configure the report template using the scope attribute of the `<Text>` element. See [Creating a Custom Report Template](#) on page 122 for more information.

What statistical information is defined by the report engine?

The following data can be rendered in a report:

- Start time, wait time, and end time
- Execution status
- IsAlertSent
- IsReminderSent
- IsEscalationSent
- Escalation List
- Reminder List
- Alert List
- Error List



## B Verisign Certificate Management

Select Identity registers users through Services. If you wish to manage Verisign certificates during the user registration process, you can rely on the SI CertificateRequestProcess workflow template. This workflow template calls the WorkflowCertificateRequest external call, which is provided by Select Identity and is configured for managing Verisign certificates.

Before you can use the WorkflowCertificateRequest external call to manage Verisign certificates, you must perform the following steps:

- 1 Create a JNDI data source on the Select Identity server to access this table.
- 2 On the Select Identity server, create a file called `Verisign.properties`, which is used by the WorkflowCertificateRequest external call in the SI CertificateRequestProcess workflow template. Here is an example of the file:

```
verisign.dataSource=jdbc/Verisign
verisign.certtablename=cert_vsaa
verisign.certtable.record.delete=no
verisign.certtable.status.new=NEW
verisign.certtable.status.approve=APR
verisign.certtable.status.reject=NEW
verisign.certificate.tag=yes
verisign.mapping=com/trulogica/truaccess/externalcall/
    certificatefunction/verisign/Verisign.Mappings
verisign.Certificate.Type=X.509
```

- 3 Create a file called `Verisign.Mappings`, which is referenced in the `Verisign.properties` file and contains parameters that are used by Select Identity to communicate with Verisign during certificate requests. Here is the format of the file:

```
FORM_VerisignTarget=VerisignTarget
FIELD_operation=operation
FIELD_mail_firstName=FirstName
```

```
FIELD_mail_lastName=LastName
FIELD_mail_email=Email
KEY_public_key=
FIELD_employeeID=EmployeeID
FIELD_challenge=CertRegPswd
FIELD_additional_field5=AddnField
FIELD_form_file=FormFile
FIELD_VHTML_FILE=VHTMLFile
FIELD_authenticate=authenticate
FIELD_additional_field3=AddnField3
FIELD_additional_field4=AddnField4
FIELD_additional_field6=AddnField6
FIELD_user_data_1=user_data_1
FIELD_user_data_2=user_data_2
FIELD_user_data_3=user_data_3
FIELD_user_data_4=user_data_4
```

The `FORM_VerisignTarget` property provides information to connect to a Verisign URL for certificate requests. It can contain a Verisign URL or a Select Identity attribute in the Certificate service that stores the Verisign URL. In the above case, the value `VerisignTarget` points to a Select Identity attribute configured at the root Service Role level in the Certificate service, and its value is fixed.



Contact Verisign for values needed in this file.

- 4 After creating these files, copy them to the `install_dir/com/truologica/truaccess/externalcall/certificatefunction/verisign` directory, which will replace the files provided by Select Identity.

Then, when configuring the Service that will register the users, perform the following:

- 1 Define attributes for the DN and CertRegPswd parameter values that are passed to the WorkflowCertificateRequest external call by the SI CertificateRequestProcess workflow template. Specifically, ensure the following:
  - The CertRegPswd attribute must be visible and updatable in the view that performs the Add operation.
  - The DN attribute must not be present in the view that performs the Add operation.

- Neither attribute should be present in any other view.
- 2 When configuring the business relationship, the Add New User and Add Service events must use the SI CertificateRequestProcess workflow.



---

# glossary

## Acronyms

### A

#### **AC**

access control

#### **AD**

Adaptive Connector

#### **ACL**

access control list

#### **AD Connector**

Active Directory Connector

#### **ADK**

application development kit

#### **ADO**

ActiveX Data Objects

#### **ANSI**

America National Standards Institute

#### **APA**

auto port aggregation

**API**

application program interface

**ARPA**

Advanced Research Projects Agency

**ASCII**

American Standard Code for Information Interchange

**B****BSIM**

Business Service Identity Management

**D****DBA**

database analyst

**DLL**

dynamic-link library

**DNS**

domain name system

**DHCP**

dynamic host configuration protocol

**DHTML**

dynamic hypertext markup language

**DSN**

data source name

**DSN**

digital switched network



**DTD**

document type definition

**E****EJB**

enterprise java bean

**F****FQDN**

fully qualified domain name

**FTP**

file transfer protocol

**G****GIF**

graphics interchange format

**GUID**

globally unique identifier

**H****HSRP**

Hot Standby Router Protocol

**HTTP**

hypertext transfer protocol

**HTTPS**

Hypertext Transfer Protocol Secure

|

**IBM**

International Business Machines

**IP**

internet protocol

**ISO**

International Organization for Standardization

J

**J2C**

Java 2 connector

**J2SDK**

Java 2 software developer kit

**J2EE**

Java 2 enterprise edition

**JAR**

Java application resource

**JCA**

Java connection architecture

**JDBC**

Java database connectivity

**JMS**

Java messaging services

**JNDI**

Java naming directory interface

**JSP**

Java server protocol

**JVM**

Java virtual machine

**K****KB**

kilobyte

**L****LDAP**

lightweight directory access protocol

**LDIF**

lightweight data interchange format

**LLB**

local location broker

**M****MAPI**

messaging application programming interface

**MB**

megabyte

**MHz**

megahertz

**MSSQL**

MicroSoft Structured Query Language

**MTA**

message transfer agent

**P****PDF**

portable document format

**R****RAR**

resource adapter archive

**RDF**

reporting data feeder

**RPC**

remote procedure call

**S****SDK**

software developer kit

**SHA**

secure hash algorithm

**SMTP**

simple mail transfer protocol

**SNMP**

simple network management protocol

**SOAP**

simple object access protocol

**SP**

service pack

**SSL**

secure socket layer

**SSO**

single sign on

**SPML**

service provisioning markup language

**SQL**

structured query language

**T****TCP / IP**

transmission control protocol / internet protocol

**U****URI**

uniform resource identifier

**URL**

uniform resource locator

**UTF-8**

unicode transformation format (eight-bit character conversion)

