

HP OpenView Select Identity

Software Version: 4.12

Administration Guide

Document Release Date: March 2007
Software Release Date: March 2007



Warranty

Hewlett-Packard makes no warranty of any kind with regard to this document, including, but not limited to, the implied warranties of merchantability and fitness for a particular purpose. Hewlett-Packard shall not be held liable for errors contained herein or direct, indirect, special, incidental or consequential damages in connection with the furnishing, performance, or use of this material.

A copy of the specific warranty terms applicable to your Hewlett-Packard product can be obtained from your local Sales and Service Office.

Restricted Rights Legend

Use, duplication, or disclosure by the U.S. Government is subject to restrictions as set forth in subparagraph (c)(1)(ii) of the Rights in Technical Data and Computer Software clause in DFARS 252.227-7013.

Hewlett-Packard Company
United States of America

Rights for non-DOD U.S. Government Departments and Agencies are as set forth in FAR 52.227-19(c)(1,2).

Copyright Notices

© 2003-2007 Hewlett-Packard Development Company, L.P.

No part of this document may be copied, reproduced, or translated into another language without the prior written consent of Hewlett-Packard Company. The information contained in this material is subject to change without notice.

This product includes software developed by the Apache Software Foundation (<http://www.apache.org/>). Portions Copyright (c) 1999-2003 The Apache Software Foundation. All rights reserved.

Select Identity uses software from the Apache Jakarta Project including:

- Commons-beanutils.
- Commons-collections.
- Commons-logging.
- Commons-digester.
- Commons-httpclient.
- Element Construction Set (ecs).
- Jakarta-poi.
- Jakarta-regexp.
- Logging Services (log4j).

Additional third party software used by Select Identity includes:

- JasperReports developed by SourceForge.
- iText (for JasperReports) developed by SourceForge.
- BeanShell.
- Xalan from the Apache XML Project.
- Xerces from the Apache XML Project.
- Java API for XML Processing from the Apache XML Project.
- SOAP developed by the Apache Software Foundation.
- JavaMail from SUN Reference Implementation.

- Java Secure Socket Extension (JSSE) from SUN Reference Implementation.
- Java Cryptography Extension (JCE) from SUN Reference Implementation.
- JavaBeans Activation Framework (JAF) from SUN Reference Implementation.
- OpenSPML Toolkit from OpenSPML.org.
- JGraph developed by JGraph.
- Hibernate from Hibernate.org.
- BouncyCastle engine for keystore management, bouncycastle.org.

This product includes software developed by Teodor Danciu (<http://jasperreports.sourceforge.net>). Portions Copyright (C) 2001-2005 Teodor Danciu (teodord@users.sourceforge.net). All rights reserved.

Portions Copyright 1994-2005 Sun Microsystems, Inc. All Rights Reserved.

This product includes software developed by the Waveset Technologies, Inc. (www.waveset.com). Portions Copyright © 2003 Waveset Technologies, Inc. 6034 West Courtyard Drive, Suite 210, Austin, Texas 78730. All rights reserved.

Portions Copyright (c) 2001-2005, Gaudenz Alder. All rights reserved.

Trademark Notices

UNIX® is a registered trademark of The Open Group.

This product includes software provided by the World Wide Web Consortium. This software includes xml-apis. Copyright © 1994-2000 World Wide Web Consortium, (Massachusetts Institute of Technology, Institute National de Recherche en Informatique et en Automatique, Keio University). All Rights Reserved. <http://www.w3.org/Consortium/Legal/>

Intel and Pentium are trademarks or registered trademarks of Intel Corporation in the United States, other countries, or both.

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

BEA and WebLogic are registered trademarks of BEA Systems, Inc.

VeriSign is a registered trademark of VeriSign, Inc. Copyright © 2001 VeriSign, Inc. All rights reserved.

Support

You can visit the HP OpenView Support web site at:

www.hp.com/managementsoftware/support

HP OpenView online support provides an efficient way to access interactive technical support tools. As a valued support customer, you can benefit by using the support site to:

- Search for knowledge documents of interest
- Submit and track support cases and enhancement requests
- Download software patches
- Manage support contracts
- Look up HP support contacts
- Review information about available services
- Enter into 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 sign in. Many also require a support contract.

To find more information about access levels, go to:

www.hp.com/managementsoftware/access_level

To register for an HP Passport ID, go to:

www.managementsoftware.hp.com/passport-registration.html

Contents

1	About this Guide	13
	Structure	13
	Audience	13
	Assumptions	13
	Getting Help	14
	Other Select Identity Documentation	15
2	Designing and Planning Your Solution	17
	An Overview of HP's Three-Phase Methodology	17
	Project Initiation	18
	Project Definition	18
	Prioritize Results	19
	Define Scope of Work	19
	Existing versus Improved Business Processes	19
	Design and Planning	19
	Introduction	19
	Task 1 - Perform Detailed Discovery	20
	Detailed Discovery Approach	20
	Detailed Discovery Process	20
	Analyze Current Processes and Environment Factors	20
	Define the Requirements for Your Solution	20
	The Detailed Discovery Summary	22
	Task 2 - Determine Architecture and Design	22
	Hardware Architecture and Design	23
	Software Architecture and Design	23
	Application Server Architecture and Design	23
	Database Architecture and Design	23
	Connectors Architecture and Design	23
	Network Architecture and Design	24
	Administrative Roles and Services Architecture and Design	24
	Audit and Reporting Architecture and Design	25
	Backup Architecture and Design	25
	Disaster Recovery Architecture and Design	25
	Release Planning Architecture and Design	25
	Setup Architecture and Design	26
	Task 3 - Define Services and Processes	27
	Select Identity Services Overview	27
3	Developing and Building Your Solution	31
	Overview of the Develop and Build Phase	32

Development	33
Developing Connectors and Agents	33
Developing External Calls	34
Installation	34
Establishing Connectivity	35
Connector Types	35
Installing Connectors	37
Defining Resources and Attributes	37
Adding New Select Identity Attributes	37
Creating Resources in Select Identity	38
Removing a Resource.	41
Configuring Parameters	42
Configure Workflows.	42
Workflow Studio Overview	43
Workflow Templates in Select Identity	43
Set Up Configuration Approval	44
Approval Workflow	44
Roles and the Configuration Approval Process	45
Configuration Approval Workflows	45
Default Configuration Workflow.	46
Custom Configuration Workflows.	46
Configure External Calls	46
Configure Notifications.	46
Notification Variables	46
Set the Challenge and Response Questions	48
Create Services	48
The Service Function in Select Identity.	49
Steps to Creating a Service.	49
Important Tips.	52
Create Administrative Roles	52
Administrative Functions and Actions.	52
Creating and Managing Administrative Roles	53
Tips.	53
Delegating an Administrative Role	54
Deleting an Administrative Role	54
Establish Auditing and Configuration Reports.	54
Audit Reports	54
Configuration Reports	55
Scheduling Reports	56
Understanding Report Parameters	57
Set Up My Identity User Tasks	57
Profile-related Tasks	57
Password-related Tasks.	59
Service-related Tasks	59
Set Up Self-Registration.	59
Configuring the Self-Registration Form.	59
Setting the Self-Registration URL	60

Data Processing	62
Overview	62
Process Summary	63
Obtain Raw Data	63
Pre-process the Data	64
Configure and Test User Import and Service Assignment	64
Creating an SPML File Containing Users and Attributes	64
Creating an SPML File Containing Entitlements	66
Uploading User Accounts, Attributes, and Entitlements	67
Scheduling User Import	67
Service Assignment	67
Checking for Service Membership Requirements	68
Schedule Services Assignment	68
Set Up Multiple-User-ID Accounts	68
Primary Accounts	69
Secondary Accounts	69
Creating Multiple-User-ID Accounts	70
Example of a Multiple-User-ID Account	70
Importing and Updating Multiple-User-ID Information	70
Examples	72
Bulk Operations	74
Bulk Dependencies	74
Bulk Procedure Overview	75
Upload Data Files	79
Viewing Job Results	79
Scheduling and Managing Bulk Jobs	79
Bulk Move	79
Updating Multiple-User-ID Accounts through Bulk Add	81
Set Up Service Reconciliation	82
4 Testing and Deploying Your Solution	83
Overview of the Test and Deploy Phase	83
Testing and Deployment Preparation	84
Testing	84
Integration Testing	84
User Acceptance Testing	84
Load and Performance Testing	84
Disaster Recovery Testing	85
Deployment Preparation	85
Documenting the Project	85
Training	86
Deployment into Production	87
Configuration Replication	87
Exporting Configurations	88
Importing Configurations	89
Secure Migration	91
Troubleshooting Configurations	92
Set Up Users	92

Conduct Final System Check	93
5 Configuring and Testing User Account Reconciliation	95
Reconciliation Overview	96
Service Membership Requirements	96
Resource Provisioning During Reconciliation	97
System Configuration Prior to Reconciliation	97
Reconciling with Authoritative and Non-Authoritative Resources	98
Reconciling with an Authoritative Resource	98
Reconciling with a Non-Authoritative Resource	99
Designating Attributes as Sync In or Sync Out	101
Writing Reconciliation Rules	102
Defining Your Reconciliation Policy	102
Attribute-Level Reconciliation Policy	103
Resource-Level Reconciliation Policy	103
Reconciliation Filter	104
User Polling	104
Add, Modify, Delete	104
Multiple-User-ID Accounts and User Reconciliation	106
Authoritative and Non-Authoritative Resources and Multiple-User-ID Accounts	107
Workflow and Multiple-User-ID Accounts	107
Scheduling and Managing Reconciliation Jobs	107
Preparations	107
Using an Agent or Web Service Interface	108
Task Status	108
Testing Reconciliation Prior to Production	108
Processing the SPML Data File	109
Understanding Dependencies	109
Resource Names	110
Creating an SPML File Containing Users and Attributes	110
Writing SPML	110
SPML Tips	111
SPML Examples	112
Web Service Reconciliation SOAP Request with Encrypted Attributes	114
Specifying Attributes in SPML	115
Creating an SPML File Containing Entitlements	115
Application Server Properties	116
Evaluating Reconciliation Job Results	119
Priority of Policies	119
Expected Policy Responses for an Authoritative Resource	119
Expected Policy Responses for a Non-Authoritative Results	120
Troubleshooting Reconciliation Jobs	122
Issues common to all reconciliation methods	122
Job Submission and Execution	123
Administrative Role and Context	123
File Upload	123
Web Service and Polling	124

Request Processing	124
SPML Data: XML Format	124
SPML Data: Fields	125
SPML Data: Operations	126
Rules: General	126
Rule: Actions	126
Engine Process: User Attribute Analysis	127
Policy Evaluation	128
Operation Analysis and Select Identity Request Composition	129
Workflow: Design and Selection	130
Workflow: Execution	131
Approval	131
Resource and Select Identity Provisioning	131
Reporting	132
Format and Contents	132
Generation and Email Notification: Type and Time	133
Generation and Email Notification: Web Service Report	133
General Error Handling	134
Performance and Configuration Tips	134
Configuration	134
Application Server/Cluster and Database Server	135
6 Recovering from Abnormal Server Shutdown	137
Understanding the Server Work List	137
Managing Requests	137
Recovering Requests in the Event of Server Failure	138
Locating Requests for Recovery	138
Recovery Procedures	138
Recovering Delegated and Self-Service Requests	138
Recovering User Reconciliation Requests	139
Recovering Bulk Add and Move Requests	139
Recovering Service Reconciliation Requests	140
A Creating Select Identity Rules	141
Understanding User Status Dependencies	141
New Users	141
Reconciliation Rules	142
Tips	142
Complete DTD Rule Definition	143
Managing Rules	146
Troubleshooting Reconciliation Rules	146
Exclusion Rules	147
Service Exclusion Rule	147
Attribute Exclusion Rule	147
Entitlement Exclusion Rules	148
Sample Rules	148
Rule Standards and ServiceNameMap	149
Sample Rule One	149

Sample Rule Two	153
B The SPML Generator Utility	155
SPML Generator Utility Package	155
Understanding Dependencies	156
Running the Utility	156
Properties Descriptions for the Properties Configuration Files	157
C Attribute Mapping Utility	163
Attribute Mapping Utility Overview	163
Accessing the Attribute Mapping Utility	164
Using a JDBC driver without an agent installed	164
Using a JDBC driver with an agent installed	165
Configuring the JDBC Datasource for a Connector	165
Attribute Mapping Utility Menus and Pages	166
Attribute Mapping Utility Menus	166
Mapping Pages	167
Attributes Home (Page 1)	168
Mapping Attributes to Select Identity	168
User Enable/Disable (Page2)	168
Specify Supported Operations (Page 3)	168
Define Relationship Definitions (Page 4)	169
Reverse Synchronization Attributes (Page 5)	169
Provisioning Information page	169
Usage Scenarios	170
Table Scenario	170
Stored Procedure Scenario	170
Defining User Mappings	171
Defining Entitlement Mappings	175
Provisioning Entitlements in the Database	177
Mapping Stored Procedure Parameters	177
User Tables and Stored Procedures Scenarios	180
One User Table	180
One User Table, One Entitlements Table	180
One User Table, One Entitlements Table, One Map Table	181
Multiple User Tables	181
Two User Tables, One Entitlements Table	181
Two User Tables, One Entitlements Table, One Map Table	182
Multiple User Tables, Multiple Entitlement Tables	182
Single Stored Procedure	183
Multiple Stored Procedures	183
Stored Procedure for Attributes	183
Tables and Stored Procedures	183
D Auditing XML and Client Sample	185
Processing the Audit XML Stream into a Database	185
Using the Audit Client	185
Configuring Connection Properties	186

Running the Audit Client.....	186
The Select Identity Audit XSD	186
E External Calls	197
Default External Calls	198
Creating an External Call For Workflow Templates	198
Creating an External Call for Attributes	199
Deploying an External Call.....	199
Glossary	201
Index	215

1 About this Guide

Welcome to the *HP OpenView Select Identity Administrator's Guide*.

This document is here to help you learn and use HP OpenView Select Identity. From it, you can discover the conceptual foundations of Select Identity, and master the skills you need to get the most from Select Identity.

However, this guide does *not* include detailed procedures that involve the Select Identity graphical user interface. That information is in the extensive online help system, described below.

Structure

The *HP OpenView Select Identity Administrator's Guide* describes how to create an identity management solution using Select Identity. Generally speaking, you will find conceptual information at a very detailed level here. You will be appropriately directed to specific procedural information that is contained in the online help system.

Some tasks are performed outside of the Select Identity graphical user interface, and in those cases, this section provides detailed procedural information and examples.

Audience

This guide is not for everyone. It was written with the following kinds of readers in mind:

- Identity management design architects, who need to fully understand specific detail necessary for a smooth and effective implementation.
- System administrators and business unit administrators, who have been delegated the authority to extend Select Identity within specific limits.
- User administrators, who are charged with the daily task of adding, changing, and removing users.
- Help desk users who want to upgrade their understanding of Select Identity and prepare for greater responsibility.

Assumptions

You should be aware that this document makes several important assumptions about you and your situation:

- This document assumes you have finished the following tasks, according to the *HP OpenView Select Identity Installation and Configuration Guide*:
 - Install and test your web services platform (WebLogic or WebSphere).
 - Install and tune the database you will use.
 - Install and optimize Select Identity.
- Select Identity administrators and deployment personnel should be comfortable reading and editing XML and SPML.
- Some users of Select Identity will need expertise in programming with Java and potentially other languages.
- You should be comfortable using your local operating systems and their utilities, such as the file system, and a plain text editor.

Getting Help

Select Identity comes with an extensive help system, including the following major subject areas:

- Administrator Online Help
- Workflow Studio Online Help
- My Identity Online Help

Other Select Identity Documentation

This is not the only source of Select Identity information. You can find all of the following documents on the Select Identity media:

- *HP OpenView Select Identity Installation and Configuration Guide* - Tells you how to install and set up HP OpenView Select Identity. Includes all available platform and migration documentation.
- *HP OpenView Select Identity Guide to Concepts* - Provides conceptual, architectural, and job-function perspective to help you understand the nature and application of Select Identity.
- *HP OpenView Select Identity Administrator's Guide* - Provides detailed information on how to design, implement, and deploy an identity management solution using Select Identity.
- *HP OpenView Select Identity Web Services Developer Guide* - Describes the Web Services SPML elements and explains how you can use them.
- *HP OpenView Select Identity Connector Developer Guide* - Describes how you can develop custom connectors.
- *HP OpenView Select Identity External Calls Developer Guide* - Describes how you can create, register, and use external calls.
- *New Information about HP OpenView Select Identity* (if provided) and the Release Notes accompanying the software - Contains late-breaking information developed after the above were finished.

2 Designing and Planning Your Solution

This chapter and the two that follow provide a step-by-step guide for designing, building, and deploying an HP OpenView Select Identity solution, using a three-phase methodology presented in this chapter as a framework.

This chapter, [Designing and Planning Your Solution](#), deals with the first phase of this methodology. [Chapter 3, Developing and Building Your Solution](#), covers the second phase. [Chapter 4, Testing and Deploying Your Solution](#), details the third and final phase.

▶ The approach described in this chapter and the next two was developed and is used by the HP Consulting and Integration team. This material is based on the insight HP has acquired through extensive experience implementing identity provisioning solutions for clients worldwide.

However, this guide does not capture the full depth and breadth of knowledge that is available from the HP Consulting and Integration organization. Contact your HP representative for information about engaging their services.

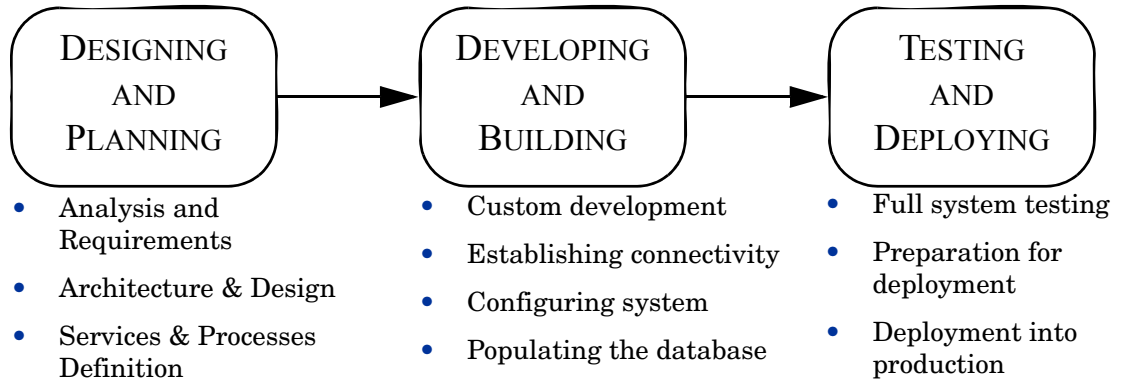
The first section of this chapter provides an overview of the methodology, and includes a brief discussion of two preliminary phases.