**V****VM**

virtual machine

## **VNC**

virtual network computing

## W

## **WAS**

web application server

## **WAR**

web application repository

## X, Y, Z

## **XML**

extensible markup language

## **XSD**

XML schema definition

## **XSL**

Extensible Style Sheet Language

## Terms

### A

#### **Access Control List (ACL)**

An abstraction that organizes entitlements and controls authorization. An ACL is list of entitlements and users that is associated with a secured object, such as a file, an operation, or an application. In an ACL-based security system, protected objects carry their protection settings in the form of an ACL.

#### **access management**

The process of authentication and authorization.

**action**

When the context is a user action in the user interface, an operation that can be carried out by an OpenView application. Actions are typically performed on managed object. Select manually executed actions through a menu item or toolbar buttons. Actions can also be configured to automatically occur in response to an event, message, or a change in information in the management database.

When the context is based on OVSI policy, an actions is an operation carried out as a result of the activation of a reconciliation policy and the successful evaluation of a rule or conditions within that policy.

See also: [capability](#)

**activate**

To make active or functional

**activity**

A logical step in a process; A task that may occur when a workflow template is executed (in Workflow Studio). Activities are the core components of workflow templates; they do the work necessary to provision users. An activity can set a property to be used throughout the workflow, track approvals, start a subworkflow, send email, call an external application, and so on.

**adapter**

Software that allows information interpretation between two or more software products or components.

**AD Connector**

Active Directory Connector. A type of interface used to connect HP OpenView Select Identity with the applications it serves on servers that communicate using the Active Directory protocol.

**admin role**

A template that defines the administrative actions performed by a user. Create an Administrative Service to provide access to roles so that users gain access to the Service. Users with administrative roles may grant their set of roles to another administrator within their Service context.

**advanced customization**

Less common types of customization which are more flexible in their capabilities and complex in their implementation than typical customizations. As with other customizations, advanced customizations are done to meet the needs and preferences of a particular customer or user.

**agent**

A program or process running on a remote device or computer system that responds to management requests, performs management operations, or sends performance and event notification. An agent can provide access to managed objects and MIB variables, interpret policy for resources and do configuration of resources; The component of an agent-based connector that resides in the same system as the resource. It listens for a changes in the user data made in the resources, then reports that change to HP OpenView Select Identity by communicating through a connector interface.

**agent-based connectors**

Two-way connector interface. There are two components: the connector that resides in the same system as HP OpenView Select Identity, and the agent, which resides in the same system as the resource. The agent listens for changes made in the resource, and contacts the resource about changes made in Select Identity.

**agentless connectors**

One way connectors. Connectors reside in the Select Identity server and does the communication brokering with the resource.

**application**

Packaged software that provides functionality that is designed to accomplish a set of related tasks. An application is generally more complex than a tool.

**application deployment**

The installation and activation of application components so that they work in the business environment.

**Application Program Interface (API)**

A set of routines, protocols, and tools used to build a software application; An interface that enables programmatic access to an application.



**approval process**

The process of approving the association, modification, or revocation of entitlements for an identity. This process is automated of these through workflow templates.

**approver**

A Select Identity administrator who has been given approval actions through an Admin Role.

**assigned policy**

A policy that has been assigned to one or more resources in the computing environment but which has not yet been deployed or installed on those resources.

**asynchronous subprocess**

A process that proceeds at its own pace independent of other processes and subprocesses.

**attribute**

An individual field that helps define an identity profile. For each identity, an attribute has a corresponding value. For example, an attribute could be “department” with possible values of “IT,” “sales,” or “support.”

**attribute external call**

Small programs that are written to generate values automatically for that attribute (value generation), define constraint values for the attribute (value constraint), or validate the value that is entered for that attribute (data validation). Each attribute can have each of these types of external calls.

**attribute name-value pair**

An attribute name-value pair is combination of an attribute identifier and the value of that attribute for a specific object. An example of an attribute-name-value pair for a person would be Name: John Smith.

**audit engine**

logs and stores all audit-related activities, e.g., when changes are made and who made them.

**Audit Report**

A report that provides regular account interaction information.

**authentication**

Verification of an identity's credentials.

**authoritative source**

A resource that has been designated as the “authority” for identity information. Select Identity accounts can be reconciled against accounts in an authoritative source.

**automatic action**

A pre-configured program or script that is executed in response to an event, message, or a change in information in the management database. without operator intervention.

**B****bandwidth**

The transmission capacity of an electronic line such as a communications network, computer bus, or computer channel. It is expressed in bits per second (for example, 56 kbps), bytes per second or in Hertz (cycles per second). When expressed in Hertz, the frequency may be a greater number than the actual bits per second, because the bandwidth is the difference between the lowest and highest frequencies transmitted. (TECH).

**block**

A special type of activity that serves two purposes: to define information to be used by a subset of activities (block-level properties) and to provide block-level reporting. For example, you might define a block that submits an approval request, waits for the response, and returns the status of the request to the workflow. In other words, think of a block as a process within a template.

**block type**

A property assigned to a block in a workflow template using the blockType property in end block activity. The report template uses this property to identify how block information is rendered in the resulting report.

### **Boolean operator**

A logical operator that defines the context in which attribute values are compared to satisfy a query or policy. For example:

AND - Both conditions have to be satisfied.

OR - At least one condition has to be satisfied.

NOT - No instance of this condition is allowed.