An Overview of HP's Three-Phase Methodology

[Figure 1](#) illustrates a recommended methodology for implementing an identity provisioning solution using Select Identity. It was derived from HP's experience developing identity provisioning solutions for clients worldwide. Following the methodology outlined in this chapter, and in [Chapter 3](#) and [Chapter 4](#), will help to ensure a successful project.

▶ Before you design and implement your Select Identity solution, you should read the *HP OpenView Select Identity Guide to Concepts* to fully understand this methodology.

Figure 1 The Three-Phase Methodology



Each phase requires different kinds of work, and often different skill sets. Few if any identity provisioning solutions are accomplished without a talented and well-motivated team of people with diverse and complementary skills.

Even before the “Designing and Planning” phase begins, there is early groundwork to be done, to set the stage for a successful project. These preliminary steps, described next, are called “Project Initiation” and “Project Definition”.

Project Initiation

The *Project Initiation* preliminary phase consists of meeting with key stakeholders, to get them involved in the project. Stakeholders are those who are directly affected by the Select Identity solution you design. Their input and feedback will be important to your design and throughout your project.

Your approach to an identity provisioning solution should be to create and use shared services that are scalable and sustainable, and this involves many different people in the organization.

Project Definition

This phase consists of organizing your project, and determining the scope of work you want to do. One good approach is to have a requirements meeting with key stakeholders. From this, you can capture the high-level data and business requirements for your project.

Be sure to include people who participate in the business processes being examined, and who understand the business well enough to contribute effectively. The focus of such a meeting should be on the business requirements – on *what*, not *how*.



Some familiarity with Select Identity will help your meeting members better set project objectives. In particular, it would be helpful for them to review the *HP OpenView Select Identity Guide to Concepts*.

In addition to a requirements meeting, one-on-one interviews with participants in the business process are also very useful. Remember to engage business people as well as technical people.

Prioritize Results

After you have gathered the requirements data, you need to prioritize the results to determine which requirements should be included in your identity provisioning solution. Regardless what criteria you choose, you must sort and prioritize the requirements before you can begin to define the scope of the project.

Define Scope of Work

Identity provisioning projects are very complex. Manageable scope is difficult to define and maintain. The project will cross organizational and departmental boundaries, and may become larger than originally estimated. You must plan and prepare accordingly. Remember, too, that identity management is generally not a one-time-done effort. You may want to think about setting a long-term strategy, implemented in a series of discrete projects.

As you go forward, be aware that “creeping improvements” can add considerable time and effort to your project. Try to keep to your original plan whenever possible, and keep a list of future enhancements.

Existing versus Improved Business Processes

A good identity provisioning solution depends on good business processes. Work with your organization to define all the business processes that are in place and working today. At this point, your objective is to look at the whole organization, and find the boundaries between various processes, so that your solution matches your business processes.

Next, identify opportunities for improvement of the business processes, such as simplification, automation, alignment, and so forth. Different viewpoints are extremely valuable, so talk with others to find opportunities.

Make sure you understand any dependencies between adjacent units and processes, such as HR, payroll, IT service management, and so forth.

Design and Planning

Introduction

In this first phase of the process, you will begin the design and planning of your Select Identity solution. There are three major tasks in this phase:

- Task 1: Detailed Discovery
- Task 2: Architecture and Design
- Task 3: Processes and Services Definition

Each one of these steps is detailed in the sections that follow.

Task 1 - Perform Detailed Discovery

The objective of this first step is to develop a realistic and finite list of requirements to serve as the basis for subsequent design efforts.

Detailed Discovery Approach

As part of your work in the preliminary project definition phase described above, you identified stakeholders. Now meet with them again, in groups or singly, and interview them in depth. Try to interview representatives from each affected business or process, as well as IT security, IT operations, the help desk, training, and human resources. Gather as much information as possible in the beginning, during this design and planning phase, and resolve issues immediately. You don't want to rework your design later to address unresolved issues.

Detailed Discovery Process

Here is a recommended approach for performing the detailed discovery:

- 1 **Analyze current situation** - Gather as much information as possible about the current processes and environment. Try to identify gaps.
- 2 **Gather requirements** from as many sources and from as many different viewpoints as possible. If someone presents an idea that seems to be more of a design than a requirement, see he or she can describe *what* they want to accomplish instead of *how* to accomplish it. Also watch for hidden assumptions. A high-level term such as "employee promotion" has a common meaning, but you need to know exactly how and when that action is performed.
- 3 **Finalize requirements** - Organize, consolidate, rationalize, and document these requirements. Break requirements down into the most basic elements possible. This helps you discover and resolve conflicting requirements.
- 4 **Review the requirements document and gain formal acceptance** - Review the draft requirements document with key stakeholders. Make sure everyone accepts and approves the requirements document before you proceed.

Analyze Current Processes and Environment Factors

You may already have multiple systems and applications relying on identity information, and processes for managing that information. These processes may be manual, semi-automated, or automated. You may already have active identity management solutions running. You must define and document all of this information. This information is an important baseline for planning the required steps to achieve the best outcome.

Start your analysis by reviewing existing use cases. Try to define the processes that are in place and their purposes. Identify when a process gets triggered, and how, and what its parameters are.

In addition, document all aspects of your current environment, including applications data stores, hardware, and so on.

Define the Requirements for Your Solution

To complete this step, you need to answer define and document the following requirements:

- **Business requirements** – the business rationale for the solution

- **Functional requirements** – how the solution will work
- **Technical requirements** – the way the solution will be implemented
- **Data requirements** – the sources of data for the solution
- **Operational requirements** – how the solution will be built and maintained

Business Requirements

The business requirements should define the business rational for implementing an identity provisioning solution. Here are some example business requirements:

- “Eliminate all paper forms used to request a new e-mail ID”
- “Automatically request a network ID for all new hires entered into the HR system”

Business requirements commonly stem from problems that need to be solved, such as meeting specific regulatory/audit requirements. They are typically couched in non-technical terms.

Functional Requirements

Functional requirements determine how the system will work. That is, they describe the expected solution. The functional requirements describe everything that your solution will do (but now *how* it will be achieved). Functional requirements include detailed information of several kinds:

- **Who** — the people who will request, approve, or be affected by a function, and the surrounding security concerns
- **What** — the data, forms, policies, and so on that are part of a function
- **Where** — data stores, applications, systems, external resources, and so on that are involved
- **When** — the circumstances under which the function is executed
- **Why** — the business requirement that the function addresses
- **How** — the interfaces, workflows, administration, and so on of the function, as well as the failure modes and recovery

Remember to deal with outside events. For example, what, if anything, happens in Select Identity when identity information is changed on a target system?

Technical Requirements

You must discover the technical requirements of your Select Identity-based solution. Technical requirements fall into four areas:

- Hardware and software
- Network
- Connectors
- Standards (including security and audit)



Avoid technical differences between development, test and production environments. Technical differences can cause late-cycle problems.

Data Requirements

The data requirements determine which set of data will be used by Select Identity. First, identify all authoritative data sources. There are three potential sources of authoritative data:

- **Human Resources** is typically the main authoritative data source for general user data such as name, employee ID, address, department, and so forth.
- **Target systems** that will be provisioned to Select Identity may have some specific data attributes that are authoritative. For example, each target system may have a local application ID that is unique to that application.
- **Other local applications** that are not being provisioned to may also have some specific data attributes that are authoritative. For example, an e-mail system could be the authoritative source for managing user e-mail IDs.

You need to discover what the primary keys are for each authoritative data source.

Operational Requirements

The operational requirements determine how the solution will be managed, particularly how the solution will be migrated into production and then maintained. Discovery should focus on defining the following activities:

- Functional testing, to assure solution quality and capability
- Performance testing, to stress the solution and certify capacity
- Migration, to assure smooth transition from development to production, and ongoing maintenance
- Disaster recovery and backup, to assure continuity of operations

The Detailed Discovery Summary

Compile and summarize the documents you developed throughout your detailed discovery process. This summary report will form a platform for discussion as you begin detailed design.

Task 2 - Determine Architecture and Design

Once the requirements have been established, you need to translate them into a detailed design. First, focus on the design and definition of the operations and physical aspects of the Select Identity architecture. You will later design the services component of the Select Identity.

In this step, you address the following components of the physical and operations architectures:

- Physical architecture, including the hardware, software, application and database servers, and so on.
- Operations architecture, including your services, administrative roles, user setup, and so on.

This task has two steps:

- 1 **Create a detailed design** - Create and document a detailed design for both the above components.
- 2 **Define your solution architecture** - Define and document the architecture for each of the above components.

The remainder of this section discusses the objectives, recommended approach, and practical tips for successfully completing tasks 1 and 2 of this step for each of the physical and operational architectural components listed above.

- ▶ Hardware software, application server, and database specifications are all driven by what the Select Identity product supports. This information can be found in *HP OpenView Select Identity Installation and Configuration Guide*.

Hardware Architecture and Design

As usual, separate application servers and database servers whenever possible. Factor in hardware associated with disaster recovery needs, and consider the extra load on Select Identity from reconciliation.

Software Architecture and Design

Check compatibility between the software operating system and all other software components. Also, be aware of internationalization and localization requirements. Select Identity supports internationalization for languages supported by the Java unicode specification.

Application Server Architecture and Design

Be sure to document the configuration of all Select Identity application servers used in your solution. Whenever possible, use Select Identity scripts for starting and stopping the application server, instead of the native interface. They were created specifically for Select Identity.

Database Architecture and Design

Document the assignment of the database (including the version used) to all database servers in all technical environments. Make sure your databases are designed and configured for high availability and optimization, as this will affect Select Identity's performance.



Do not manipulate data at the schema level, such as deleting records from tables, modifying indexes, restoring individual tables and so forth. Operating at the native schema level will produce unpredictable results, and will likely leave the entire Select Identity installation in an unstable state.

Connectors Architecture and Design

You need to determine which connectors can be used from the library of standard connectors provided with the Select Identity product, and which connectors need to be developed. Use standard Select Identity connectors wherever possible. Validate that the standard connector provides the functionality required, and verify that there version support on Select Identity and the application server.

When using custom connectors, translate the detailed discovery business requirements into a set of custom connector requirements, and formulate an approach for how to develop the custom connector.

Make sure you validate all connector functionality. Forward provisioning may be functional, but other requirements, such as the ability to set password expiration, may not be.

Network Architecture and Design

The physical layout of your Select Identity components is greatly influenced by your disaster recovery strategy.

First, inventory all previously identified hardware components, capturing or assigning physical network data for them, such as hostname, IP address, and so on. Note additional information about each hardware component, such as its purpose in your solution, and its current configuration. After diagramming this information, decide on load balancing strategies for various components.

Administrative Roles and Services Architecture and Design

The design and architecture of administrative roles and services establishes a set of roles with varying administrative and functional privileges within Select Identity. It includes designing the business processes for managing the assignment of users to these roles.

- 1 Design a business service based on your detailed discovery requirements.
- 2 Using its workflow as a guide, look for all possible participants (except the target user) in the workflow. Each of these participants will be an administrative role.
- 3 Use the detailed discovery requirements as a guide to identify who should have the ability to modify various aspects of this business service (for example, change the workflow, modify service attributes and so forth.). In many cases, one administrative role will be defined that has control over all aspects of the business service. Sometimes, however, that control is sub-divided and spread across multiple administrative roles. Finally, consider which users you would expect to be assigned to each administrative role. [Figure 2](#) on page 24 illustrates one way to organize the information.

Figure 2 Administrative Roles Chart

Role Name	User Management	Request Status	Approval	Service Management	Attribute Management	Forms Management	WorkFlow Management	Notification Management	Report Management
SISA (Select Identity System Admin)	x	x	x	x	x	x	x	x	
SysAdmin (System Administrator Role)	x	x		x	x	x	x	x	
Del_Admin (Delegated Administrator Role)	x	x							
HR_Approver (Designated HR Approver Role)		x	x						
App_Approver (Application/Data Owner Approver Role)		x	x						
Audit_Admin (Audit Administrator)									x
Services_Dev (Services Developer Role)				x	x	x	x	x	
Services_Admin (Services Administrator Role)				Modify Service Modify Service Attribute Values and Properties Create/Modify Service Forms Create/Modify Service Roles Create/Modify Service Context					
SI_User (Select Identity User)	Add Service Membership Reset Password Modify profile data (Limited)	x							

- 4 Continue designing all other business services. Reuse existing administrative roles where possible. When you have identified all administrative roles, see if some of them can be consolidated (that is, used across more than one business service).

Administrative services are used to assign users administrative roles. To design an administrative service, follow the same steps as for business services.

When users are assigned to an administrative service, make sure you specify which service they can manage, and determine within which context they can manage a service.

Audit and Reporting Architecture and Design

When designing the Select Identity audit and reporting functionality you want to provide transactional and snapshot information about Select Identity operations. Select Identity includes a set of standard audit and reporting features. These are as follows:

- Transaction Logs — low-level detail, primarily for debugging purposes; stored on file system
- Audit Reports — pre-configured reports on transactions over time; stored in the Select Identity database
- Configuration Reports — pre-configured reports containing a snapshot of current data; stored in the Select Identity database

Make sure you establish and document the parameters for transaction logging in production. Refer to the *HP OpenView Select Identity Installation and Configuration Guide* and [Establish Auditing and Configuration Reports](#) on page 54 for more information.

Backup Architecture and Design

Select Identity does not enforce any specific backup procedure. Your solution needs a backup procedure for recovery purposes. There are two levels of backup in a Select Identity solution:

- File system level, such as installation files, property files, and so on.
- Database level, such as user data and Select Identity configuration data.

When you design your backup procedures, work with your IT operations team to design the file system backup procedures. Also, work with your database team to align the Select Identity database backups with current practices.



Because Select Identity configuration information is stored in the database, you can periodically export the configuration to SPML files, which are stored on the file system and backed up at that level. Restoration of a specific configuration item is easier if you can isolate a specific SPML file to re-import, instead of having to restore the entire database. See [Configuration Replication](#) on page 87.

Disaster Recovery Architecture and Design

Your solution should have availability and reliability built into the system. As usual, avoid single points of failure. Plan to set up redundant hot standby servers, preferably at a distant site. Of course, regular data backups are pivotal.

Release Planning Architecture and Design

Release planning defines your a road map for the phased deployment of Select Identity. Think about which services to deploy and when, to cause the least disruption to business operations. This may be different for different user communities, and be influenced by compliance and regulatory schedules.

For business requirements that cannot be met in the current state of the product, maintain a “deferral list”. This list, along with future requirements, can be the starting point for planning the next phases.

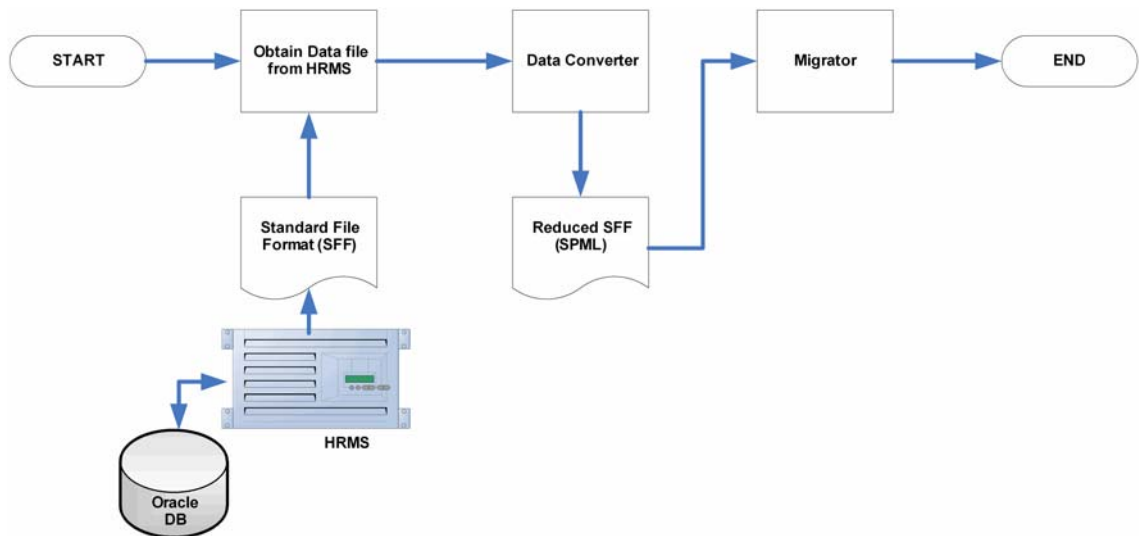
- 🚩 **Keep project scope manageable** - Minimize your risk. Aim for what you can accomplish in a few months. Anything more is a major project and likely to have too much complexity and risk. It is much less risky to break a large project up into smaller sized increments.

Setup Architecture and Design

Now you need to decide how to initially populate Select Identity with a set of users from the existing infrastructure when the solution is ready for production deployment.

These users are from existing data repositories, which you should have identified in the discovery phase. The user import function in Select Identity requires input data to be in SPML format. Since most information systems today are incapable of producing SPML data extracts, in most cases, you should to plan to convert your existing data into SPML format. [Figure 3](#) shows a typical process for converting data into SPML format.

Figure 3 Typical Process for Converting Data into an SPML File



You should have already determined, during detailed discovery, the systems that need data pre-loaded into Select Identity, and in what format the data extract can be made available.

Your design for user import should define the following for each system:

- The data extract file format and legend for each field
- The field from the data extract that will serve as the primary key
- The kind of user accounts from the data extract that will be processed. Some accounts will probably be excluded for example, root accounts.
- The fields from the data extract that will be processed

Service assignment follows user import. Because of this, all required attributes for each service should be designed in your setup plan.

You may need to integrate multiple existing user repositories (for example, HR, AD, Finance). There is possibly no common primary key across repositories. For example, how do you match “Cathy Müller” in HR with “cmuller” in AD and “Muller, Cathy A.” in Finance? If there is no primary key across the repositories, you have the following options:

- Match on a different field that may not be a key field in either repository, but is unique and exists in both repositories.
- Combine, truncate, append or otherwise manipulate one or more fields to create a new field that contains the desired key value.
- Expose the desired key field and make it available in the data extract file. It is possible that this might be the only option.

This issue is easily identified, but it may require program changes on your legacy systems to resolve the issue.

Many systems have lax controls on old accounts, orphan accounts, duplicate accounts and so forth. Some accounts may not even be a true user, but may be a physical meeting room, a printer or even a technical account, for example, “testuser1”. Some fields may use different spellings in one repository versus another for common values, or even different character sets (for example, Müller vs. Mueller). As a result, you may need to perform manual data cleansing.

Task 3 - Define Services and Processes

At this point in your project, you are ready to translate the business service and process requirements discovered earlier into a detailed design of the services in your solution.

Select Identity Services Overview

In its common usage, the term **service** often means a single application. Users subscribe, or are assigned, to this kind of service. Lotus Notes, a Customer Relationship Management (CRM) system or a Benefits Management system are examples of this kind of service.

Users are granted specific access to components of these services. Examples could include access to discussion groups, online account management, or payroll and stock option plan management. Access rights to various components are set according to institutional standards. Each set of pre-defined access rights is called an **entitlement**.

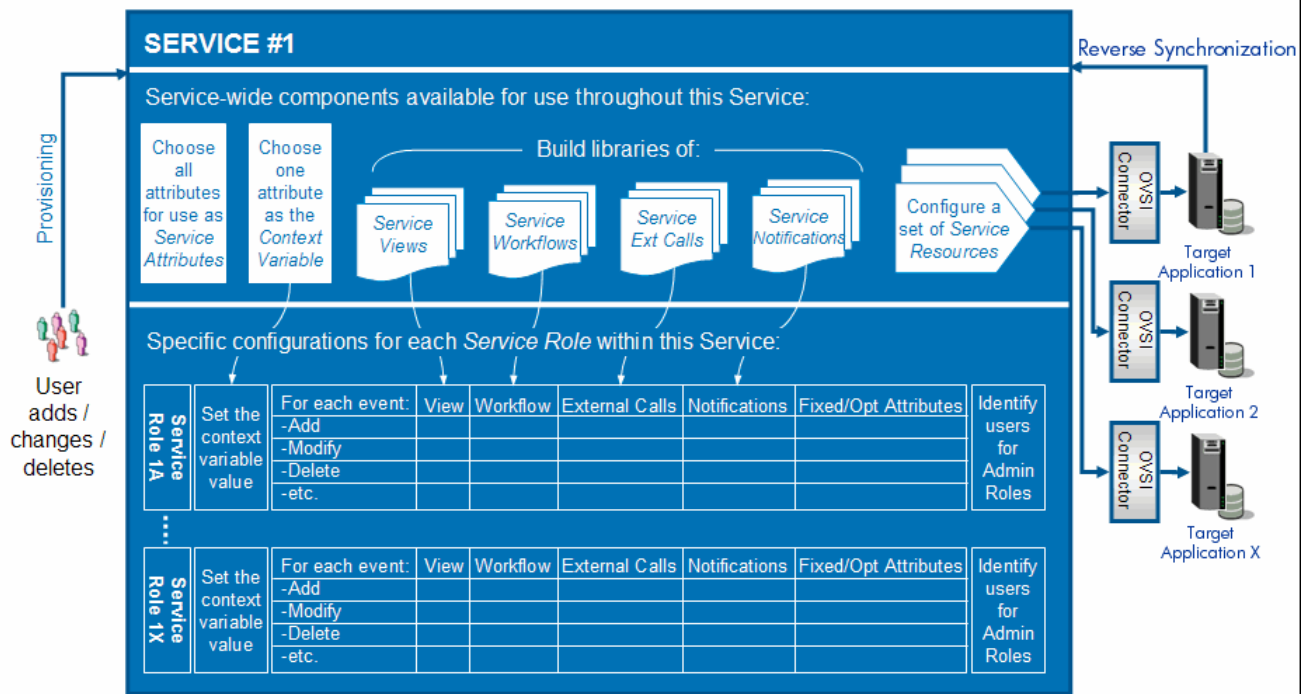
Within each of these services, there are underlying procedures, such as the following:

- How to submit an application to request membership
- How to check credentials for qualification
- How to approve requests
- How to modify access, terminate membership, and so forth.

When you design a service in Select Identity, you must decide how to set up the standard Select Identity service-wide components to meet the business and process requirements that you discovered in the earlier steps.

Figure 4 shows a blueprint for a typical Select Identity business service. The major sub-components of the service are listed across the top. The lower half of the blueprint depicts the configurations for each service role associated with the service. The design of the sub-components and service roles make up the bulk of this section.

Figure 4 Select Identity Business Service Blueprint



* These are the major sub-components only; others, such as primary key, service description etc. are not shown.

When designing your Select Identity service, keep in mind that each service can have reusable components available for use throughout the service. These include:

- Reusable request forms (service forms)
- Reusable approval processes (workflow templates)
- Reusable e-mail templates (also known as notification templates)

Also keep in mind that service roles are directly associated with a context or context value. For example, to get benefits, a user must qualify for benefits based on the region he or she belongs to. Each service role is set up using the reusable components above, and determines which service forms, approval workflows, notifications, and so forth are used.

Steps To Success

Thoughtfully following these steps should lead you to a good overall design of your Select Identity services:

- 1 Define service attributes
- 2 Select the context variable
- 3 Define a library of service forms
- 4 Define a library of workflows
- 5 Define external calls
- 6 Define notifications
- 7 Define a set of resources
- 8 Define service roles
- 9 Design reconciliation processes

10 Organize multiple services



This is a complex step in your project. It is also critically important, and well worth spending time on. Don't let impatience drive you to implementation too soon.

HP offers an HP OpenView Select Identity Solution Consulting training course that addresses each of the Select Identity service components depicted in [Figure 4](#) in terms of the following:

- What does the sub-component do and why is it necessary?
- How to derive the component.
- Practical tips, best practices, design considerations, and lessons learned about designing this component.

Contact your HP account representative to obtain more information about this training course.



Details on how to configure service components are covered in [Chapter 3, Developing and Building Your Solution](#).

3 Developing and Building Your Solution

The methodology presented in [Chapter 2, Designing and Planning Your Solution](#) has three major phases:

- 1 **Design and plan**
- 2 **Develop and build**
- 3 **Test and deploy**

This chapter explores the second phase. This is where most of your interaction with Select Identity takes place during implementation.

In performing the planning and review activities in [Chapter 2](#), you should have generated several important documents:

- From the discovery phase, you should have a document that clearly describes the current state of your identity management processes and the requirements you have for its replacement. Part of this should describe your best estimation for requirements that are likely to come into play in the future.
- From the design phase, you should have detailed design documents. These should address the physical, operational, and services architectures of the final solution, and describe, in full detail, the processes and services of your solution. Your design documents should cover all of the following:
 - Diagrams of your business and IT services
 - Service attributes of each service
 - Context variables of each service
 - A library of service views defined for each service
 - A library of service workflows
 - Specifications for the external calls that need to be developed
 - A list of the notifications for each service
 - A list of the set of resources for each service
 - Specification of the service roles for each service
 - Reconciliation plans for each service
 - Statement how multiple services will be organized to provision multiple resources

In this phase, you will build any necessary extensions to Select Identity, and develop every aspect of your solution.

You are likely to find that your experience isn't as tidy as the steps in this chapter might suggest. It is normal to repeat a step many times with slightly different parameters, or even backtrack to perform more iterations of an earlier step. You may find that an initial assumption requires you to revisit some part of your plan, and rework some of your design.

However, by following the general order of work in this chapter, you can be confident that you are building a solution in logical stages.



If you don't yet have good planning documents, you should take time now to develop them. You want to begin your identity management project with a strong understanding of exactly what is needed, and why, and how it will be accomplished. This will ensure efficiency during the development phase.

Overview of the Develop and Build Phase

This section introduces the basic tasks for developing and building your solution, which later sections cover in detail. This phase consists of five distinct and substantial tasks:

- 1 Development** The objective of this task is to enhance Select Identity's functionality in accordance with your plan. Activities include customizing and/or creating connectors and external calls to resources and applications. See *Development* on page 33.
- 2 Installation** This task is explained fully in the *HP OpenView Select Identity Installation and Configuration Guide*. Also, see *Installation* on page 34.
- 3 Establishing Connectivity** The objective of this step is to define resources and determine rules. Activities in this task mainly involve installing and configuring the connectors supplied by Select Identity, and setting up the associated resources. See *Establishing Connectivity* on page 35.
- 4 Configuring Parameters** The objective of this task is to implement the design and configure it into the system. Activities include configuring Select Identity attributes, services, workflows, external calls, notifications, and so on. You should also establish auditing reports. See *Configuring Parameters* on page 42.
- 5 Data Processing** The objective of this task is to import existing data into Select Identity. Activities in this task include importing users, setting up reconciliation, and creating pre-processing scripts, SPML generators, and the like. See *Data Processing* on page 62

Again, these tasks may overlap during an actual project.

Development

Most projects involve at least one external call or customized connector. Some require more. In your planning phase, you should have identified what you need in this area.

There is a range of tasks involved in developing the extensions and enhancements to Select Identity that your plan calls for. You may need to do any or all of these:

- Implement new resource connectors for provisioning to target systems.
- Implement new resource agents for feeding information about changes back into Select Identity.
- Implement external calls to enhance Select Identity's functionality.
- Develop custom tools for data migration and data pre-processing (described more in detail in *Data Processing* on page 62).

You should look at each customized or newly created connector, or external call to a resource or application as a separate effort. Plan each one as a “mini-project” within your total solution. Each new connector or external call requires standard software practices:

- Unit testing
- Integration testing
- Defect fixing
- Regression testing
- System testing

This development effort for external calls and custom connectors can be substantial. You should make sure that you budget appropriate time and resources to the task.

Outsourcing any of this work is possible, though the results sometimes require extra integration effort.

Developing Connectors and Agents

You can build custom connectors, using a Java API provided by Select Identity. This lets you integrate Select Identity with resources that do not yet have a connector supplied by Select Identity.

See the *HP OpenView Select Identity Connector Developer Guide* and the *HP OpenView Select Identity Installation and Configuration Guide* for details. These documents are available on the Select Identity Connector media. They provide details about developing and deploying custom connectors, including an API overview, methods to be implemented using the API, packaging instructions, and an installation procedure.

Whenever possible, try reusing and modifying an existing connector, rather than starting from scratch. also, check whether the resource can be accessed using remote function calls (e.g. using LDAP or ADSI).

In addition, for resources that are not accessible remotely, you have to develop a connector to be deployed in Select Identity, and an agent to reside on the target system. You must ensure that communication between the two is reliable. In short, this requires considerable effort, though it may be necessary.



Note again that connectors can only provision to resources, and cannot detect changes on the resources. If you need to reconcile changes made on a resource back into Select Identity, there must be an agent installed on it that can communicate with both the resource and Select Identity.

Agents are available in Select Identity for a selection of target systems.

Developing External Calls

External calls can be used to perform the following types of tasks:

- Generate a user ID, password, email address, or attribute value.
- Provide a list of possible attribute values.
- Validate the value of an attribute.
- Verify the value of an attribute.
- Query an external system for a list of approvers.
- Set workflow variables to drive your workflow.
- Retrieve a certificate from an external system.
- To implement approver selections (e.g. “get direct manager approval”)
- To populate “list” or “drop down” box with data from an external source
- Alter an original request (reconciliation SPML request filter)
- Perform some direct operation to resource
- Update attributes during workflow

External calls typically are simpler than custom connectors to develop. In addition, data dependencies in external calls require closer interaction with your systems. Consequently external calls may be less attractive for outsourcing.

See [Appendix E, External Calls](#), for more details, and the *HP OpenView Select Identity External Calls Developer Guide* for complete information about external calls.

Installation



Many people install Select Identity first in a development environment, and later in a production environment. If that is your plan, try to simulate the same complexity in your development environment as in the production environment. This will minimize the effort required to put the solution into production later.

A typical installation process proceeds in a logical order, such as this:

- 1 Install the database server
- 2 Install the application server

- 3 Establish communication between the application server and the database
- 4 Execute the Select Identity database script
- 5 Install the Select Identity application
- 6 Patch the Select Identity application and database to the recommended level (deploy after the installation)

Full installation details can be found in the *HP OpenView Select Identity Installation and Configuration Guide*.

Establishing Connectivity

In this phase, you get Select Identity communicating with your resources. This mainly involves installing and configuring the predefined connectors within Select Identity, and any custom connectors you have developed. For each connector, you should follow this procedure:

- 1 Install the connector and check that it is communicating with the target system.
- 2 Configure the resource in Select Identity to use the connector.
- 3 Perform attribute mapping as stated in the design document.
- 4 Test that the resource in Select Identity is talking to the target system using the connector.
- 5 Implement authoritative resource or attributes according to your design.



Try to finish these steps with one connector before beginning the next. This helps assure a smooth experience.

Connector Types

HP OpenView Select Identity communicates with enterprise applications and resources to configure and manage user accounts and entitlements in those systems. This is accomplished using a Select Identity component called a **connector**. A connector uses protocols defined the the target resource to send commands to the resource, and so forms the communication channel between Select Identity and a specific type of resource.

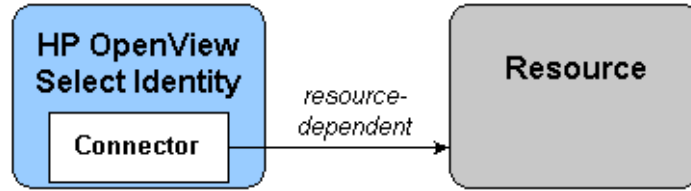
Select Identity uses one-way connectors and two-way connectors:

- **One-Way**

A one-way connector can send communications *to* a resource, but gets no data *from* the resource. That is, all communications travel one way through the communication channel provided by the connector.

Provisioning operations that are initiated by Select Identity are passed to the resource by the connector. The following diagram illustrates the flow of data:

Figure 5 A One-Way Connector



See your product data sheet at <http://managementsoftware.hp.com/> for a list of the connectors available on your product media. Additional connectors are added periodically. Check with your HP representative for the latest list.

The connector is installed on the Select Identity server, and sends requests from Select Identity to the resource according to the protocol defined by the resource.

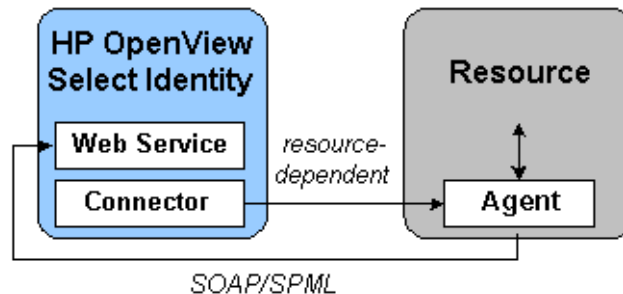
The one-way connectors provided with Select Identity do not require an agent on the resource to achieve provisioning activities, although that model is possible.

- **Two Way**

A two-way connector contains the connector that resides on the Select Identity server and, typically, an agent that resides on the resource. The connector communicates with the agent, and the agent then performs the provisioning operations.

The agent also listens for changes on the host resource, and sends notices to Select Identity when changes are detected. Thus, a two-way connector enables data to flow in two directions, as illustrated in the following diagram. Changes to user accounts can occur on either system, and be kept synchronized.

Figure 6 Data flow between Select Identity and a two-way connector, through a resource agent



See your product data sheet at <http://managementsoftware.hp.com/> for a list of the connectors available on your product media. Additional connectors are added periodically. Check with your HP representative for the latest list.

The connector issues a request according to the communication protocol used by the resource.

When the agent issues a request to Select Identity's web service, it typically uses the Simple Object Access Protocol (SOAP) with a Service Provisioning Markup Language (SPML) payload, through HTTP or HTTPS. Other mechanisms can also be used.

Fundamentally, a one-way connector always initiates any communication with a resource; the resource can not initiate communication. Because of this, changes to user identity data must be first updated in Select Identity, and then pushed to the resource.

In contrast, a two-way connector lets either Select Identity or the resource initiate communication. This is because it includes an agent that resides on and monitors the resource. Data can flow in two directions, and updates to user accounts can occur on either system, and the other gets informed.

There is also a limited set of two-way agentless connectors. Select Identity can poll data from these connectors.

Installing Connectors

You need to install one connector for each type of resource that you want to use, but only one. That is, multiple resources of a given type can use the same connector. For example, in order to connect to three LDAP servers, you need to deploy only one LDAP connector. That one connector handles communications with all resources of its type.

Each predefined connector is supplied with two installation documents:

- The *Connector Deployment Guide* describes the common tasks to be performed on the application server and in Select Identity to install any connector.
- A connector-specific *Connector Installation and Configuration Guide* that contains information about that specific connector and attribute mapping files.

Use the appropriate guides for each connector you install.

Defining Resources and Attributes

At this point in developing your solution, you are ready to create the service attributes you identified in the planning phase (See *Steps To Success* on page 28), add your resources into Select Identity, and map the attributes of one to the other.

Adding New Select Identity Attributes

The Select Identity attributes are defined through the **Attributes** pages. Not all of these attributes must map to outside resources. They can be specific to Select Identity, or to your business. If attributes are not mapped to a resource, they are valid in Select Identity only. See [About Attribute Mapping](#) on page 39 for information about attribute mapping.

You can add any number of attributes to manage identity information. Later, if you map an attribute to a resource, it will be managed like other resource attributes during account addition and updates.



The online help defines a procedure for adding a new attribute in Select Identity that you want to map to multiple resources that already exist. If, however, you need to map all of a resource's attributes to the matching Select Identity attributes at one time, see [Creating Resources in Select Identity](#) on page 38.

Complete the following tasks to create each new attribute. See the online help for specific procedures for each task:

- 1 View the attribute list
- 2 Add a new attribute
- 3 Select resources to map to the new attribute (optional)
- 4 Add constraints and external calls to the new attribute (optional)

Configure Attributes

As you create Select Identity attributes, you have to configure those attributes within Select Identity, to make them meaningful.