### **browser**

A module within a work space that presents one or more views of objects and provides functionality for interacting with the objects and the views.

### **business service**

A product or facility offered by, or a core process used by, a business in support of its day-to-day operations. Example business services could include an online banking service, the customer support process, and IT infrastructure services such as email, calendaring, and network access.

See also: [service](#)

### **Business Service Identity Management (BSIM)**

An organizational model that introduces new abstractions that simplify and provide scale to the business processes associated with identity management. These abstractions are modeled after elements that exist in businesses today and include Services and Service Roles.

## **C**

### **capability**

Actions that can be performed within the HP OpenView Select Identity client.

See also: [action](#)

### **challenge and response**

A method of supplying alternate authentication credentials, typically used when a password is forgotten. Select Identity challenges the end user with a question and the user must provide a correct response. If the user answers the question correctly, HP OpenView Select Identity resets the password to a random value and sends email to the user. The challenge question can be configured by the administrator. The valid response is stored for each user

with the user's profile and can be updated by an authenticated user through the Self Service pages.

### **client**

When the context is network systems, a computer system on a network that accesses a service from another computer (server).

When the context is software, a program or executable process that requests a service from a server

### **client console**

An instance of the user interface that appears on the client system while the application runs on a server.

### **condition type**

An abstraction or categorization of a condition that determines to the particular kind of data that is valid for the parameter values in the condition and how those values will be used. For example a condition type could be Source IP Address which indicates that values must have 4 numbers separated by decimals with the value for each number being in the range of 0 to 255. Since the condition type is "Source" IP Address, the IP addresses will only be evaluated for sources not destinations.

### **configuration file**

A file that contains specifications or information that can be used for determining how a software program should look and operate.

### **configuration**

In a hardware context, a particular set of inter-related components that make up a computer system. For example the components of a computer system may include a keyboard, pointing device, memory, disk drives, modem, operating system, applications and printer. The configuration of the computer system determines the way that it works and the way that it is used.

In a network context, the complete set of inter-related systems, devices and programs that make up the network. For example the components of a network may include computer systems, routers, switches, hubs, operating systems and network software. The configuration of the network determines the way that it works and the way that it is used.

In a software context, the combination of settings of software parameters and attributes that determine the way the software works, the way it is used, and how it appears.

**configure**

To define and modify specified software settings to fulfill the requirements of a specified environment, application or usage.

**Configuration Report**

A report that provides current system information for user, administrator, and Service management activities.

**connection**

A representation of a logical or physical relationship between objects.

**connector**

A J2EE connector interface that communicates with the system resource applications that contain your identity profile information.

**console**

An instance of the user interface from which the user can control an application or set of applications.

**context**

An HP OpenView Select Identity concept that defines a logical grouping of users that can access a Service.

**credential**

A mechanism or device used to verify the authenticity of an identity. For example, a user ID and password, biometrics, and digital certificates are considered credentials.

**customization**

The process of designing, constructing or modifying software to meet the needs and preferences of a particular customer or user.

customize

To design, construct or modify software to meet the needs and preferences of a particular customer or user.

**customize**

To design, construct or modify software to meet the needs and preferences of a particular customer or user.

D

**database**

A repository of data that is electronically stored. Typically databases are organized so that data can be retrieved and updated.

**data file**

An SPML file that enables you to define user accounts to be added to Select Identity through Auto Discovery or Reconciliation.

**data type**

A particular kind of data; for example

**deactivate**

To deliberately stop a component or object from working.

**delegated administration**

The ability to securely assign a subset of administrative roles to one or more users for administrative management and distribution of workload. Select Identity enables role delegation through the Self Service pages from one administrator to another user within the same Service context.

**delegated registration**

Registration performed by an administrator on behalf of an end user.

See also: [self-registration](#)

**deploy**

To install and start software, hardware, capabilities, or services so that they work in the business environment.

**deployed application**

An application and its components that have been installed and started to work in the business environment.

**deployed policy**

A policy that is deployed on one or more resources in the computing environment.

**deployment**

The process of installing and activating software, hardware, capabilities or services so that they work in the business environment.

**deployment package**

A software package that can be deployed automatically and installed on a managed node.

**deprecate**

To lower the status of a hardware or software object to indicate that it can be taken out of use in the future

**device**

A generic term for a piece of hardware equipment that can be attached to a computer or a network. Examples of a device are a printer, a router, a switch, a load-balancer, a disk drive or a modem.

**disable**

To make unable to be used.

**dismiss**

Dismiss is an action that causes a message or other notification associated with a problem or situation to be removed from the browser. Messages are typically dismissed when the operator has resolved the situation that led to the message.

**disown**

The act of relinquishing responsibility for resolving a problem or situation associated with a message or other notification.

**DNS domain**

A set of computers and other network devices that are collectively addressable by a portion of an IP address or by the highest subdivision of the domain name that indicates the entity owning the address. For example all computers whose host name share the suffix .hp.com are in the same DNS domain.

**domain**

A set of computers and other network devices that are treated or managed as a unit.

**double-click**

To press and release a pointing device's button twice in rapid succession. Double-clicking is a time-dependent action. Clicking twice in the same location at slow speed (click-delay-click) is not a double-click.

**downtime**

The amount or percentage of time that a service, software, or hardware resource remains non-functional.

**dynamic parameters**

Parameters whose values are determined during program execution.

**E****enable**

To make able to use.

**end user**

A role associated to every user in the Select Identity system that enables access to the Self Service pages.

**entitlement**

An abstraction of the resource privileges granted to an identity. Entitlements are resource-specific and can be resource account IDs, resource role memberships, resource group memberships, and resource access rights and privileges. Entitlements are also considered privileges, permissions, or access rights.



**event**

An event is an unsolicited notification such as an SNMP trap or WMI notification generated by an agent or process in a managed object or by a user action. Events usually indicate a change in the state of a managed object or cause an action to occur.

**event attribute**

A characteristic or property of an event.

**event correlation**

The evaluation of multiple events or notifications that are related to a single incident or problem, to produce a single message. Event correlation is used to reduce the number of messages that are presented to an operator in a message browser.

**event creation time**

The time an event was created in Universal Coordinated Time (UTC)

**event syntax**

The rules governing the structure and content of an event.

**event type**

A classification of an event into a particular category that further defines the nature of the event.

**export**

To format and move information from the current application to a location outside the current application.

**expression**

A combination of workflow variables and constant values to be evaluated. An expression can be assigned to a new variable or passed to an application as an argument. If you are familiar with a programming language, an expression used in a workflow template is like C or Java expression. Example of expressions can be found in action input parameters, application return values, and transition conditions.

**extend**

The act of increasing the capabilities, scope, or effectiveness of a program.

**extensible**

Capable of being extended.

**external call**

A programmatic call to a third-party application or system for the purpose of validating accounts or constraining attribute values.

**external system ID**

An identifier that uniquely identifies a principal that is an external system.

**F****filter**

A software feature or program that functions to screen data so that only a subset of the data is presented or passed. Filters allow matching-relevant information to be extracted and acted on while non-matching-irrelevant information is held back.

**find**

The act of seeking of specific data or objects within the management application or set management applications based on specified criteria.

**form**

An electronic document used to capture information from end users. Forms are used by Select Identity in many business processes for information capture and system operation; A presentation mechanism that contains information and controls for obtaining user input (for example, text fields, radio buttons, lists).

**foundation**

A program that acts as the basic structure to support other software modules or programs that provide additional functionality for the user.

**function**

A general term for a portion of a program that performs a specific task.

## H

### **hierarchy**

Elements organized in successive levels with each lower level being subordinate to the one above.

### **HP OpenView**

A family of network and system management products, and an architecture for those products. HP OpenView includes development environments and a wide variety of management applications.

## I

### **icon**

An on-screen image that represents objects that can be monitored or manipulated by the user or actions that can be executed by the user.

### **icon class**

The portion of an icon that identifies the type or classification of the object being represented by the icon. For example, the network object class is represented by a circle surrounding a more complex image.

## **ID**

identifier

### **identifier**

A name that within a given scope that uniquely identifies the object with which it is associated.

### **identity**

The set of authentication credentials, profile information, and entitlements for a single user or system entity. Identity is often used as a synonym for “user,” although an identity can represent a system and not necessarily a person.

### **identity management**

The set of processes and technologies involved in creating, modifying, deleting, organizing, and auditing identities.

**import**

To format and move information from a location outside the current application into the current application.

**install**

To load a product or component of a product onto a computer system or other network or system device. Installation typically involves running initial configuration scripts that are part of the installation process.

**instance**

See: [workflow instance](#)

**internationalization**

The design of software so that a single binary can support the varied cultural and linguistic conventions that exist in different countries or locales. Internationalized software allows users to interact with the software in the user's native language including the input and output of data in the native language, as well as support for the conventions and rules applicable to the user's locale. The ANSI locale model is used in internationalized software.

**J****Java**

Object oriented programming language.

**JCA**

Java Connection Architecture. Architecture used to build interfaces between J2EE compliant products and other resources.

**JVM**

Java Virtual Machine. A platform independent execution environment that conversant Java bytecode into machine language then executes it.

**L****LDIF**

File that modifies and deletes directory objects.

**list**

If the context is a GUI, a set of selectable items. If the context is data, a variable-length ordered set of values all of the same data type.

**locale**

The locale collectively represents the location or country of the user, the language of the user, and the code set in which the user's data is represented. The locale is related to the language sensitive presentation of applications.

**locale model**

The software through which the user declares their desired language at application start up. The local model determines the set of files, tables, or collection of programs that are used to initialize an application so that it is sensitive to the user's language.

**localization**

Localization refers to the set of tasks that need to be accomplished to enable a product to work acceptably in a specific locale. The localization tasks include translating documentation, translating text and graphics that are presented to the user, and providing locale specific fonts and other functionality when needed.

**M****management**

The ongoing maintenance of an object or set of objects, including creating, modifying, deleting, organizing, auditing, and reporting.

**message key**

A message attribute that is a string used to identify messages that were triggered from particular events. The string summarizes the important characteristics of the event. Message keys can be used to allow messages to acknowledge other messages, and allows for the identification of duplicate messages.

## N

### **node**

When the context is network, a computer system or device (for example, printer, router, bridge) in a network.

When the context is a graphical point to point layout, a graphical element in a drawing that acts as a junction or connection point for other graphical elements.

### **notifications**

The capability that enables you to create and manage templates that define the messages that are sent when a system event occurs.

## P

### **package**

A set of related programs or software files grouped together as a single object for a common purpose.

### **password reset**

The ability to set a password to a system-generated value. Select Identity uses a challenge and response method to authenticate the user and then allow the user to reset or change a password.

### **persistent variable**

A variable that is persisted after an instance is passivated. To extend the variable life cycle to the entire instance, you must create the variable to be persistent. This enables the variable to be created before a wait activity, and it will be accessible after the workflow instance resumes. To make a variable persistent, precede the name with \$. For example, the \$retryCount variable is persistent while retryCount is not.

See also: [workflow variable](#)

### **policy**

A set of regulations set by an organization to assist in managing some aspect of its business. For example, policy may determine the type of internal and external information resources that employees can access.

**policy management**

The process of controlling policies (for example, creating, editing, tracking, deploying, deleting) for the purposes of network, system or service management.

**port**

If the context is hardware, a location for passing information into and out of a network device.

**process**

A repeatable procedure used to perform a set of tasks or achieve some objective. Whether manual or automated, all processes require input and generate output. A process can be as simple as a single task or as complicated a multi-step, conditional procedure.