To do this, you can assign external calls to an attribute for the following functions:

- **Value Generation**, which generates a specific value for the attribute.
- **Value Constraint**, which constrains the attribute value to a particular value. You can specify values, or use an external call that provides dynamic values. Constraining an attribute that will be used during reconciliation can accelerate the reconciliation process. Learn more about reconciliation in [Chapter 5, Configuring and Testing User Account Reconciliation](#).
- **Value Validation**, which calls an external program to validate the value or format of the attribute.
- **Value Verification**, usable only on passwords, is used to verify the users password in place of the standard database lookup.

For more information about creating external calls for attributes, see [Appendix E, External Calls](#), and the *HP OpenView Select Identity External Calls Developer Guide*.

Using Attributes to Facilitate User Searches

User accounts can consist of many attributes. Typically, users are searched based on certain key attributes (email, SSN, employee ID). Certain user profile attributes can be added to the `TruAccess.properties` file and used to expedite search functions. If these attributes are set, the `TAUser` database table must be extended by adding extra columns that reflect these values. The extra attributes must then be mapped to those columns.

To specify searchable attributes you must do the following:

- 1 Identify the key attributes, such as Social Security Number, Employee ID, or email. You must make sure these are defined within Select Identity and within the mapping file used for each system resource in which data is stored.
- 2 Add corresponding columns to the `TAUser` table in the database.
- 3 Add entries in the `TruAccess.properties` file.

See the *HP OpenView Select Identity Installation and Configuration Guide* for information about editing the database tables and `TruAccess.properties` file.

Creating Resources in Select Identity

The next step will be to define your Select Identity resources. Before doing this, however, you need to understand two important concepts:

- Resource classes
- Attribute mapping

As you define your resources, you will perform attribute mapping. This is essentially the process of establishing the link between each attribute defined on a resource and the corresponding attribute in Select Identity.

About Resource Classes

Select Identity defines two classes of resources:

- An authoritative resource, which contains the most trustworthy information about key aspects of a user's identity.
- Non-authoritative resources typically need to stay synchronized with regard to the key aspects of a user's identity that originate at the authoritative resource, and may also contain other resource-specific identity data.

Select Identity lets you employ an authoritative resource, which it uses as the “master” source for important user identity attribute values. An identity's attribute values that exist on an authoritative resource are taken to be true and correct values. These values should override any conflicting values in Select Identity, and furthermore, may be propagated to other resources that use them.

For example, your human resources system may contain all of your company's most current employee information. This might be a good choice for your authoritative resource. If so, you would add this application as a resource, and designate the system as the *authoritative* resource for user information.

Changes on the authoritative resource propagate to other accounts during a process called “reconciliation”, which synchronizes identity data across resources.

Reconciliation is a large topic itself. (See [Chapter 5, Configuring and Testing User Account Reconciliation](#), for details.) Essentially, reconciliation is the way Select Identity synchronizes its data with the data in a resource. Reconciliation ensures that Select Identity and the resource are up-to-date and consistent. Select Identity propagates new data to non-authoritative resources as necessary.

You can set an attribute's synchronization properties — called Sync-In and Sync-Out — to determine which resources have authoritative data for it.

For much more information on this topic, see [Chapter 5, Configuring and Testing User Account Reconciliation](#).

About Attribute Mapping

Select Identity lets you define the way in which user identities are managed and stored. Each user profile typically comprises many attributes, such as `username`, `first name`, `last name`, and `email address`. The resources that you add often define additional resource-specific attributes.

Each resource has a specific connector which operates as the interface between Select Identity and the resource being provisioned. A mapping file is associated with each connector, which contains resource-specific attributes. This file maps the connector to the resource, and defines where and how identity information is stored on that resource. During the resource deployment procedure, you can view the file that the connector uses to map resource attributes; see the online help for details.

The Select Identity **Attribute** pages allow you to map Select Identity attributes to the attributes defined in the connector mapping file. The attribute mapping process enables access to services which control the provisioning of resource accounts.

For example, let's say you define two resources: Oracle Financials and Hyperion Reports. Each application is defined as a separate resource on Select Identity, and has its own distinct connector. Within each application, each of these resources has its own system administration function that manages the identity of users on that application alone.

On Oracle Financials, the user's first name is defined in a field called `First`. Hyperion Reports defines the field as `Name First` while you created an attribute in Select Identity that defines the field as `FirstName`. Attribute mapping correlates the `FirstName` field to the

corresponding attribute on each resource. Then, when you change the field in Select Identity, the reconciliation process updates `First` on Oracle Financials and `Name First` on Hyperion through their respective connector mapping files.

The following is a sample of a mapping file:

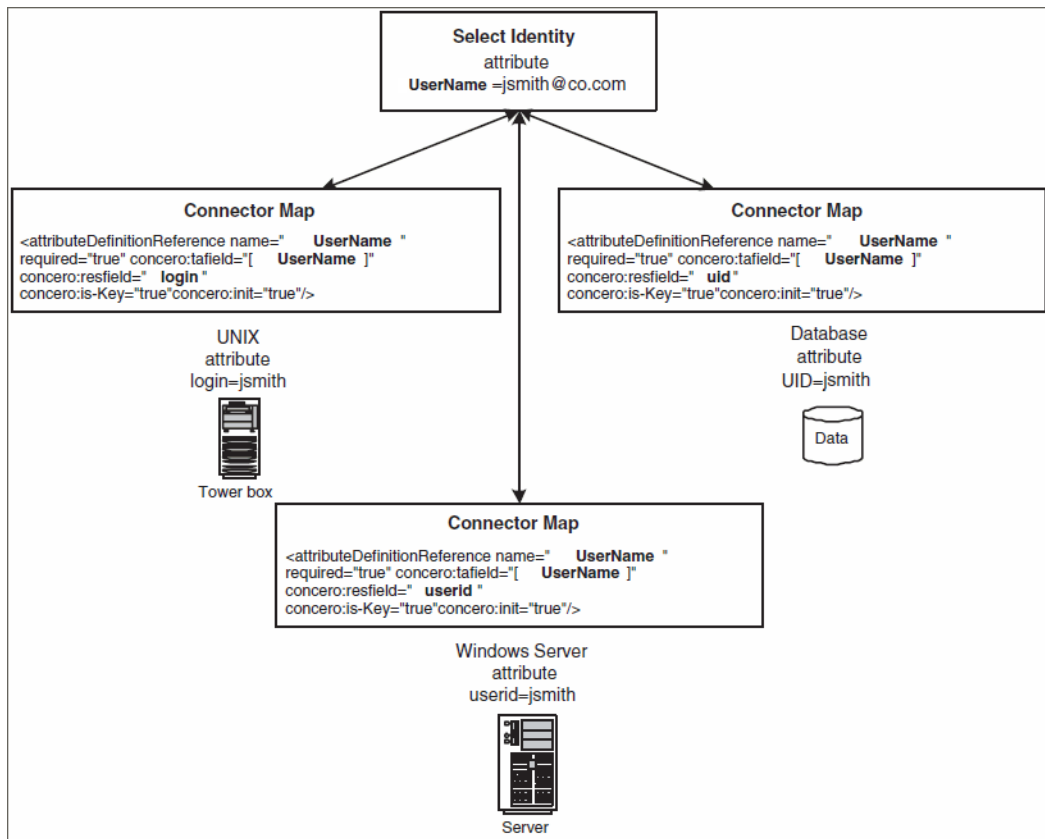
```
<memberAttributes>
<!--
  For iPlanet
-->
<attributeDefinitionReference name="UserName" required="true" concero:tafield="[UserName]"
concero:resfield="uid" concero:isKey="true" concero:init="true" />

<attributeDefinitionReference name="Password" required="false"
concero:tafield="[Password]" concero:resfield="userpassword" concero:init="true" />
```

The Password will appear on the left side of the mapping page in the user interface.

Figure 7 on page 40 illustrates how the attribute “UserName” is mapped to multiple resources through each connector mapping file.

Figure 7 Attribute Mapping Example



If you offer one or more services that rely on these three resources, users who register for the service can be mapped accordingly. This lets you create a standard set of profile attributes for your users that are relevant for your business, and then map them to any of your system resource applications, regardless of how the attribute is defined on the resource.

Attributes that are automatically mapped between Select Identity and resources are **key attributes** (attributes that are required by the resource) and **entitlement attributes**.

When you add a new user through any service, you must define the `UserName` attribute, as well as any other attributes that are required by the service resources. For all other operations, at least the `UserName` is required.

- ▶ The `Password` attribute is required if Select Identity is managing the password. If however, a third-party single sign-on solution is being used to manage user passwords, then the password is not required.

Creating Resources

In your planning, you identified resources that will be part of your identity management solution. You now need to add each one to Select Identity, and map the resource attributes to the corresponding Select Identity attributes.

- ▶ See [About Resource Classes](#) on page 38, [Attribute-Level Reconciliation Policy](#) on page 103, and [About Attribute Mapping](#) on page 39 for important background information you need to know when you create a resource.

If you have multiple resources to add that are very similar, save time by adding one, and then copying it and changing those fields that should be different. Note that all of the connection and configuration information is copied as well.

For each resource in your solution, you will perform the following steps

- 1 Add the resource to Select Identity according to the procedure provided in the online help. If the connector supports mapping (and the connectors supplied with Select Identity do), you need to reference the location for the for the schema (mapping) file, as specified in the connector installation guide.
- 2 After creating a new resource you must map the resource attributes to the corresponding Select Identity attributes.
 - ▶ The online help defines a procedure for mapping a list of attributes for a new resource to the matching attributes on Select Identity. However, if instead you need to map a new Select Identity attribute to multiple resources, see [Adding New Select Identity Attributes](#) on page 37.
- 3 Set resource reconciliation policies. for the resource attributes according to the procedures in the online help. Reconciliation supports the following operations based on the policies you create:
 - Add, delete, enable, or disable a service
 - Enable, disable, or terminate a userSee [Chapter 5, Configuring and Testing User Account Reconciliation](#) for more details about reconciliation policies.
- 4 Define the resource entitlement caching policies you want to use. Entitlement caching reduces the need to retrieve entitlements from resources, and so improves the overall performance of the system.

Removing a Resource

At some point, you may need to remove a resource from a service. This can happen if a resource is obsoleted, if your business processes change in some significant way, or for other reasons. Delete a resource from Select Identity when no service or user uses it anymore.

For procedures to remove a resource from a service, or from Select Identity altogether, see the online help. See also [Set Up Service Reconciliation](#) on page 82, because you may need to update users after removing a resource from a service.

Configuring Parameters

Up to this point in the “Build and Develop” phase of your project you have developed the extensions to Select Identity that you need, installed all required software for Select Identity to function, and established connectivity between Select Identity and your resources.

Now it is time to implement your identity management design by configuring it in the system. There are a number of tasks involved:

- 5 Configure workflows (See [page 42](#))
- 6 Set up configuration approval (See [page 44](#))
- 7 Register external calls (See [page 46](#))
- 8 Configure notifications (See [page 46](#))
- 9 Set the challenge and response questions (See [page 48](#))
- 10 Create services (See [page 48](#))
- 11 Create administrative roles (See [page 52](#))
- 12 Establish auditing and configuration reports (See [page 54](#))
- 13 Set up My Identity user tasks (See [page 57](#))
- 14 Set up self-registration (See [page 59](#))

The order presented above, and in the body of this section, is a logical flow. However, you may find that in practice these tasks aren’t always performed such a systematic fashion.

Configure Workflows

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 then pushing the user account to the external resource. Or, provisioning can require approval by multiple Select Identity administrators. The approval process may 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 new responsibilities, the employee must be granted new entitlements and access privileges. Before giving the employee access to these systems, upper-level management must 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 Overview

Workflow Studio enables you to create the workflow templates representing the provisioning process. A workflow template models this process in order to automate the actions approvers and systems management software must perform. The workflow process can also rely on an external call to a third-party system or database. See [Default External Calls](#) on page 240 for more information.

An administrator with access to **Workflow Studio** actions defines the workflow templates and processes by which users are added to, updated, or removed from the system. A workflow may require one or more steps before completion.

Each approver is notified by email when a new account needs to be reviewed. That administrator can then log in and access the **Worklist** section of the Select Identity client, where a **Pending Tasks** notification appears at the bottom of the home page, in the section titled **Requests**.

The template creation process can be as complex as your business security policies dictate. The workflow studio online help describes concepts and procedures you require 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 created as part of a service. See [Task 4](#) on page 50 for details.)

For example, you can assign a simple provisioning template to an add service request for self service. This template might perform user provisioning and request a single approval. Then, when a user requests access to the new service via the **Add Services** option on the **My Services** page, the template is invoked and an administrator must approve the request before the user is added to the supporting systems.

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** page.



Select Identity automatically grants the Approver role only for approvers gathered from an approver external call, and not from “get approvers by role”.

Approvers acquired through “get approvers by role” are assumed to have the necessary role already.

Those acquired from an external call need to be given that role. Otherwise, you might have a situation where someone has approver responsibility, but can not access the necessary functionality.

See [Create Administrative Roles](#) on page 52 for more information.

- Sending email using one of the email templates created on the Notifications page. See [Notification Variables](#) on page 46 for more information.
- Executing an external call to access 3rd party systems. See [Appendix E, External Calls](#).

Set Up Configuration Approval

As noted in the *HP OpenView Select Identity Guide to Concepts*, configuration approval lets you regulate high-risk changes within Select Identity by establishing an approval workflow for configuration changes. Using configuration approval is not required, but is highly recommended.

Approval Workflow

The configuration approval feature establishes an approval workflow for changes to Select Identity setup. Approval workflows can be established for changes to the following:

- Workflow Templates
- Resources
- Services
- Attributes
- External calls
- Rules
- Email notification templates
- Administrator roles
- Connectors
- Notifications
- Configuration approval setup

Changes requiring approval include additions, modifications, deletions, or updates from import. Copying an item that requires approval initiates the same workflow as adding an item. If an approval workflow is established for updates to an item, all future changes will require approval. For example, if the LDAP resource requires approval, any additions or updates to the LDAP resource initiates the approval workflow for resources.

Approval requirement defaults are established for each item type. Additional control is available on an individual item basis. For example, you may elect to require approval for changes to all provisioning workflow templates, with the exception of the password expiration email workflow template. Establishment of approval workflow requirements can be very granular, giving you the flexibility to establish controls appropriate for your environment.

For each item, a checkbox, **Approval Required**, appears on the **Add** or **Modify** pages. If this box is unchecked, modifications can be made to the specific item without approval. If the box is checked, approval workflow is required for the change.

The default setting for the **Approval Required** checkbox for each item depends upon the setting for the item type in configuration approval setup.

- If configuration approval is enabled for the item type, the box is checked by default, meaning approval is required for changes. If configuration approval is enabled for an item, clearing (unchecking) the **Approval Required** box is treated like a modification and requires approval.
- If configuration approval is not enabled for the item, the checkbox is unchecked by default. Checking the box does not require approval. However, any further changes made to the specific item *will* require approval.

Changes to the following items are not controlled by configuration approval:

- Workflow Application Definitions
- Workflow Reporting Template
- Property File Changes, including the encryption key
- Connector Mapping updates made using the Attribute Mapper utility
- System Hint questions and Challenge configurations
- Saved or Scheduled Reports
- Scheduled Reconciliation Jobs
- Scheduled Bulk Jobs
- System Security
- Object Migration Configuration


Roles and the Configuration Approval Process

A new role and three new permissions are part of the configuration approval process. These are the new administrator role permissions used for configuration change management:

- **Approve Configuration Changes** – Approval of requests for configuration changes.
- **Configuration Change Management** – Ability to modify the configuration approval setup.
- **Manage Configuration Change Request** – Ability to view and manage configuration change requests that have been created.

A new administrator role, **Configuration Approver**, is used in this process. This role is similar to the workflow approver role, but is used for configuration approvals. It has the Approve Configuration Change permission.

None of the new permissions are assigned to any other role initially, including the Conero Sys Admin (sisa) account. Keeping this role separate from the system administrator role divides the responsibility for system changes.

 Users cannot approve their own configuration change requests.

Configuration Approval Workflows

Workflows for configuration approval are created using **Workflow Studio**. The `Workflow Type` property is used to indicate whether a workflow is used for configuration or user management.

There is no limit to the number of approval or notification blocks that can be added to the workflow.

An additional block, **Commit**, is required for configuration workflows. This block executes the changes in Select Identity after approvals are complete.

A new application is available for approval, **Get Approvers by Role for Config. Changes**

External calls that access user data and email verification blocks are not allowed in a configuration workflow.

The following persistent workflow variables are created in every configuration workflow:

- `$RequestId`
- `$_instId`

- `$(ConfigObjectName)`
- `$(ConfigObjectType)` (Service, Resource, Workflow, etc.)
- `$(RequestActionName)`

Default Configuration Workflow

Select Identity includes a standard workflow for configuration approvals, *SI One Stage Approval for Config*. This workflow notifies all users with the configuration change management role of a pending configuration change request. Only one configuration approval user has to approve the request. This default workflow can be copied and then customized to meet the requirements for your company.

Custom Configuration Workflows

You can create custom configuration workflows or use a copy of the default configuration workflow as a starting place. All configuration workflows must contain the **Commit** block.

Configure External Calls

Register any external calls that your plan called for and that have been developed. See the online help for specific procedures.

Configure Notifications

Email notices can be sent to a user when an account is approved, rejected, or modified. Email can also be sent when an account password or hint is reset. The Notifications facility lets you define the content of these email notices as templates that are filled out automatically.

Select Identity is equipped with a number of standard notification templates. You can use them with changes, modify them to suit your organization, or create new templates for your own purposes.

Notification Variables

When creating notification templates, you use variables inside the notification to stand in for user and request information specific to the situation. The variables are replaced with actual values when a notification is sent using the email template.

- ▶ Email notifications can contain potentially sensitive or confidential data, such as passwords. These emails are stored in the Select Identity database, but should not be stored as clear text. To encrypt private data, wrap it with the `<ovsi-encrypt>` tag.

For example, use `<ovsi-encrypt>[RQT:Password]</ovsi-encrypt>` for a “New Account Password” notification.