See also: [approval process](#)

**profile**

Descriptive attributes associated with an identity, such as name, address, title, company, or cost center.

**property**

See: [workflow property](#)

**provisioning**

The process of assigning authentication credentials to identities.

**R****reconciliation**

The process by which Select Identity accounts are synchronized with a system resource. Accounts can be added to the Select Identity system through the use of an SPML data file.

**registration**

The process of requesting access to one or more resources. Registration is generally performed by an end user seeking resource access, or by an administrator registering a user on a user's behalf.

See also: [delegated registration](#), [self-registration](#)

**request**

An event within the Select Identity system for the addition, modification, or removal of a user account. Requests are monitored through the Request Status capability.

**resource**

Any single application, database, or information repository. Resources typically include applications, directories, and databases that store identity information.

**role**

A simple abstraction that associates entitlements with identities. A role is an aggregation of entitlements and users, typically organized by job function.

See also: [admin role](#)

**rule**

A programmatic control over system behavior. Rules in Select Identity are typically used for programmatic assignment of Services. Rules can also be used to detect changes in system resources.

**S****self-registration**

Registration performed by an end user seeking access to one or more resources.

See also: [delegated registration](#)

**self service**

The ability to securely allow end users to manage aspects of a system on their own behalf. Select Identity provides the following self-service capabilities: registration, profile management, and password management (including password change, reset, and synchronization).

**service**

A business-centric abstraction representing resources, entitlements, and other identity-related entities. Services represent the products and services that you offer to customers and partners.



**service attribute**

A set of attributes and values that are available for or required by a Service. Attributes are created and managed through the Attributes pages.

See also: [attribute](#)

**service role**

A Select Identity abstraction that defines how a logical grouping of users will access a Select Identity Service. The Select Identity Service is a superset of all the identity management elements of a business service.

**service view**

A restricted view of a Service that is valid for a group of users. Views enable you to define a subset of Service registration fields, change field names, reorder fields, and mask field values for specific users.

**single sign-On (SSO)**

A session/authentication process that permits a user to enter one set of credentials (name and password) in order to access multiple applications. A Web SSO is a specialized SSO system for web applications.

**SPML Data File**

See: [data file](#)

**submodule**

A portion of a software module that provides a subset of the functionality provided by the module. A sub-module performs a specific task or presents a specific set of data.

**suspend**

To halt for a time a computer operation preserving the state of that operation.

**synchronous subprocess**

A process that must complete before the invoking process can proceed.

**syntax**

The rules governing the structure and content of a language or the description of an object.

**system administrator**

The role of a person who does configuration and maintenance on a computer system or the software on the system.

**T****template**

See: [workflow template](#)

**trace log**

An output file containing records of the execution of application software

**transit delay**

The difference between current time and the event's creation time.

**transition**

The definition of a relationship between activities. You can define that one activity always follows another, or you can define a condition that must be met before the workflow transitions from an activity to one or more others. For example, you can define a transition that only allows the workflow to progress if at least two administrators approve a request. If the request is not approved, the workflow can transition to an activity that sends email notification to an administrator.

**U****URL**

Acronym for Uniform Resource Locator or Universal Resource Locator, the address of a computer or a document on the Internet.

**user import**

The process of adding user accounts to the Select Identity system for a specified Service through the use of a data file.

**users**

The functionality that provides consistent account creation and management across Services.

## V

### **variable**

See: [workflow variable](#)

### **variable expression**

See: [external call](#)

## W

### **Web Service Definition Language (WSDL)**

File format that the Application Definition file uses to define a web service application to be a workflow application. The workflow engine reads the web service invocation parameters through WSDL. A web service can reference a WSDL URL remotely or download it first as a local file and then read the file locally at run-time.

### **workflow engine**

A system component that executes workflows and advances them through their flow steps.

### **workflow external call**

A “subroutine” that is called during the workflow process. This could be an external application invocation such as a small custom application that calls external processes outside of the normal workflow process

### **workflow instance**

An invocation of a workflow template. An instance starts when it is created and ends when it completes (when the last activity is executed). An instance’s status and other associated information can be viewed once an instance is created.

### **workflow process**

The tasks, procedural steps, organizations or people involved, and required input and output information needed for each step in a business process. In identity management, the most common workflows are for provisioning and approval processes.

**workflow property**

A name-value pair, where the value is a text string. A property stores static data that cannot be changed at runtime. It can be accessed by the workflow API and report template. There are three levels of properties: global, block, and activity.

**workflow studio**

The functionality that enables you to create and manage workflow templates.

**workflow template**

A model of the provisioning process that enables Select Identity to automate the actions that approvers and systems management software must perform.

**workflow variable**

A name-value pair that can be created or changed at runtime in a workflow instance through actions, a workflow API call, or returned by an application invocation. It can be accessed by workflow API, workflow template, and report template. There are levels of variables: global, block, and activity.

See also: [persistent variable](#)

---

# Index

## Symbols

- <Block> block, 115
- <Block> element, 114
- <Column> element, 118, 120
  - attributes, 120
- <Column> element attributes
  - ColId, 120
  - ColLabel, 120
- <Instance> block, 115
- <Table> element, 119
  - attributes, 119
- <Table> element attributes
  - IgnoreNul, 119
  - Name, 119
  - Variable, 119
  - VariableType, 119
- <Table> tag, 118
- <Text> element attributes
  - Name, 116
  - Scope, 116
  - Value, 116

## A

- accessing user attributes, 95
- action
  - Call Subworkflow, 99
- Action Handlers, 5