Table 1 Notification Template Variables

Variable Type	Description	Variable
Request	<p>Variables for the Request Object. The Request variable provides the ability to reference “request” information in an email template.</p> <p>Following are predefined Request variables:</p> <p>[REQ:ParentRequestId] [REQ:ServiceName] [REQ:RequestId] [REQ:RequestActionName] [REQ:RequestActionDescription]</p>	REQ:
RequestTarget	<p>Variables for the userID that is being created. The RequestTarget variable provides the ability to reference information about the target user being provisioned in an email template. Any attributes associated with the user for the given service in the request may be accessed.</p> <p>Example: [RQT:UserName]</p>	RQT:
User-Defined	<p>The User-Defined variable provides the ability to reference user-defined variables defined in a workflow for use in an email template.</p> <p>Following are a list of predefined User-Defined variables that can be used in email notifications:</p> <p>[USERDEF:Status] — Denotes the status of the service. [USERDEF:ResetStatus] — Denotes the status of provisioning for a resource within a service. [USERDEF:Action] — Action performed against the targeted user. [USERDEF:ServiceName] — The service associated with the workflow request. [USERDEF:pendingTaskURL] — If an approver is required for a request, this variable contains the URL string in Select Identity used to approve the request.</p>	USERDEF:
Requestor	<p>Variables for the administrator making the request. The Requestor variable provides the ability to reference information about the person submitting a request in an email template. Any attributes associated with the administrator or requestor requesting the action (such as modify user), can be accessed in the email template.</p> <p>Example: [RQSTR:UserName]</p>	RQSTR:

Variable Type	Description	Variable
Workflow	<p>Variables defined in the workflow template. The Workflow variable provides the ability to reference variables defined in the workflow template. Variable names of persisted variables begin with \$ and are stored in the Select Identity database, even when a workflow instance ends. Access these variables at any time once the workflow instance is created.</p> <p>Select Identity provides the ApproverComments variable, but you can create your own.</p> <p>Example: [WF:\$ApproverComments]</p>	WF:
Environment	<p>Variables are defined for the environment within the properties file. The environment variable provides the ability to reference variables defined for the Java Virtual Machine (JVM) environment. By default, Select Identity adds all properties from the TruAccess.properties file to the Java Virtual Machine (JVM) environment. Any value that performs a System.getProperty() can be used for this variable.</p> <p>Example: To access a version of Select Identity, you might use the following in an email template:</p> <p>[ENV:truaccess.version]</p>	ENV:

Set the Challenge and Response Questions

Users sometimes forget their passwords. Select Identity can let users reset their password by answering Challenge and Response questions. The user is asked a question — the challenge — and must provide the correct answer — the response.

The Select Identity challenge and response policy governs password hints for the system. You can restrict login attempts with this policy, and force users to configure a password hint the first time they log in.

In Select Identity, System Administrators create sets of standard challenge questions. In the settings area, Administrators select the number of standard and personal questions the user must answer. Users create their own personal questions when they establish themselves in the system.

Formulating a good challenge question requires careful planning and forethought. In an increasingly global and multi-cultural world, a question may have relevance to some people and no relevance to others. For example, in some countries, the American standard challenge questions of “your mother’s maiden name” or “your father’s middle name” are not meaningful because of cultural differences.

Create Services

This section describes how you create services. Procedural details are available in the Select Identity online help.

The Service Function in Select Identity

The **Service Studio** function lets you add, modify, and delete the services that are accessed by the employees, customers, and business partners who use your resource applications. These services, in addition to the service role structure that you establish, form a management structure for users of the Select Identity system.

You can also set service roles and context user groups from the **Service Studio** pages. Services are made available to your customers and partners by creating a service role discussed in [Task 4](#) on page 50 for details.



Omit the `Password` attribute from the view that will be used to modify users. Administrators should use delegate (**Reset Password**), and end-users should use self-service (**Change Password**) to update passwords.

Steps to Creating a Service

Creating a service is a critical process in deploying your identity management solution. It involves performing a sequence of related tasks. The tasks listed below represent the overall flow of work during this process, and should direct your efforts.

For detailed steps to perform each task, see the online help.

Task 1: [Verify that all of the prerequisites have been met.](#)

Ensure that the following points are true:

- Resource applications and data stores required to support each service have been set up, and the appropriate attributes have been mapped
- Notification templates exist as necessary to support provisioning
- Workflows needed to create and maintain your services have been created, and include appropriate approvals.

Task 2: [Create the service, which defines the service type, the superset of resources, and the attributes that are required for access to the service, including the context attribute.](#)

Create composite services to simplify maintenance by allowing you to update common attributes from one request. Composite services combine two or more similar services into one composite service unit. Most composite services contain an administrator service and at least one business service.

Task 3: [Define attribute values and properties, which determine the attribute characteristics that are acceptable for this service.](#)

Define a set of attributes and values that are available for or required by the service. Attributes are created and managed through the **Attributes** pages. The attributes that are available for a service are determined, in part, by the resources that are selected to support it. Additional attributes that are specific to the Select Identity system or your business may also be available. Adding a resource to a service will add `<resource name>` entitlements and `<resource name>` key attributes in the service.



When possible, HP recommends that you use constraints to restrict the values that a user can select from when registering for a service. This improves performance, especially if there are a large number of values. For example, you may have the attribute “Country” available and want to restrict value options to “USA,” “Korea,” and “Japan” for a particular service.

The `UserName` attribute should be in all services, in addition to the context attribute and resource attributes.

- ▶ The online help describes two distinct procedures: one to map multiple attributes to one new service — which is what you would do at this point in creating a service — and one to map a single attribute to multiple services.

Task 4: Create service roles, which define the way in which users access the service.

You may recall from the *HP OpenView Select Identity Guide to Concepts* that all possible entitlement privileges are defined at the service level. Within that set of entitlements, some are fixed and some are optional.

A subset of service roles may be defined to constrain entitlements to different user contexts. The root service role may be given a subset of optional entitlements existing on the service. Service roles are divided into parents and children. Children receive all fixed entitlements and a subset of the optional entitlements available to the parent. A child service role can never have more entitlements than its parent.

The concept of fixed and optional values are also applied to attributes other than entitlements. You can fix a value or constraint a list of values for a certain attribute.

Service roles define the business process which associates entitlements, workflows, and policies that will be used to access the service. Service contexts enables you to assign a service role to a group of users, thus providing access to the service under the terms you established.

Defining service roles lets you assign granular rights, or levels of service, to various user groups.

Service roles are hierarchical in terms of management and create a secure way to share services across different companies or locations. Service role settings take precedence over the service configuration.

When creating the first service role for a service, the `parent` option is not available.

- ▶ The first service role that you create for each service defines the superset of options (parent) for all other service roles created for this service. On the **Service Information** page, all the events and templates are populated by default for the first service role. Simply select and delete the events and templates you do not want to use.

Request Events

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

Table 2 Delegated-registration Request Events

- Add New User
- Add Service
- Delete Service Membership
- Disable Service Membership
- Enable Service Membership
- Modify User
- View Service Membership
- Move User

Viewing a service membership does not use workflow. In addition, moving a user does not use a workflow from a service role. It picks up a workflow from the `TruAccess.properties` file.

Table 3 Reconciliation Request Event

- Add Service
- Delete Service Membership

Table 4 Bulk Request Events

- Add Service
- Add New User
- Modify User
-

Table 5 Self-Service Request Events

- Add New User
- Modify User
- Add Service
- View Service Membership
- Delete Service Membership

Service Reconciliation Request Event

- Modify User



You can attach a view for an approver to a request event. Click the **Request Event** you want to reference, and select the associated workflow from the **Workflow Process** list. See the online help for specific procedures.

Task 5: Create service forms, which determine the registration criteria for access to the service.

After you create a service, create forms that are valid for different groups of users. For example, if you want a specific set of users to see only certain fields when registering for the service, define a form that makes only those fields available.

You can also use these forms to determine what information approvers see when requests are processed through workflow steps. Each approval block within a workflow can have a different service form associated with it. See [Task 4](#) for information on creating service roles and the associated forms.

Additionally, service forms can be used to create a multi-page form. A multi-page form can be used in delegated user management, self service, self registration, and the approval pages to determine what users see for each action. The display of these forms must be ordered accordingly.

Task 6: Create service context groups, which defines a logical grouping of users accessing the service.

Service context enables you to assign a service role to a context value. You can also select workflows for request events in case you select multiple service roles. There can be one wildcard context in the service context tree, and it must be at the lowest level in the tree. Enter a value to identify one context as a wildcard.

Service context lets you assign a service role to a group of users based on a common attribute, thus providing appropriate access to the service.



When selecting a context variable, you can simplify specification by using wildcards. A wildcard context has a value of "*" (asterisk). This means that any value that is not specifically defined on a context will fall into the wildcard context.

- A wildcard context with no constraints on the context attribute lets you type in your own context value, and any value you use will be accepted.
- A wildcard context with constraints allows any of the constrained values, but you do not have to specifically create a separate context for each value.

See [Configure Attributes](#) on page 38 for more information about value-constrained attributes.

Important Tips

- When you want to create a service that is very similar to an existing service, copy the existing service. Once the service has been copied under a new name, then simply modify those part of the service that are different from the original
- Occasionally when you are creating services you may make an error during the addition of new contexts. If users have NOT been imported or otherwise added, simply delete the context, and re-enter it correctly.
- When you need to change the capabilities of an entire group of users at one time, use Select Identity's **Move User** feature to modify the context of users assigned to a service. By changing context, the users get the entitlements of the associated service role, and lose the entitlements of the prior service role. See the online help for instructions.

Create Administrative Roles

When you create a service, you have the option to define it as an administrative service. You can then add one or more users to this administrative service to assign administrative roles to your users in charge of system administration. See [Create Services](#) on page 48 for information about creating a service.

Administrative roles govern the permissions that each administrator can perform within Select Identity. Administrative roles are made available through administrative services that are designed specifically to manage business services. You can then add users to this administrative service, who will serve as administrators of your business services.

Granting administrative access to resources through services lets an administrator manage multiple resources and data stores provisioned by various services. For example, one administrator might administer customer access to your purchasing and billing systems while another administers access to internal support applications.

Administrators may delegate their roles to other users, which prevents you from having to create and assign a new role each time an administrator is absent for any reason. See [Delegating an Administrative Role](#) on page 54.

Administrative Functions and Actions

You should become familiar with HP OpenView Select Identity administrative functions and actions before creating administrator roles. Roles and actions are designed to represent all of the management functions within Select Identity and are named accordingly. Each grouping of actions is represented by a management link in the client application.

Understanding Functions and Actions

Use roles to grant an HP OpenView Select Identity administrator permission to perform a group of actions performed within each management functional area in the client application. The actions assigned to each administrator help form a view of the system. Management functions may be assigned to more than one administrator, but each administrator must have at least one function.

You can select the permissions you want an administrator to have when you create an administrative role. The permissions for roles are grouped much like they are in the **Tools** menu.

See the online help for detailed information about creating administrative roles, and for detailed information about the functions and actions that are available to administrative roles.

Reviewing Default Roles

HP OpenView Select Identity offers default administrative roles. Use the existing roles “as-is” or modify one or more to better match your business requirements.

End User

All users added to Select Identity are granted this role automatically. Additionally, users added through user import are granted this role when a user first logs in. Each user receives a default set of permissions. You can change the default permissions granted to end users by modifying this role.

An end user is simply a user of Select Identity services. Accounts with this user role have only the entitlements granted through the registration of a service. Users granted the end user role gain access to the **My Identity** self service menu which may be customized to meet your organization’s specific needs.

Approver

An approver, often called the workflow approver, performs account provisioning actions. The system automatically grants this role to users assigned any approval task. A user with this role can approve user account additions, modifications, or deletions for those users within the approver’s context user group.

Configuration Approver

A configuration approver is authorized to approve configuration changes to Select Identity. See [Roles and the Configuration Approval Process](#) on page 45 and the *HP OpenView Select Identity Guide to Concepts* and for more information.

Concero Sys Admin

The Select Identity system administrator, still known as the *Concero Sys Admin*, has the most powerful administrative role, and is able to perform any action related to administrator roles, connectors, resources, workflows, services, notifications, users, external calls, reconciliation, and attribute management actions. You cannot delete this role.

Creating and Managing Administrative Roles

The administrative role function lets you create, modify, and manage the roles defining each administrator’s permission to make changes in the Select Identity system.

You can add any number of roles to meet your management needs. Roles are later assigned to users through administrator services. Save time when creating a new administrative role that is very similar to an existing role by copying and modifying the matching role instead.

After creating an administrative role according to the procedures given in the online help, you need to grant permissions to the new administrative role

Take great care to review your permission selections to be sure you are granting the right authority to this new role.



If the role will be granted all or most of the available permissions on the page, then click the **Include All Permissions** check box.

Tips

- Change the authority granted to an administrator by modifying the permissions granted to the administrative role.

- If you have permission to do so, you can create new administrative roles by copying and modifying existing administrative roles. You must have permission to both create and modify a role before you can complete the copy procedure. Use this method to save time when you have to create one or more administrator roles that are nearly the same.

Be aware that modifying permissions from a copy operation is not permitted. Access to view permissions is available; however attempts to change permissions from this screen fail.

Delegating an Administrative Role

Delegation of administrative roles is optional. Administrators delegate their roles using the **My Identity** self service tools when the Select Identity system has been set up to provide this privilege. See the online help for more information.

Deleting an Administrative Role

Any custom-created administrative role can be deleted, as long as no users are associated with it. The predefined administrative roles (**Concero Sys Admin**, **Workflow Approver**, **Configuration Approver**, and **End User**) can not be deleted.

Establish Auditing and Configuration Reports

Select Identity auditing and reporting features let your organization produce context-driven, standard, and custom reports of user entitlements and system event history. Better reports and audits allow tighter control over information, reduced risk of security breach, and enforce higher levels of compliance with requirements and regulations.

This chapter provides details for all of the actions that you can perform within **Audit Report** and **Configuration Report** capabilities. Access to each of these functional areas is determined by the administrative roles you assign a user's account.



These reports are not the only reports available in HP OpenView Select Identity. You can generate reports pertaining to specific areas of Select Identity, such as User Management, on the function pages.

Audit Reports

Generate an Audit report to view configuration activities for one or more specified accounts or services over a period of time.

Audit reports detail all actions related to users within Select Identity. To view a report, you must generate the report. Audit reports can be generated based on different parameters including a specific user, service, or service and context. Data displayed for each report is configurable and may include a wide variety of columns.

HP OpenView Select Identity provides audit reports for all Select Identity system functions. Audit reports provide the history of activities within the system. Audit reports detail historical transactions that have occurred in Select Identity.

You can generate a single report or create a report template, which can be accessed each time you click Audit Reports.

Note that Select Identity does not record changes to the documents themselves.

- ▶ The system audits changes to document attributes by logging the change to the document ID. You may choose to use a third party tool to view more extensive audit logging reports. The information necessary to access audit logging data is described in detail in [Appendix D, Auditing XML and Client Sample](#), beginning on [page 185](#).

Available Audit Reports

Audit reports include the following detailed reports and summary reports:

Audit User	Reports all changes to specified user accounts related to one or more specified Services within HP OpenView Select Identity over a defined period of time. The Audit User Report may be filtered so that only specified changes are reported such as only accounts that have been added or deleted.
Audit User Summary	A detailed report listing the users added to services.
Audit Service	Reports all changes to user accounts related to one or more specified Services within HP OpenView Select Identity over a defined period of time.
Audit User Creation	A detailed report listing the users added to services.
Audit User Deletion	A detailed report listing the users deleted from services.
Audit User Termination	A detailed report listing all the users terminated from the system.
Audit User Password	A detailed report listing all password actions and activities.
Audit User Creation Summary	A summary report listing the number of users added to each service
Audit User Deletion Summary	A summary report listing all the users deleted from each service.
Audit User Termination Summary	A summary report listing all the users terminated from the system.
Audit User Password Summary	A summary report showing the number of users involved in each type of password action.
Audit User Login	A detailed report of user activity.
Audit Hint	A detailed report showing how many times users set their password hints.
Audit Summary Hint	A summary report showing the number of users who set their password hints.

See the online help for information on how to generate any of these reports.

Configuration Reports

HP OpenView Select Identity provides configuration reports for user, administrator, and service management activities. Configuration reports represent the state of Select Identity at the time the report is created.

For example, an administrator can display all users associated with a service context at a given time. You can generate a single report or create a report template that can be accessed each time you click **Configuration Reports**.

The following are descriptions of the available configuration reports.

Report Name	Description
Admin Configuration Report	Generate this report to summarize all current users with administrative privileges. This report displays user information, administrative service and context affiliation, and managed contexts and services.
Resource Entitlements Report	Generate this report to summarize users/entitlements that are in Select Identity, but are not provisioned to a resource. Only differences are reported.
Resource Reconciliation Report	Generate this report to compare users and entitlements in the Select Identity database with the information in a selected LDAP or UNIX resource.
Resource Users Report	Generate this report to summarize users and entitlements that are different between HP OpenView Select Identity and the resource. Only differences are reported.
User Configuration Report	Generate this report to view active user accounts within Select Identity sorted by context. You can only view users that are currently active within your service context. The report can be generated based on a specific user, a specific service, or specific service and context. The type of data displayed for each report is configurable and may include user ID, first name, last name, email, service, and context.
User Configuration Detail Report	Generate this report to summarize all current user accounts by context attribute and value. The report displays data columns including service name, context, and count.
User Configuration Summary Report	Generate this report to summarize all current user accounts by service and context within Select Identity. The report can be generated either by specific service or by specific service and context. The report displays data columns including service name, context, and count.

See the online help for information on how to generate each of these reports.

Scheduling Reports

You can schedule reports to be run once on a designated time and day or to run at regularly determined intervals. When a scheduled report is generated, Select Identity emails the report if it is within the predetermined size limit or emails a notification that the report is ready to designated recipients. Email notifications include a link to the generated report, which is automatically stored at a predetermined location.

See the online help to learn how to schedule reports.



Tips

- Save time creating reports by copying a similar report, then editing the report display options and settings to fit your needs.
- Inactivating a report keeps the report from printing until it is reactivated, although report settings can still be changed. If the report will not be used again, do not inactivate the report, delete the report instead.
- Deleting a report removes it from Select Identity. It cannot be retrieved at a later date. If you think you may need this report at some time in the future, simply deactivate the report instead.

Understanding Report Parameters

Most reports within HP OpenView Select Identity can be generated as standard reports providing high level information, detailed reports providing information at the transaction level, and summary reports providing numeric totals only.