- actions, 44, 46, 129, 143
  - Add a set of Users to a List of Services, 97
  - Add a set of users to a list of services, 97
  - Add Item to List, 100
  - Add Item to Map, 100
  - Application invocation, 83
  - Application Name, 84
  - Approvers External Call, 95, 96
  - Call Subworkflow, 99
  - Call subworkflow, 99
  - Check email Verification, 85
  - Create Workflow Approver, 88
  - defining, 82
  - definition, 3
  - Email Notification, 91
  - email notification, 90
  - External Call, 95
  - external calls, 95
  - Get approvers by role, 86
  - Get Block Variable, 103
  - Log Message, 98
  - Notify approvers, 87
  - Notify Approvers (Default Email Template), 87
  - overview, 61
  - Post Provision, 92, 93
  - Post Provision in Bulk Processing to save data, 94
  - Post Provision in Reconciliation to save data, 94
  - Provisioning Task, 89
  - Recover From Last Error, 144
  - Recover from Last Error, 102
  - Recovery From Last Error, 144
  - Run script, 98
  - Save Email Notification, 86
  - Send an Email Notification, 91, 92
  - Send Email, 101
  - Set Block Variable, 103
  - Set Variable, 97
  - Throw Exception, 98, 144

- XPath, 102
- activities, 140
  - actions, 82
  - add, 45, 46
  - begin, 42, 79
  - blocks, 132
  - catchException, 141, 144
  - catch exception, 79
  - concurrently running, 142
  - definition, 3
  - End Approval, 134
  - end block, 141, 142, 143
  - errorEnd, 11, 46, 80
  - fault handler, 79, 144
  - general properties, 81
  - Id property, 81
  - Notify Error, 46
  - overview, 78
  - PostProvision, 45
  - properties, 81
  - Provision, 45, 46
  - Provisioning, 129
  - run concurrently, 142
  - start block, 141, 142
  - synchronizing, 142
  - types, 79
  - variables, 74
  - wait, 140, 143, 144
  - Wait Activity check box, 81
- Activity ID Variable parameter, 99
- Add a set of Users to a List of Services
  - action, 97
- Add a set of users to a list of services action, 97
  - actions
    - Add a set of Users to a List of Services, 97
- Add Item to List action, 100
- Add Item to Map action, 100

- Alert Handler property, 70
- Alert Repeat Count property, 70
- Alert Timeout property, 70
- Application invocation actions, 83
- Application Name action, 84
- applications
  - Add a set of users to a list of services, 97
  - Approvers External Call, 95
  - calling, 83
  - Check Email Verification, 85
  - Create Workflow Approver, 88
  - Email Notification, 90
  - external, 124
  - External Call, 95
  - Get approvers by role, 86
  - Notify Approvers, 87
  - Notify Approvers (Default Email Template), 87
  - Post Provision, 92
  - Post Provision in Bulk Processing to Save Data, 94
  - Post Provision in Reconciliation to Save Data, 94
  - Provisioning Task, 89
  - Send an Email Notification, 91
  - Stated Post Provision, 93
- approver lists in workflow templates, 62
- Approver List Variable parameter, 87, 89
- Approver List Variable parameter, 88
- Approvers External Call action, 95, 96
- architecture
  - Workflow Studio, 4
- Asynchronous Invocation fields, 85, 86, 87, 88, 89, 91, 92, 93, 94, 95, 96, 97
- Asynchronous parameter, 99

- attributes
  - <Column> element, 120
  - <Table> element, 119

## **B**

- begin activity, 42, 79
- block activities, 59, 132
- block activity, 142
- Block Id property, 66
- block properties
  - defining, 142
- block property, 65
- blocks, 110, 141
  - <Block>, 115
  - <Instance>, 115
  - activities, 132
  - creating, 141
  - creation guidelines, 81
  - definition, 4
  - end block activity, 141, 142
  - overview, 78
  - purpose, 78
  - start block activity, 141, 142
  - time out, 143
  - variables, 74
- Block Type property, 67
- block types, 110
  - approval, 110
  - emailVerify, 110
  - Instance, 111
  - postprovisioning, 110
  - provisioning, 110
  - ReconciliationPostProvision, 110
  - UserServiceProvisioning, 110
- Block variable, 76
- block variables, 142
  - accessing, 142

bulk request event, 2  
buttons, on Workflow Studio toolbar, 9

## C

calling other workflows, 99  
Call Subworkflow action, 99  
Call subworkflow action, 99  
catchException activity, 80, 141, 144  
CC Variable parameter, 101  
certificate management, 147  
Check email Verification action, 85  
Child Callback parameter, 99  
concepts and terms, 3  
copy templates, 39  
create, 42  
    custom report template, 122  
    provisioning template example, 40  
    workflow template, 59  
Create Workflow Approver action, 88  
Create Workflow Approver Task, 89  
creating  
    blocks, 81  
    external applications, 124

## D

Default Approver parameter, 96, 97  
Default Email Template parameters, 88

default templates, 12  
    ReconciliationDefaultProcess, 33  
    ReconciliationDefaultProcessMove, 35  
    SI CertificateRequestProcess, 20  
    SI Email Only, 24  
    SI EmailVerifyAndApproval, 26  
    SI OneStageApproval, 29  
    SI Password Change Provisioning, 17  
    SI PasswordExpire Email, 25  
    SI Provisioning Only, 12, 16, 17  
    SI Provisioning Only Bulk, 14  
    SI Provisioning Only With ExclusionRul,  
        16, 17  
    SI Recon User Enable Disable Workflow,  
        38  
    SI ThreeStageApproval, 32  
    SI TwoStageApproval, 30  
    SI User Enable Disable Workflow, 36  
    UserAccountExpirationWF, 25

definitions  
    actions, 3  
    activities, 3  
    blocks, 4  
    transitions, 4

delegated-registration request events, 2  
delete, 40  
delete templates, 40

## E

Element Name Variable parameter, 100  
elements  
    <Block>, 114  
    <Column>, 118, 120  
    <Table>, 119  
    <Text>, 115  
Element Value Variable parameter, 100  
Element Value Variable parameter, 100  
email in workflow templates, 62



- email notification action, 90
- Email Template parameter, 91, 92
- email templates referenced in workflow templates, 60
- End Approval activity, 134
- end block activity, 143
  - activities
    - end block, 142
- End Block Id property, 66
- engine-defined variable
  - alertList, 119
  - escalationList, 119
  - exceptionInfoList, 119
  - provisioningStatReport, 119
  - pushList, 119
  - ReconPostProvisionStatReport, 119
  - remindList, 119
- errorCode variable, 90
- errorEnd
  - activity, 80
  - example, 46
- errorEnd activity, 11, 46, 80
- error logging, 46
- errors
  - handling, 46, 80
- Escalate To property, 68
- Escalation Handler property, 68
- Escalation property, 69
- Escalation Repeat Count property, 69
- escalation timeout property, 69
- exception
  - logging, 144
- exception expression, 141

- exceptions
  - catchException, 80
  - in workflow templates, 64
  - recovering from, 102
- external applications, registering, 124
- External Call action, 95
- External Call Name parameter, 95, 96, 97
- external calls, 127, 131
  - Approvers, 95
  - Entitlement Rules, 129
  - LoadUserServices, 130
- external calls action, 95
- external calls referenced in workflow templates, 60

## F

- FAQ, 139
- Fault handler activity, 79
- fault handler activity, 144
- fields
  - Asynchronous Invocation, 85, 86, 87, 88, 89, 91, 92, 93, 94, 95, 96, 97
  - Return Variable Name, 85, 86, 87, 88, 89, 91, 92, 93, 94, 95, 96, 97
- frequently asked questions, 139
- From Variable parameter, 101

## G

- Get approvers by role action, 86
- Get BlockVariable action, 103
- getting started, 7
- global attribute, ReportTemplateId, 66
- guidelines for creating blocks, 81

## I

Id property, 81  
Instance block, 111  
Introduction, 1

## J

join count, 65  
Join Count property, 68, 134, 142  
join transition, 104  
Join Type check box, 81  
Join Type property, 81

## L

List Variable Name parameter, 100  
LoadUserServices external call, 130  
log an exception, 144  
logging in workflow templates, 64  
Log Message action, 98

## M

map variable, 43  
Map Variable Name parameter, 100  
modify templates, 39

## N

Name-value Map Variable parameter, 92  
New Collection parameter, 100  
non-persisted variables, 75  
Notification Action parameter, 87, 88  
notifications in workflow templates, 63  
Notify Approvers, 87  
Notify Approvers (Default Email Template),  
88

Notify Approvers (Default Email Template)  
action, 87

Notify approvers action, 87

Notify Error activity, 46

## O

overview  
actions, 61  
creating a template, 11  
Workflow Studio interface, 7

## P

parallel approval workflow design, 132

parameters

Activity ID Variable, 99  
Approver List Variable, 87, 88, 89  
Asynchronous check box, 99  
CC Variable, 101  
Child Callbackactions  
    Call Subworkflow, 99  
Default Approver, 96, 97  
Default Email Template), 88  
Element Name Variable, 100  
Element Value Variable, 100  
Email Notification action, 91  
Email Template, 92  
External Call Name, 95, 96, 97  
From Variable, 101  
List Variable Name, 100  
Map Variable Name, 100  
Name-value Map Variable, 92  
New Collection, 100  
Notification Action, 87, 88  
Results List, 102  
Result Variable Name, 102  
Subject Variable, 101  
Template ID Variable, 99  
To Variable, 101  
XPath Variable, 102

- persisted variables, 75
- persistent variables, 143
- Post Provision action, 92, 93
- PostProvision activity, 45
- Post Provision in Bulk Processing to save data, 94
- Post Provision in Reconciliation to save data, 94

- properties
  - Alert Handler, 70
  - Alert Repeat Count, 70
  - Alert Timeout, 70
  - approvers, 62
  - block, 65
  - Block Id, 66
  - Block Type, 67
  - email, 62
  - End Block Id, 66
  - Escalate To, 68
  - Escalation, 69
  - Escalation Handler, 68
  - Escalation Repeat Count, 69
  - escalation timeout, 69
  - Id, 81
  - Join Count, 68, 134, 142
  - join count, 65
  - Join Type, 81
  - notifications, 63
  - overview, 65
  - provisioning, 64
  - Reminder Handler, 70
  - Reminder Repeat Count, 72
  - Reminder Timeout, 71
  - ReportTemplateId, 66
  - Role Name, 68, 86
  - Split Type, 81
  - Start block Id, 66
  - system defined, 66
  - Timeout After, 72
  - Timeout At, 72
  - using, 65
  - variables, 62
  - Wait Activity check box, 81
  - workflow, 65
  - XML Variable Name, 102
- properties in workflow templates, 62
- Provision activity, 45
- provision block
  - add end activity, 45

- provision block properties, 43
- Provisioning activity, 129
- provisioning block
  - add error end activity, 46
  - add transition, 44, 45, 46
  - create post-provisioning block, 44
  - define actions, 43
  - define activity, 46
  - define properties, 43
- provisioning in workflow templates, 64
- provisioningResult variable, 90
- Provisioning Task action, 89

## R

- reconciliation request events, 2
- reconciliation, 130
- ReconciliationDefaultProcessMove template, 35
- ReconciliationDefaultProcess template, 33
- reconciliation event
  - service change, 3
- Reconciliation workflow template, 129
- Recover From Last Error action, 144
- Recover from Last Error action, 102
- Recovery From Last Error action, 144
- registering external applications, 124
- Reminder Handler property, 70
- Reminder Repeat Count property, 72
- Reminder Timeout property, 71
- reporting on workflows, 105