Different reports have different parameters. See the online help for a detailed description of the parameters for any report you are interested in.



When an optional parameter field does not contain a value, the default value is `All`.

For example, if you want to see the modifications to a specified user's accounts, but you do not specify a service, all modifications made on any service supported are reported. However, when one or more options are listed in a list box, only those options that are highlighted are reported.

Set Up My Identity User Tasks

You can give your end-users permission to perform some simple administrative tasks, typically related to managing details of their own account, such as their passwords.

This alleviates the burden of some of the most common administrative tasks from your IT or support staff. Consequently, most system administrators will want to set up the **My Identity** self-service function for end users.

End users can perform certain administrative tasks through the **My Identity** function, if you give them permission to do so:

- Profile-related tasks
- Password-related tasks
- Service-related tasks

You can give users permission to perform these tasks through the **Admin Roles** function under the **Tools** menu.

Profile-related Tasks

Profile-related tasks include permitting end-users to perform the following actions:

- Modify and/or view **My Profile**. This gives a user access to the profile attributes you specify, such as the following:

- UserID (not modifiable)
- First Name
- Last Name
- Email address
- View request status. If you give end users permission to modify their profiles, passwords, or subscribe to a service (see below), then also give them permission to view the status of their requests.
- View role permissions. Users can see their role permission only if they are granted the **View Role Permissions**.
- Delegate or remove profile attributes
 - ▶ The **Delegate My Permissions** section appears only for users with the **Delegate Admin Role** permission and **View My Role** permission.

You can give users with administrative privileges permission to delegate their administrative roles to another Select Identity administrator or end user within their service context, or remove roles that were delegated. Users can delegate their role for a specific period of time or an indefinite period of time.

Regardless of administrative role (Administrator or End-User), granting this permission can be used to allow a user to delegate responsibility to a co-worker during an absence.

You can also give end users the **View My Role** permission to allow them to view their own delegated role given to them. In order to perform the **Delegate Admin Role** action, the user must be allowed to view his or her role permissions.

Password-related Tasks

Password-related tasks include permitting end-users to perform the following actions:

- Change passwords. You can give end users permission to change their Select Identity login password, or change their password or passwords on one or more resources on which their account is located.
- Change password questions. You can give end users permission to change their password reset questions (hints).

You initially specify the password reset questions through the **Challenge/Response** page. See [Understanding Functions and Actions](#) on page 52 for information about setting the password policies for users.

Users need permission to change their password questions before they can reset their passwords if they forget their password (see [Password-related Tasks](#) on page 59).

Service-related Tasks

Service-related tasks include permitting end-users to perform the following actions:

- View services. Users must have **View Service Membership** permission to view their services.
- Self-subscribe. You can give end users permission to add themselves to additional services.
- View resource accounts. If you give end users permission to view their profile information, they can also view their resource accounts.

Set Up Self-Registration

Users can add themselves to Select Identity through the self-registration process. (For a general description of workflow templates, see [Workflow Studio Overview](#) on page 43, and for detailed information and examples, see the online help.)

For a service, the `Self Add New User` event must be defined with a workflow and view in the service role. The **Self-Registration** page opens differently based on how you configure it.

You perform the following tasks to set up Self-Registration:

- Configuring the Self-Registration Form
- Setting the Self-Registration URL

Configuring the Self-Registration Form

You can configure the Self-Registration form to open for the end user as one of the following forms:

- Self-registration form with a pre-defined context and context value. See [Self-Registration Form with Predefined Context and Context Value](#) on page 60 for more information.
- Self-registration blank form, in which the end user selects the context.

For both these forms, you can set the default self registration view in the service form that displays what attributes should be on the first page.

Setting the Self-Registration URL

When you set up the service, you can define a specific URL to access the self-registration form. (For information about creating notifications, see [Notification Variables](#) on page 46.)

The URL you use determines which self-registration form opens first:

- Self-Registration Form with Predefined Context and Context Value
- Self-Registration Blank Form

Self-Registration Form with Predefined Context and Context Value

Specify the following URL, which opens the self-registration form with a pre-defined context and context value.

```
http://<host_name>:<port_num>/lmz/selfregistration.do?  
serviceName=<service>&contextvalue=<value>&contextName=<name>
```

When using the URL with context defined, the user is presented with the form defined in the service role for the self add event. The actual URL has appropriate values for the parameters in the above template:

- `<host_name>` is the application server (WebLogic or WebSphere)
- `<port_num>` is the server port number
- `<service>` is the service name
- `<value>` is the context attribute value
- `<name>` is the context attribute name

► Everything after `<port_num>` is case-sensitive. That is, `contextvalue` is not the same as `contextValue`, and the error message resulting from a mistake is not clear.

For example, if you replace the values with the following options:

- `<host_name>` is localhost
- `<port_num>` is 7001
- `<service>` is gvA1
- `<value>` is HP
- `<name>` is Company

then you would use the following URL:

```
http://localhost:7001/lmz/selfregistration.do?serviceName=gvA1&contextvalue=HP  
&contextName=Company
```

When the end user clicks on this URL, the **Register to Service: Service Name** page opens with the pre-defined context and context value. In the example above, the context value is predefined as HP. That means a user registering via this URL will be added to the HP context.

► The supporting service role must have a workflow defined to handle the Add New User Self Registration event. To learn more about creating service roles see [Task 4](#) on page 50.

Self-Registration Blank Form

Specify the following URL, which opens the self-registration blank form. When using the URL with no context, Select Identity does not know the context yet, and therefore does not know the service role or view to use, and so opens the **Register to Service** page with blank fields:

```
http://<host_name>:<port_num>/lmz/selfregistration/services.do?serviceName=<name>
```

You must use appropriate values for the parameters in the above template:

- `<host_name>` is the application server (WebLogic or WebSphere)
- `<port_num>` is the server port number
- `<service_name>` is the name of a service that was specified in the workflow.

For example, if you replace the values with the following options:

- `<host_name>` is `betelgeuse`
- `<port_num>` is `7001`
- `<service_name>` is `LDAP70`

then you would use the following URL:

```
http://betelgeuse:7001/lmz/selfregistration.do?serviceName=LDAP70
```

When the end user clicks on this URL, the **Register to Service** page opens with blank fields.

Data Processing

At this point in the **Develop and Build** phase of your project, you have largely implemented the identity management design you created in [Chapter 2](#). However, while everything is set up, there are no users in the system yet. The meal is ready, so to speak, but the diners have not arrived.

The data in your authoritative resource can be used to populate your Select Identity solution user data. This, to extend the above analogy, gets the diners to the table.

Overview

There are three ways to populate and maintain user data in your Select Identity solution:

User Import

User import is typically used for the initial loading of users into Select Identity. You can import user data from authoritative resources and from other target resources. Typically, as part of user import you will also perform service assignment, which assigns the new users to services based on their attributes.^a

Bulk Add

Bulk add inserts users into Select Identity who do not already exist in the target resources. An example might be when two companies merge, and the incoming employees need to be added to Select Identity.

In addition to the users attribute values and resource entitlements, a bulk add operation stipulates which services the users should be assigned to.

Thus, bulk add can be seen as a delegated add operation for a group of users. It is useful when your solution is already in production, and should be used instead of user import in that situation.^a

Reconciliation

Reconciliation is typically used to keep Select Identity data synchronized with the data in various resources, in an ongoing fashion. For example, you probably want Select Identity to keep up with changes to the data in your authoritative resource, as when a user gets a promotion and needs different access rights.^a

Reconciliation is a substantial topic, and for practical reasons it has its own chapter, [Configuring and Testing User Account Reconciliation](#), beginning on [page 95](#). Look there for details about how to set up and use reconciliation.

- a. Both reconciliation and bulk operations invoke workflows; user import does not.



The data to be imported is rarely clean and consistent from the beginning. For example, there may be no unique identifier across the enterprise, or different data sources may be stale, inconsistent, or unreliable.

In the best case, this situation was identified during the design phase, and the necessary clean-up work was factored into your plans. If not, you should plan to spend some time creating a workable data source for user import and bulk operations.

The **user import** function lets you add your organization's existing users to Select Identity. User import is used as a one-time import mechanism to populate users into Select Identity. It does not provision users. The list of users and their associated attributes are specified in SPML files and are subsequently loaded to Select Identity through the **Schedule User Import** action.



Schedule User Import is for one-time file upload only. Multiple files for user import can be uploaded automatically by copying files to the (adroot) upload directory.

After users are added to Select Identity, entitlements or other resource specific attributes associated with users are determined by specifying the resource from which the users originated. These entitlements, like the user attributes, are specified in an SPML file and associated with a user's unique identifier.

After both the users and entitlements are loaded to Select Identity, the **Schedule Services Assignment** action is used to associate the users to the proper services. Associating users to services creates an account which can now be maintained through Select Identity's service-based model.

Process Summary

In this "Data Processing" step of your Select Identity implementation, you will populate your database, and set up reconciliation for more automated maintenance. These are the tasks involved:

- 1 Obtain raw data
- 2 Pre-process the data
- 3 Configure and test user import
- 4 Configure and test multiple-user-ID accounts
- 5 Configure and test bulk data uploads
- 6 Configure and test reconciliation

Obtain Raw Data

During the Design and Planning phase, you identified authoritative resources and probably obtained data samples. Now you will need to obtain full copies of actual production user data, including entitlements.

For one-time user import, obtain full dump files covering all users. For example, obtain all users irectory in LDIF format.

In addition, retrieve delta files for resources that will have ongoing reconciliation; a delta file indicates what user data has changed in a specific repository. Either the resource is able to generate delta files, or you have to create them based on two data exports that show the changes.

You must validate all of this data, both from the authoritative resource and delta files. To perform user import, the data should be accurate and current, and have consistent formats. Duplicate entries are allowed, but rejected.

Pre-process the Data

To obtain clean user data in the required SPML formats there are several things you can do:

- 1 Find and make use of, or build, scripts and utilities to generate the necessary SPML files. These tools should perform operations such as these:
 - LDIF to SPML conversion
 - CSV to SPML conversion
 - Sending SPML files to Select Identity's web interface
- 2 Use XSL style sheets to perform pre-processing and data conversion, such as removing root users, or generating common telephone number formats.
- 3 Use the SPML Request Filter external call, which can be used for filtering and pre-processing.

It is generally best to avoid manual editing of SPML files, as this is prone to error.



The Sync-In and Sync-Out properties must have been set correctly during attribute mapping. See [Attribute-Level Reconciliation Policy](#) on page 103 for further information. If you are unsure, check those properties now.

In addition, make sure the user import properties in the `TruAccess.properties` file are configured. The user import SPML files have to be placed in the upload directory, unless you run the import manually.

To see an example of an add user request file, refer to the `\SampleXML` directory on the HP OpenView Select Identity product CD.

Configure and Test User Import and Service Assignment

Your authoritative resource contains account information and attributes for each user account used to update all other accounts. For example, your authoritative resource might have an employee number and attributes associated with the employee number (First Name, Last Name, Address, Phone, Social Security). See [Using Authoritative Resources](#) on page 78 for detailed information.


Before building your SPML file, identify your authoritative resource and determine which attributes are to be loaded to Select Identity.

Creating an SPML File Containing Users and Attributes

Many resources today have a utility or mechanism for exporting user data to an XML or SPML format. Create the SPML format needed for user import, using one of the following processes:

- Use the Select Identity SPML Generator tool to convert the data to SPML from CSV and XML inputs. See [Appendix B, The SPML Generator Utility](#), beginning on [page 155](#).
- Export your data in the resource to LDIF format and use a parser to convert the data to SPML. To see an example of a user import file after the LDIF format to SPML conversion, view the sample files located in the `\SampleXML` directory on the HP OpenView Select Identity product CD.
- Export your data in the resource to XML or DSML format. Convert it to SPML using an XML parser and XSLT style sheet.

- Use a third-party mapping tool to convert your data to SPML format.
- Programmatically build the file by reading through your resource and writing out a data record for each user.

 When specifying attributes in the SPML file, be sure to use the mapped resource attribute's name. This may differ from the Select Identity attribute name. Attributes uploaded to Select Identity must be mapped to a resource. For information related to attribute mapping, see [About Attribute Mapping](#) on page 39, and [Appendix C, Attribute Mapping Utility](#), beginning on page 163.


The syntax used must comply with SPML; the semantics differ according to the Select Identity tags. When creating the input file containing the user attributes, specify the unique identifier attribute associated with each user. The `<operationalAttributes xmlns=>` section of the SPML file specifies the identifier and is designated as a value in the `keyFields` attribute. Select Identity's default attribute for identifying accounts is **UserName**. The following is a sample of this section of the SPML file:

```
<?xml version="1.0" encoding="ISO-8859-1"?>
<batchRequest xmlns:countries="countries.uri" smlns:cities="cities.uri"
smlns:dsml
<operationalAttributes xmlns="">
  <attr name="urn:hp:selectidentity#keyFields">
    <value>UserName</value></attr>
</operationalAttributes>
```

The `keyFields` attribute is used to look up the user.

In addition to specifying the operational attribute in the header of the file, you need to specify two operational attribute values for each add user request. The following shows a sample of the SPML file:

```
<addRequest requestID="1">
  <operationalAttributes xmlns="">
    <attr name="urn:hp:selectidentity#taUserName">
      <value>avaughan</value>
    </attr>
    <attr name="urn:hp:selectidentity#taResourceKey">
      <value>AQ4100</value>
    </attr>
  </operationalAttributes>
```

 Select Identity maintains backwards compatibility, so that the form `<attr name="urn:trulogica:concerro:2.0#keyFields">` is still valid.

The `taUserName` field value represents the unique value used to identify each account in Select Identity. The `taResourceKey` uniquely identified the account on the resource from which the user originates.

The file data portion must begin and end with `<batchRequest></batchRequest>`.

Each account to be added begins and ends with `<addRequest></addRequest>`. Select Identity requires the operational attributes and values listed for each add request. An account cannot be added without these attributes and values.

If the `UserName` attribute is set up with a value generation function and a user import request is made from an authoritative resource, the `taUserName` does not need to be specified in the SPML file and the `UserName` should not be specified as `keyField` in the operational Attribute section. Select Identity invokes the value generation function to generate the `UserName`. The user is provisioned in Select Identity with this generated `UserName`.

If another resource attribute is mapped to Select Identity `UserName` and is present in the SPML record, then the `taUserName` is not needed and that resource attribute is used as the `UserName`.

Following is a sample SPML file without the `taUserName`:

```
<batchRequest xmlns:countries="countries.uri"
xmlns:cities="cities.uri" xmlns:dsml="urn:oasis:names:tc:DSML:2:0:core"
xmlns:spml="urn:oasis:names:tc:SPML:1:0" xmlns="urn:oasis:names:tc:SPML:1:0"
requestID="1085774668899">
  <operationalAttributes xmlns="">
    <attr name="urn:trulogica:conceroc:2.0#keyFields">
      <value>Email</value></attr>
    </operationalAttributes>
<addRequest requestID="1">
  <operationalAttributes xmlns="">
    <attr name="urn:trulogica:conceroc:2.0#taResourceKey">
      <value>john.smith@yourdomain.com</value>
    </attr>
  </operationalAttributes>
  <attributes xmlns="">
    <attr name="Employee ID"><value>HP</value></attr>
    <attr name="LastName"><value>Jones-Smythe</value></attr>
    <attr name="Email"><value>john.smith@yourdomain.com</value></attr>
    <attr name="FirstName"><value>John</value></attr>
    <attr name="State"><value>TX</value></attr>
    <attr name="Address2">
      <value>Info Field 1 from Recon file</value>
      <value>Info Field 2 from Recon file</value>
    </attr>
    <attr name="city"><value>Plano</value></attr>
    <attr name="Title"><value>Manager</value></attr>
    <attr name="Business Phone"><value>8886661122</value></attr>
    <attr name="Zip"><value>77777</value></attr>
    <attr name="Address 1"><value>Rolling Drive</value></attr>
    <attr name="Password"><value>abc123</value></attr>
  </attributes>
</addRequest>
</batchRequest>
```

Creating an SPML File Containing Entitlements

After building the SPML file containing your list of users and associated attributes, review the resources containing the entitlements or permissions associated with your users. Users may have entitlements or unique attributes from multiple resources. To upload these entitlements, create a separate SPML file containing the entitlements for each resource. Use one of the methods described in [Creating an SPML File Containing Users and Attributes](#) on page 64 to create this SPML file.

The operational attributes `keyFields`, and `taResourceKey` are required for assigning entitlements. These are specified in the file you created to add users to the system. The attribute `keyFields` is only listed once at the beginning of each file. The attribute `taResourceKey` is listed for each user account. If the attribute `keyFields` value exists in SPML, it is used to locate the user in Select Identity. Otherwise an identifier is used.

For each resource and attribute file created, determine the unique identifier on the resource linking the entitlement or attribute to the designated user. This unique identifier is specified in the SPML file as the `taResourceKey` field. In addition, specify the `userId` or user name so you can associate the entitlements and attributes to the correct Select Identity account. This is designated in the identifier tag as follows:

```
<identifier xmlns="" type="urn:oasis:names:tc:SPML:1:0#UserIDAndOrDomainName">
<id>AEE200</id></identifier>
```

In the example above, the entitlement is associated with an account called `AEE200` in Select Identity.

To see an example file for adding entitlement to an existing user, refer to `\SampleXML` directory on the Select Identity product media.

Uploading User Accounts, Attributes, and Entitlements

You can upload the user accounts, attributes, and entitlements through the **User Import** pages. See [Scheduling User Import](#) on page 67 for details.

- ▶ You can improve the performance of **User Import** tasks by breaking large files into pieces of 5MB or less, and running the files in parallel. Performance improves significantly when running on a multi-CPU server. You should run multiple files in parallel when uploading files for user import. Select Identity processes authoritative resource files before non-authoritative resources.

Scheduling User Import

You can configure Select Identity to add user accounts on a specified date. This process enables Select Identity to add account data to the system from a data file you create. See [Creating an SPML File Containing Users and Attributes](#) on page 64 for information about creating a data file. Connectors and resources must be deployed for systems with identity information you want to import. All necessary resource and Select Identity attributes must be mapped within the connector mapping file and the Select Identity **Attributes** function.