- report template, 110
  - <Table>, 118
  - <Text> element, 115
  - block-level reporting, 114
  - creating, 123
  - creating a custom template, 122
  - custom, 122
  - default, 118, 123, 124
  - editing, 124
  - exporting, 123, 124
  - instance-level reporting, 111
  - structure, 110
- ReportTemplateId property, 66
- request event
  - reconciliation, 2
  - self service, 2
- request events
  - assign to a workflow template, 2
  - bulk request, 2
  - delegated-registration, 2
- request objects, 95
- Results List parameter, 102
- Result Variable Name parameter, 102
- Return Variable Name field, 85, 86, 87, 88, 89, 91, 92, 93, 94, 95, 96, 97
- Role Name property, 68, 86
- role referenced in workflow templates, 60
- rules, 127
- Run script action, 98

## S

- Save Email Notification action, 86
- scenarios, 127
  - adding services to a user, 130
  - enforcing entitlement rules, 127
  - parallel approval workflow design, 132
- Scripting Manager, 5

- self-service request events, 2
- Send an Email Notification action, 91, 92
- Send Email action, 101
- service change reconciliation event, 3
- Service Role, 2
- Set BlockVariable action, 103
- Set Variable action, 97
- SI CertificateRequestProcess template, 20
- SI Email Only template, 24
- SI EmailVerifyAndApproval template, 26
- SI OneStageApproval template, 29
- SI PasswordExpire Email template, 25
- SI Provisioning Only Bulk template, 14
- SI Provisioning Only template, 12, 16, 17
- SI Recon User Enable Disable Workflow, 38
- SI ThreeStageApproval template, 32
- SI TwoStageApproval template, 30
- SI User Enable Disable Workflow, 36
- split transition, 104
- Split Type, 81
- Split Type property, 81
- Start block Id property, 66
- status of workflows, 105
- Subject Variable parameter, 101

- system-defined properties, 66
  - Alert Handler, 70
  - Alert Repeat Count, 70
  - Alert Timeout, 70
  - Block Id, 66
  - Block Type, 67
  - End Block Id, 66
  - Escalate To, 68
  - Escalation, 69
  - Escalation Handler, 68
  - Escalation Repeat Count, 69
  - Escalation timeout, 69
  - Join Count, 68
  - Reminder Handler, 70
  - Reminder Repeat Count, 72
  - Reminder Timeout, 71
  - Report TemplateID, 66
  - Role Name, 68
  - Start block Id, 66
  - Timeout After, 72
  - Timeout At, 72

## T

- template actions, 61
- Template ID Variable parameter, 99

- templates, 40
  - actions, 61
  - copy, 39
  - creating, 11, 59
  - default, 12
  - default report, 123, 124
  - managing, 39
  - modify, 39
  - Reconciliation, 129
  - report, 110, 123, 124
  - saving, 47
  - see default templates, 32, 33, 35, 36, 38
  - SI CertificateRequestProcess, 20
  - SI Email Only, 24
  - SI EmailVerifyAndApproval, 26
  - SI OneStageApproval, 29
  - SI PasswordExpire Email, 25
  - SI Provisioning Only, 12
  - SI Provisioning Only Bulk, 14
  - SI Provisioning Only with ExclusionRule, 16
  - SI TwoStageApproval, 30
  - UserAccountExpirationWF, 25
  - workflow, 129
  - workflow in OVSI, 2
- Throw Exception action, 98, 144
- Throw Exception activity, 144
- Timeout After property, 72
- Timeout At property, 72
- To Variable parameter, 101
- transition, 45
- transitions, 45, 46
  - adding, 44, 45, 46
  - definition, 4
  - join, 104
  - Join Type, 81
  - overview, 103
  - split, 104
  - Split Type, 81

## U

- UserAccountExpirationWF template, 25

- users

- adding services, 130

## V

- variable names

- passing, 144

- variables

- activity-level, 74

- Block, 76

- block, 142

- block-level, 74

- changing, 74

- creating, 74

- errorCode, 90

- non-persisted, 75

- overview, 74

- persisted, 75

- persistent, 143

- provisioningResult, 90

- workflow, 140

- workflow engine defined, 76

- Verisign certificate management, 147

## W

- wait activity, 140, 143, 144

- Wait Activity check box property, 81

- wait instance, 139

- workflow

- create, 42

- macros, 140

- testing, 135

- workflow behavior, 143

- Workflow Design, 132

- workflow engine defined variables, 76

- workflow instance, 144
  - activated, 139
  - started, 139
- Workflow Meta Data Model, 4
- workflow process, 1
- workflow property, 65
- workflows, 132, 139
  - design, 132
  - engine-defined variables, 140
- Workflow State Persistency Manager, 5
- workflow status, 105
  - report template, 110
  - viewing, 105
- Workflow Studio
  - architecture, 4
  - concepts and terms, 3
  - overview, 1
  - overview of creating a template, 11
  - overview of the interface, 7
  - template creation overview, 11
  - toolbar buttons, 9

- workflow templates, 4, 39, 131
  - add, 42
  - approver lists, 62
  - calling other workflows, 99
  - copy, 39
  - creating, 59, 139
  - creating blocks, 141
  - creating provisioning template example, 40
  - defaults, 12
  - delete, 40
  - dependent information, 60
  - designing, 139
  - editing, 129
  - email, 62
  - exceptions, 64
  - general creation steps, 11
  - integration with OVSI, 2
  - logging, 64
  - modify, 39
  - non-persisted variables, 75
  - notifications, 63
  - overview, 1
  - persisted variables, 75
  - process, 139
  - properties, 62, 65
  - provisioning, 64
  - request events, 2
  - saving, 47
  - variables, 74
- workflow variables, 140

## X

- XML, searching, 102
- XML Variable Name property, 102
- XPath action, 102
- XPath Variable parameter, 102