Service Assignment

Scheduling **Service Assignments** functionality associates newly discovered users with existing services in Select Identity. To take advantage of Select Identity, a user must be associated with a service. Service assignment is generally a one-time event and is used in the early phase of establishing the Select Identity environment.

- ▶ Service assignment is the last step of the user import process. Service assignment should only be done after all user accounts and entitlements are loaded into Select Identity. All services should be created before performing this action.

Select Identity assigns all user accounts qualifying for an existing service to the qualifying service automatically. Qualification is based on attribute and entitlement matches for each account. Once you assign services, use Select Identity's **User Management** functionality to maintain user accounts.

The system goes through each user account and evaluates whether that user exists on the specified service's resources. The user must have an account on all resources for the assignment to succeed.

For example, suppose **Service #1** provisions to iPlanet and **Service #2** provisions to iPlanet and SAP. If all other criteria are met, the following are true:

- User1, found in iPlanet only, will be assigned to **Service #1**.
- User2, found in SAP only, cannot be assigned to **Service #1** or **Service #2**.
- User3, found in iPlanet and SAP, will be assigned to **Service #1** and **Service #2**.

Therefore, user import of each resource must be complete before the service assignment process starts.

Checking for Service Membership Requirements

Once added, users can be assigned service memberships based on their defined attributes. For users to be assigned to a service, the following must be true:

- Users must have access to all of the resources that a service requires.
- Users must have all required attribute values for a service.
- Users must have matching values for the service's context attribute.
- Users must have matching fixed attribute values for a service based on context value. The set of all fixed attribute values starting from the current context or business role level all the way up to the root is a subset of the corresponding user attribute set. When no fixed attribute value is defined, this evaluation is skipped.
- Users must have matching optional attribute values for a service based on context value. The optional attribute value set is gathered starting from the current context or business role. When a value set is defined, the lookup process stops. Otherwise, the search continues up to the root level. The optional value set is a super set of the corresponding user attribute set. When no optional attribute value is defined, this evaluation is skipped.
- When an attribute is present, the length and pattern must match the field definition.
- When an attribute is present and the field has a constrained value set, the attribute value must be one of the constrained values. The constrained values are either from the static attribute definition, or an external call function is defined for the attribute.

Schedule Services Assignment

Service assignments must be performed by a Select Identity system administrator who has access to all contexts on all services.

As users are added to the system, you can schedule service access. This is particularly helpful when a new system is coming online and you must set up access for many users at one time. You can do the service assignment early knowing that it will take place on the appropriate go-live date.

See the online help for specific procedures.

Set Up Multiple-User-ID Accounts

An individual user may need multiple identity accounts for a resource. Users may have multiple accounts on a resource based on their role, and want to consolidate and manage these accounts at person level in HP OpenView Select Identity. For example, say John Smith has two UNIX® accounts;

- A UNIX administrator account

- A conventional, non-privileged UNIX user account

Because John as a UNIX administrator has a set of entitlements which are different from his entitlements as a non-privileged user, you may want to consolidate and manage all accounts for John in Select Identity while maintaining his role-based resource account information.

Select Identity provides a consolidation and management capability for multiple-user-ID accounts:

- You can group multiple resource accounts for a person into a multiple-user-ID account containing primary and secondary users.
- You can manage (add, modify, or delete) entitlements of a person based on roles
For example, John as a UNIX administrator has privileged access to all accounts on UNIX, while his non-privileged account has restricted access
- You can transfer accounts from one person to another. For example, John's UNIX administrator account can be transferred to another person, while retaining his existing non-privileged UNIX user account
- You can terminate a single account, or all of a user's resource accounts

In addition, multiple-user-ID accounts let you support situations where a person's multiple accounts on a resource are maintained independently of each other on the resource, but need to be linked externally using Select Identity.

Each user in Select Identity has an internal account identifier. A user can have a primary account, with multiple secondary accounts associated with it. Primary and secondary accounts can be established one-at-a-time, or multiple accounts can be added by using the import and reconciliation features. After a secondary account grouping is established, it can be transferred to another primary account. For example, if a staffing or role change occurs, you can move the secondary accounts to another primary account.

Additions and changes to primary and secondary user accounts go through the standard approval workflows. If the approver rejects the primary account request, the secondary requests are also rejected. Secondary accounts cannot exist without the primary account.

Users with multiple-user-ID accounts are managed through the same functions as normal users. If a user has multiple accounts, the accounts are listed on the user's **User Profile** and **Services** tabs.

Primary Accounts

The primary account in Select Identity is the main user account for the individual. A user can only have one primary account. Select Identity can only be accessed by the primary account user.

Secondary Accounts

Secondary accounts are used to access resources. Secondary user accounts are associated with a primary account. The resources, services, and entitlements for the secondary account can vary from the primary account. Select Identity establishes secondary accounts in one of two ways: through the **Service Subscriptions** function, or through the batch processing functions - Reconciliation, Bulk Add, User Import, and Web Services. Secondary accounts are only valid for resources and cannot be used to access Select Identity. A secondary account cannot be added to another secondary account.

Creating Multiple-User-ID Accounts

Follow the instructions in the Select Identity online help to set up multiple-user-ID accounts.

Example of a Multiple-User-ID Account

In **Figure 8**, user Alan F. Smith has a Select Identity user account of AlanF. On this primary account he has access to two resources through assignments to Service 1 and Service 2. He has three accounts on Resource 1, one of which, AlanF, is his primary account. He also has three secondary accounts on Resource 2. Each of the secondary accounts has a different entitlement on the resource.

Figure 8 Example of Multiple-User-ID Accounts

SI User Id: AlanF

Service Memberships: Service 1, Service 2

Service 1

(SI Acct Id: AlanF) PRIMARY

Resource1: First: Alan
Last Name: Smith
Cost Center 99
Entitlement: Test

Resource2: Desc: AlanNI
Last Name: Smith
Region: NI
Entitlement: FR

(SI Acct Id: AlanS)

Service 1 has Resource 1 & 2

Service 2 has Resource 2

Resource1: First: Alan2
Last Name: Smith
Cost Center 20
Entitlement: Sales

Resource2: Desc: AlanNE
Last Name: Smith
Region: NE
Entitlement: US

Service 2

(SI Acct Id: Smith423BA)

Resource2: Desc: AlanBA
Last Name: Smith
Region: BA
Entitlement: AR

User "AlanF" has 3 accounts on 2 different resources (Resource 1 and Resource 2)

Resource 1

Account 1	Account 2	Account 3
Id: AlanF	Id: AlanS	Id: AlanTest
First: Alan	First: Alan2	First: Alan
Last Name: Smith	Last Name: Smith	Last Name: Smith
Cost Center: 10	Cost Center: 20	Cost Center: 99
Entitlement: Finance	Entitlement: Sales	Entitlement: Test

Resource 2

Account 1	Account 2	Account 3
Id: Smith423	Id: Smith423BA	Id: Smith423NI
Desc: AlanNE	Desc: AlanBA	Desc: AlanNI
Last Name: Smith	Last Name: Smith	Last Name: Smith
Region: NE	Region: BA	Region: NI
Entitlement: US	Entitlement: AR	Entitlement: FR

Importing and Updating Multiple-User-ID Information

Multiple accounts for a person on one resource can be consolidated and managed in Select Identity. As a result of consolidation, multiple accounts for a person on one resource are represented in Select Identity using one primary and one or more secondary accounts.

Account consolidation can happen when users are imported into Select Identity via user import. (This is also possible via reconciliation; see [Multiple-User-ID Accounts and User Reconciliation](#) on page 106).

This section explains how to import and update multiple-user-ID accounts using SPML `addRequests`.

Multiple-User-ID Accounts and Import

Multiple-user-ID accounts can be added or updated through the import process, and like any import, can use one or more authoritative resources. Creating or updating a multiple-user-ID account uses the same process as importing any account.

Additional information is required in the SPML file for multiple-user-ID accounts, including two new attributes, `primaryAcctKey` and `primaryAcctValue`:

- `primaryAcctKey` – indicates the attributes that will be used as keys to identify the primary account.
- `primaryAcctValue` – contains the values of attribute names indicated by `primaryAcctKey` above. For multiple key values, the order of `primaryAcctKey` and `primaryAcctValue` must be maintained as shown in this example:

```
<attr name="urn:hp:selectidentity#primaryAcctKey">
  <value>Email</value>
  <value>Employee ID</value>
</attr>
<attr name="urn:hp:selectidentity#primaryAcctValue">
  <value>john.doe@abc.com</value>
  <value>abc</value>
</attr>
```

You determine the `primaryAcctKey` based on your organization's needs. For example, you could use the `Email_Address` attribute as the primary key. You can also combine multiple attributes to build the primary key. For example, you could combine the `FirstName` with `Address_1` to produce a unique primary key.



- Select Identity user IDs are created for new resource accounts, either generated by Select Identity or assigned in the batch file.
- If the primary resource account cannot be found, the add request for the secondary account is rejected.
- If there is no `primaryAcctValue` specified in the SPML file, the add account request is treated as a normal account (not a multiple-user-ID account).

Typically, a primary account represents a user's profile information. The primary account must exist prior to the addition of the secondary account. You can add the primary account prior to import, or include the primary account in the import file before the secondary account.

During user import, secondary accounts can be created *implicitly*, using `keyFields` information in the input SPML. When adding a user in Select Identity this way, if the `keyFields` in the `operationalAttributes` of the SPML identifies an existing user in Select Identity, the existing user account is designated as the primary account, and the new user account is designated as a secondary account. The first new account on a resource is merged, if applicable, with the existing user account in Select Identity, and no primary or secondary accounts are created. Subsequent accounts for the same user on that resource are created in Select Identity as secondary accounts.

In addition, secondary accounts can be created *explicitly*, using `primaryAcctKey` and `primaryAcctValue` tags. To do this, you explicitly specify the `primaryAcctValue` in the operational attributes of secondary user in the input SPML. Unlike using the `keyFields` approach, this explicitly creates a secondary account irrespective of whether it is the first account for the user on the resource or not.

For more information about the import process, see [Data Processing](#) on page 62.

Examples



User import uses resource attribute names, not Select Identity attribute names.

These examples use a fictional person, Joe Smith, who is an employee of two different sister companies, namely CompanyABC and CompanyXYZ.

Import from an Authoritative Resource

The following sample SPML file uses `addRequests` to import Joe Smith's multiple HR resource account information. Secondary accounts are created implicitly using `keyFields` information in the input SPML.

After the import, user Joe will have one account in Select Identity with two resource keys: one for CompanyABC and the other for CompanyXYZ.

```
<?xml version="1.0" encoding="ISO-8859-1"?>
<batchRequest xmlns:countries="countries.uri" xmlns:cities="cities.uri"
xmlns:dsml="urn:oasis:names:tc:DSML:2:0:core" xmlns:spml="urn:oasis:names:tc:SPML:1:0"
xmlns="urn:oasis:names:tc:SPML:1:0" requestID="1085774668899">
  <operationalAttributes xmlns="">
    <attr name="urn:hp:selectidentity#keyFields">
      <value>CompanyABC_HR_ID</value>
    </attr>
  </operationalAttributes>

  <addRequest requestID="1">
    <operationalAttributes xmlns="">
      <attr name="urn:hp:selectidentity#taResourceKey">
        <value>CompanyABC/001</value>
      </attr>
    </operationalAttributes>

    <attributes xmlns="">
      <attr name="LastName"><value>Smith</value></attr>
      <attr name="FirstName"><value>Joe</value></attr>
      <attr name="DateOfBirth"><value>1968-07-23</value></attr>
      <attr name="Gender"><value>Male</value></attr>
      <attr name="Address 1"><value>Address Line 1</value></attr>
      <attr name="CompanyABC_HR_ID"> <value>CompanyABC/001</value></attr>
      <attr name="TypeOfHR"><value>Employee of company CompanyABC</value></attr>
      <attr name="BeginDate"><value>1993-01-01</value></attr>
      <attr name="ORGUnit"><value>Research</value></attr>
      <attr name="Password"><value>password1</value></attr>
    </attributes>

  </addRequest>
</batchrequest>
```

```
<?xml version="1.0" encoding="ISO-8859-1"?>
<batchRequest xmlns:countries="countries.uri" xmlns:cities="cities.uri"
xmlns:dsml="urn:oasis:names:tc:DSML:2:0:core" xmlns:spml="urn:oasis:names:tc:SPML:1:0"
xmlns="urn:oasis:names:tc:SPML:1:0" requestID="1085774668899">
  <operationalAttributes xmlns="">
    <attr name="urn:hp:selectidentity#keyFields">
      <value>FirstName</value>
      <value>LastName</value>
    </attr>
  </operationalAttributes>

  <addRequest requestID="1">
    <operationalAttributes xmlns="">
```



```

    <attr name="urn:hp:selectidentity#taResourceKey">
      <value>CompanyXYZ/001</value>
    </attr>
  </operationalAttributes>

  <attributes xmlns="">
    <attr name="LastName"><value>Smith</value></attr>
    <attr name="FirstName"><value>Joe</value></attr>
    <attr name="CompanyXYZ_HR_ID"><value>CompanyXYZ/001</value></attr>
    <attr name="ORGUnit"><value>Planning</value></attr>
    <attr name="Password"><value>password2</value></attr>
  </attributes>
</addRequest>
</batchrequest>

```

Import from a Non-Authoritative Resource

The following sample SPML includes `addRequests` to import Joe Smith's multiple accounts from a given non-authoritative resource. Secondary accounts are created explicitly using `primaryAcctKey` and `primaryAcctValue` tags.

During this import, Joe's existing normal account in Select Identity, created using the above SPML, is designated as a primary account, and two new secondary accounts are created to represent each of his resource accounts.

```

<?xml version="1.0" encoding="ISO-8859-1"?
<batchRequest xmlns:countries="countries.uri" xmlns:cities="cities.uri"
xmlns:dsml="urn:oasis:names:tc:DSML:2:0:core" xmlns:spml="urn:oasis:names:tc:SPML:1:0"
xmlns="urn:oasis:names:tc:SPML:1:0" requestID="1085774668899">
  <operationalAttributes xmlns="">
    <attr name="urn:hp:selectidentity#keyFields">
      <value>RescID</value>
    </attr>
  </operationalAttributes>

  <addRequest requestID="1">
    <operationalAttributes xmlns="">
      <attr name="urn:hp:selectidentity#taResourceKey"><value>RescID1</value></attr>
      <attr name="urn:hp:selectidentity#primaryAcctKey">
        <value>CompanyABC_HR_ID</value>
      </attr>
      <attr name="urn:hp:selectidentity#primaryAcctValue">
        <value>CompanyABC/001</value>
      </attr>
    </operationalAttributes>

    <attributes xmlns="">
      <attr name="RescID"><value>RescID1</value></attr>
      <attr name="FirstName"><value>Joe</value></attr>
      <attr name="LastName"><value>Smith</value></attr>
      <attr name="Password"><value>password3</value></attr>
      <attr name="urn:hp:selectidentity#groups">
        <value>MailAccount</value>
        <value>PortalAccount</value>
      </attr>
    </attributes>
  </addRequest>

  <addRequest requestID="2">
    <operationalAttributes xmlns="">
      <attr name="urn:hp:selectidentity#taResourceKey"><value>RescID2</value></attr>
      <attr name="urn:hp:selectidentity#primaryAcctKey">
        <value>CompanyABC_HR_ID</value>
      </attr>
      <attr name="urn:hp:selectidentity#primaryAcctValue">
        <value>CompanyABC/001</value>
      </attr>
    </operationalAttributes>
  </addRequest>

```

```

    </attr>
  </operationalAttributes>

  <attributes xmlns="">
    <attr name="RescID"><value>RescID2</value></attr>
    <attr name="FirstName"><value>Joe</value></attr>
    <attr name="LastName"><value>Smith</value></attr>
    <attr name="Password"><value>password4</value></attr>
    <attr name="urn:hp:selectidentity#groups">
      <value>MainframeMailAccount</value>
    </attr>
  </attributes>
</addRequest>

</batchRequest>

```

Bulk Operations

Select Identity allows uploading of multiple user accounts to multiple services. This lets you populate your system without having to add hundreds or thousands of individual user accounts. Bulk can be used to create new users, and to add services to existing users.

User accounts are uploaded to the system using an SPML data file. The data file maps all Select Identity attributes defined for a service to the new accounts.



Unlike reconciliation and user import, bulk operations use Select Identity attribute names, not resource attribute names.

This section covers the following:

- Bulk Dependencies
- Bulk Procedure Overview

Bulk Dependencies

Before running a bulk job, ensure the following dependencies are met:

- Connectors and resources are deployed for systems for which you want to upload data.
- All necessary resource and Select Identity attributes are mapped within the connector mapping files and Select Identity **Attributes** function.
- One or more services are created to use the resources with which you want to upload data and the default workflow template for bulk jobs (`SIBulkOneStageApproval`) is associated in the service. You can also create and assign a custom template. See the workflow studio online help for information about workflow templates.
- Three event handlers are added in the service you want to add to users:
 - Bulk - Add New User
 - Bulk - Add Service
 - Bulk - Modify User

The event handler is set in the service role to validate the request. If it is not added, the bulk job fails.

- Make sure the following is in place when creating the SPML data file:
 - Unlike reconciliation and user import, bulk operations use Select Identity attribute names, not resource attribute names.

- The file name must begin with an underscore (`_`) if it is used by an automated job and stored in the `reconroot` directory. Select Identity reads data files from the `reconroot` directory and the underscore lets the system differentiate reconciliation files from bulk upload files. If the files are uploaded as onetime task, there are no naming restrictions.
- You can specify the services to be added to users in the **Job Submission** page, or in the data file. In the data file, services can be specified at batch operational attribute or request operational attribute levels. Request level has highest precedence if services are specified at all levels.

Bulk Procedure Overview

Select Identity bulk jobs use the SPML file of the type discussed in [Create an SPML File Containing Users and Attributes](#) on page 75. This file should include the new account information and attributes required by the services to which you are adding. The file is then uploaded to Select Identity.



You must have the Select Identity system administrator role with bulk permission granted to perform bulk tasks.

You may need to increase the JTA time-out seconds to 300 on WebLogic for bulk jobs to work properly.

This section shows you how to perform bulk jobs.

Create an SPML File Containing Users and Attributes

All attributes specified in this file are Select Identity attributes, not resource attributes. The `requestID` for each request must be unique to reflect properly in the results report. When the request fails or the user name cannot be parsed, Select Identity uses the `requestID` to indicate the error location in the original SPML file.

The file must begin and end with `<batchRequest></batchRequest>`. Each account to be added begins and ends with `<addRequest></addRequest>`.

Use the following methods to specify the services:

Specify the services you want assigned to users in the batch level of the file to add all users to all add requests in the file.

- Specify a group of common services listed in the batch level of the file, and add others specific to the user within the user's add request (as shown below). Request level services take precedence over those at the batch level.
- If you specify Service 1 in the batch level and Service 2 in the request level, the user is added to both services.



The attributes listed for each account are the Select Identity attribute names defined in this service through the **Service** pages. The attribute name must match the field name exactly. If a required field is missing or a data field does not meet the service constraints, an exception is listed in the results file. See [Defining Resources and Attributes](#) on page 37 for more information.

When creating the data file containing the user attributes, specify the unique identifier attribute associated with each user. The `<operationalAttributes xmlns=>` section of the SPML file specifies the identifier and is designated as a value in the `keyFields` attribute. Select Identity's default attribute for identifying accounts is **UserName**. The following is a sample of this section of the SPML file:

```
<?xml version="1.0" encoding="ISO-8859-1"?>
```

```
<batchRequest xmlns:countries="countries.uri" xmlns:cities="cities.uri"
xmlns:dsml="urn:oasis:names:tc:DSML:2:0:core" xmlns:spml="urn:oasis:names:tc:SPML:1:0"
xmlns="urn:oasis:names:tc:SPML:1:0" requestID="1085774668899">
```

Batch Level

```
<operationalAttributes xmlns="">
  <attr name="urn:trulogica:concerro:2.0#keyFields">
    <value>UserName</value>
  </attr>
  <attr name="urn:trulogica:concerro:2.0#serviceName">
    <value>dkLDAP70</value>
    <value>dkLDAP72</value>
    <value>dkCombo</value>
  </attr>
</operationalAttributes>
```

The "urn:trulogica:concerro:2.0#keyFields" operational attributes specify the field in an individual request to use to check for the existence of a user in Select Identity. If this field is not provided, no check is performed and the job generates a create user internal event. If a the user already exists, the job generates an add service internal event.

In addition to specifying the operational attribute in the header of the file, you can specify operational attribute values for the services you want assigned for each add user request:

Request Level

```
<addRequest requestID="1">
  <operationalAttributes xmlns="">
    <attr name="urn:trulogica:concerro:2.0#serviceName">
      <value>FinanceService</value>
    </attr>
  </operationalAttributes>
  <attributes xmlns="">
    <attr name="UserName">
      <value>JohnB</value>
    </attr>
    <attr name="Password">
      <value>abc123</value>
    </attr>
    <attr name="Email">
      <value>johnb@company.com</value>
    </attr>
  </attributes>
</addRequest>
```

Example: Adding Users to Services with Common Attributes

The following example adds one user to the Finance Service and another user to the Finance and Market Services. The listed attributes are required attributes for the services.

```
<?xml version="1.0" encoding="ISO-8859-1"?>

<batchRequest xmlns:countries="countries.uri" xmlns:cities="cities.uri"
xmlns:dsml="urn:oasis:names:tc:DSML:2:0:core" xmlns:spml="urn:oasis:names:tc:SPML:1:0"
xmlns="urn:oasis:names:tc:SPML:1:0" requestID="1085774668899">
<operationalAttributes xmlns="">
<attr name="urn:trulogica:concerro:2.0#keyFields"><value>UserName</value></attr>
</operationalAttributes>

<addRequest requestID="1">
  <operationalAttributes xmlns="">
    <attr name="urn:trulogica:concerro:2.0#serviceName">
      <value>Finance Service</value>
    </attr>
```

```

</operationalAttributes>

<attributes xmlns="">
  <attr name='Company'>
    <value>YOUR.COM</value>
  </attr>
  <attr name='UserName'>
    <value>newuser1</value>
  </attr>
  <attr name='Password'>
    <value>book1</value>
  </attr>
  <attr name='FirstName'>
    <value>David</value>
  </attr>
  <attr name='LastName'>
    <value>d'Antonio</value>
  </attr>
  <attr name='Email'>
    <value>newuser1email@yourcom.com</value>
  </attr>
  <attr name='State'>
    <value>NY</value>
  </attr>
  <attr name='urn:trulogica:concero:2.0#groups:Resource1'>
    <value>West Washington</value>
    <value>West California</value>
  </attr>
</attributes>
</addRequest>
<addRequest requestID="1">
  <operationalAttributes xmlns="">
    <attr name="urn:trulogica:concero:2.0#serviceName">
      <value>Finance Service</value>
      <value>Market Service</value>
    </attr>
  </operationalAttributes>

  <attributes xmlns="">
    <attr name='Company'>
      <value>YOUR.COM</value>
    </attr>
    <attr name='UserName'>
      <value>newuser2</value>
    </attr>
    <attr name='Password'>
      <value>book1</value>
    </attr>
    <attr name='FirstName'>
      <value>Kathleen</value>
    </attr>
    <attr name='LastName'>
      <value>O'Bryan</value>
    </attr>
    <attr name='Email'>
      <value>newuser2email@yourcom.com</value>
    </attr>
    <attr name='State'>
      <value>NY</value>
    </attr>
    <attr name='urn:trulogica:concero:2.0#groups:Resource1'>
      <value>Group4</value>
      <value>Group5</value>
    </attr>
  </attributes>
</addRequest>

```

</batchRequest>

Example: Adding Users to Services with Specified Entitlements

Users can be added to multiple services sharing a common resource. In order to specify service specific entitlements unambiguously, the following convention is used:

#serviceName#<value-of-servicename>#<resourcename>_ENTITLEMENTS

In the following example, the user name is generated (provided user name generation is turned on):

```
<?xml version="1.0" encoding="ISO-8859-1" ?>
  <batchRequest xmlns:countries="countries.uri" xmlns:cities="cities.uri"
xmlns:dsml="urn:oasis:names:tc:DSML:2:0:core" xmlns:spml="urn:oasis:names:tc:SPML:1:0"
xmlns="urn:oasis:names:tc:SPML:1:0" requestID="1085774668899">
  <operationalAttributes xmlns="" />
  <addRequest requestID="1">
    <operationalAttributes xmlns="">
      <attr name="urn:trulogica:conceroc:2.0#serviceName">
        <value>Service1</value>
        <value>Service3</value>
      </attr>
    </operationalAttributes>
    <attributes xmlns="">
      <attr name="Email">
        <value>bulk1@company.com</value>
      </attr>
      <attr name="FirstName">
        <value>Bulk1</value>
      </attr>
      <attr name="LastName">
        <value>Bulk</value>
      </attr>
      <attr name="State">
        <value>TX</value>
      </attr>
      <attr name="Company">
        <value>TL</value>
      </attr>
      <attr name="LDAP70_ENTITLEMENTS">
        <value>$UNIX1</value>
      </attr>
      <attr name="urn:trulogica:conceroc:2.0#serviceName#Service1#LDAP70_ENTITLEMENTS">
        <value>$UNIX2</value>
      </attr>
      <attr name="urn:trulogica:conceroc:2.0#serviceName#Service3#LDAP70_ENTITLEMENTS">
        <value>$UNIX3</value>
      </attr>
    </attributes>
  </addRequest>
</batchRequest>
```

Upload Data Files

Upload the data files including user accounts, attributes, and entitlements through the **Bulk** pages. See the online help for a complete procedure.

Viewing Job Results

As each of the jobs completes, the requestor receives an emailed report about it. (You can also use **Tools** → **Bulk** → **Bulk Task List** and request a report.) The report lists users that were successfully created and those that failed. Use this report to make any needed corrections to your SPML file and resubmit the file with only those accounts that failed. You must create a new job with a unique name to upload the file in the Select Identity client.

- ▶ You cannot give a new job the same name as the existing one. Each job must be assigned a unique name.

The XML file can be used to extract failed users to be resubmitted after corrections are made.

Scheduling and Managing Bulk Jobs

Bulk jobs provide a means of adding or moving users in large blocks. Provision blocks of users using the bulk add functionality. Use bulk move to schedule bulk jobs to move a context user group from one context to another.

For example, if the Northwest division closes and all the employees within that division move to the North division, move the users in one block using bulk move.

A bulk move is a job that runs one time. For example, if the context attribute for a service is “City” and you need to move a division in your company from one city to another, you can do so with the bulk function.

Changing a user’s context may remove access to a service or modify the entitlements granted within the service. A user cannot be added to a service with this function. A user is added to services as the external calls, rules, and workflow are defined.

- ▶ To perform this task, you must have administrative rights to the services affected by the context change. You need bulk move permission granted.

See the online help for specific procedures to schedule and manage bulk add and bulk move jobs.

Bulk Move

You can move a group of users from one context to another. This is a job that runs one time. For example, if the context attribute for a service is `City` and you need to move a division in your company from one city to another, you can do so with the bulk function.

Changing a user’s context may remove access to a service or modify the entitlements granted within the service. A user cannot be added to a service through a bulk move. A user is added to services as defined by the external calls, rules, and workflows.

- ▶ To perform this task, you must have administrative rights to the Services affected by the context change. You need bulk move permission granted.

Carry-over of Existing Entitlements

In many cases, when you move users from one context to another, you will want the users to retain any existing entitlements that are valid or available for their new context group. In other cases, you may prefer that users carry over none of their existing entitlements.

During service configuration, you can specify for each attribute (with the exceptions noted below) which if any entitlements are to be carried over in a bulk move. This policy is configurable from the **Attribute Properties** page of service management. The choices for this policy are as follows:

Transfer on Move (the default policy)	<ul style="list-style-type: none">• The user will get fixed entitlements according to the target context group.• For bulk user move, the user will carry over all existing entitlements that exist in the optional list of the target context group.• For a single user move, you have the option to carry over all existing entitlements that exist in the optional list of target context group. Those will be selected in confirmation page.
Not Transfer on Move	<ul style="list-style-type: none">• The user will get fixed entitlements according to the target context group.• For single user move, all optional entitlements will be unselected. You will have the option to select entitlements in confirmation page.• For bulk user move, the user will not get any entitlements from the optional list of the target context group.

At any service role, a user gets all fixed entitlements defined in that level or its parents up to the root service role.

Also, if there is no optional value defined in any level of service role, the user inherits all optional values from the parent service role. The root service role inherits values from the service level if any are defined; otherwise it inherits values from attribute level.



The transfer on move policy applies for all service attributes except the following:

- The context attribute of the service
- Resource keys
- SI Username
- SI Password
- GUID
- RoleServiceContext
- SIService
- SIAdminRole
- FirstName
- LastName
- Email
- ExpirationDate
- ExpirationProcessPeriod

Any changes are applied in the next move operation. Also, the move policy does not apply to a move based on reconciliation.

See the online help for detailed procedures to run a Bulk Move.


Updating Multiple-User-ID Accounts through Bulk Add

The bulk add feature in Select Identity can be used to add multiple accounts to services. Just as with normal user accounts, multiple-user-ID accounts may be assigned to services during bulk-add. See [Bulk Operations](#) on page 74 for more information about bulk operations.

Two additional attributes are required when adding multiple-user-ID accounts:

- `primaryAcctKey` – Indicates the information that will be used to set the primary account value. If the default for `primaryAcctKey` is not specified, `UserName` will serve as the default value.
- `primaryAcctValue` – Indicates the primary account to be associated with a secondary account.

The snippet that follows illustrates the use of `primaryAcctKey` and `primaryAcctValue` to create a primary and secondary account during bulk add. The `operationalAttributes` section in the second request uses the `primaryAcctValue` attribute to indicate that this user will be added as a secondary user with the primary user identified by `company=TL` and `Email=bulk1@corp.com`. The `primaryAcctKey` specified at the batch level serves as the default. The primary user is created in the first request.

 The attributes shown in the snippet are Select Identity attributes.

```
.....
<operationalAttributes xmlns="">
  <attr name="urn:hp:selectidentity#keyFields"><value>PersonNumber</value></attr>
  <attr name="urn:hp:selectidentity#primaryAcctKey">
    <value>Company</value>
    <value>Email</value>
  </attr>
</operationalAttributes>

<addRequest requestID="1">
  <operationalAttributes xmlns="">
    .....
  </operationalAttributes>
  <attributes xmlns="">
    <attr name="Company"><value>TL</value></attr>
    <attr name="Email"><value>bulk1@corp.com</value></attr>
    .....
  </attributes>
</addRequest>

<addRequest requestID="2">
  <operationalAttributes xmlns="">
    <attr name="urn:hp:selectidentity#primaryAcctValue">
      <value>TL</value>
      <value>bulk1@corp.com</value>
    </attr>
  </operationalAttributes>
  <attributes xmlns="">
    .....
  </attributes>
</addRequest>
```

For more information about the bulk add process, see [Bulk Operations](#) on page 74, and the [Select Identity online help](#).

Set Up Service Reconciliation

Select Identity allows you to reconcile user accounts following certain changes to a service that are not automatically visible to current users of the service:

- Adding resources to the service
- Deleting resources from the service
- Adding fixed attributes or entitlements to a service role
- Deleting optional attributes or entitlements from the service or a service role

Service reconciliation updates the service users so that their resources and attributes match those in the new version of the service, as if they were added to the service after it changed. This relieves you from manually changing every user profile before or after every service modification.

Service reconciliation is performed from the **Modify Service** page. See the online help for procedure details. The only setup involved for service reconciliation is to define the `SVCCHGRECON:Modify User` event for each service to be reconciled.



When modifying a service that you want to reconcile, make all of your changes to the service first. Then reconcile once rather than reconciling multiple times. When service reconciliation runs, it generates a single parent request, which contains a child request for each user in the service.

When service reconciliation is complete, the administrator who requested it receives an email containing a reconciliation report.

4 Testing and Deploying Your Solution

The methodology presented in [Chapter 2, Designing and Planning Your Solution](#) has three major phases:

- 1 **Design and plan**
- 2 **Develop and build**
- 3 **Test and deploy**

This chapter explores the third phase. In performing the development and configuration steps in [Chapter 3](#), you implemented the design you created in [Chapter 2](#). The tasks in the third phase ensure that your solution is robust, and that your organization is prepared to begin using it. Finally, there are specific tasks to smoothly move your solution from the development environment into production.

Overview of the Test and Deploy Phase

This section introduces the basic tasks for testing and deploying your solution, which later sections cover in detail. This phase consists of two distinct and substantial tasks:

- 1 **Testing and Deployment Preparation** The objective of this task is to perform thorough testing of the solution, and to thoroughly document the solution to ensure that future design enhancements are aligned with the existing solution.
- 2 **Deployment into Production** The objective of this step is to deploy the final solution into production in the organization.

These two tasks should not overlap. That is, you should be fully satisfied with the quality of the solution, and have your organization properly prepared, before you bring the system into production.

Testing and Deployment Preparation


This task has two major areas of focus: testing, and deployment preparation.

Testing

There are several kinds of tests you should perform at this point:

- Integration testing
- User acceptance testing
- Load and performance testing
- Disaster recovery testing

You should create a formal test plan, with test cases, test data definitions, and expected results. Take care during testing to exercise all test cases and document the actual results.

 The test environment must have the same complexity as the production environment. This is crucial to ensure that your testing delivers meaningful results. Some defects or design shortfalls only become apparent when the solution faces full-scale pressures.

Integration Testing

This is also called system testing. Your test plan should call for a thorough exercise of all the functionality you have created. Testing should ensure that all resources behave as expected.

User Acceptance Testing

There will probably be several different categories of users of your solution. Test it with representatives from each category to make sure that your solution meets the needs and expectations of each. User acceptance testing could include members of any of the following groups:

- End users, such as requesters, approvers, and so on
- Administrators, such as IT operations, help desk, and others
- IT management and IT security personnel
- Other groups according to the way Select Identity will be used in your organization.

Ideally, each major group of users will specify what they need and expect from your solution, with some degree of detail. If that does not happen, you should return to your design criteria and make sure that, in writing your test plan, all key user groups are addressed.

Load and Performance Testing

Load and performance tests help you verify if the planned hardware can meet the load and performance requirements of your solution. These tests also help verify that all configuration parameters have been set correctly for the expected load. Parameters of interest include queue sizes, memory configuration, database pool sizes, the number of database processes, and so on.

Your plan should test the stability of the overall system, especially custom connectors and external calls. Custom components are most likely to reveal any defects during load testing.

Typical issues encountered during load testing include situations such as these:

- Finding that there is an insufficient number database processes on the server or target systems, or that disk space is inadequate.
- Finding that performance limitations in some area, such as a custom connector, slows down the provisioning process.
- Discovering that an external call does not fail gracefully, or generates inconsistent results when multiple threads run.

Disaster Recovery Testing

Any mission-critical solution should be hardened against the possibility of disasters of any scale. To protect the integrity of your identity management solution, you should perform failover testing for all components:

- Databases
- Application servers
- Web servers

Make sure that you can restore your environment from backups, and that you are able to recover from an abnormal server shutdown. See [Appendix 6, Recovering from Abnormal Server Shutdown](#)).

You should generate a disaster recovery plan that accounts for different scenarios, ranging from a simple power outage to catastrophes like fire or flood.

Deployment Preparation

Preparation for deployment involves documenting the solution, and training your users and support staff.

Documenting the Project

In preparing for deployment, make sure that the project documents are complete, up to date, and appropriately available to all interested parties.

Good project documentation includes the following:

- Detailed discovery documentation
- Design documentation
- Test specification, test plans, and test results
- Status reports
- An “Operations Manual” for each major class of users, which describes how to achieve their objectives using your solution
- A record of change orders
- Issue logs
- Product documentation and release notes
- Links to HP support and online documentation

Training

Training should take place before migrating your solution into production. It is very likely that your solution will be new to many of your users, and cause a more or less significant change in some of their duties and processes. The following kinds of training should be provided:

- Overview training for managers who need to understand the final solution. These managers should be given a good overview of the new identity management solution, and apprised of any significant changes to the original plan.
- End user training and informational resources, so that your whole organization understands or can at least obtain information about the new identity management solution. For example, users with administrator roles need to know the steps required to delegate administrative duties.

IT operations and other support resources need thorough training on your solution. Product training on Select Identity is a prerequisite to solution training.

If someone else will be maintaining your solution, he or she will need in-depth solution training with an architecture overview covering all components. Detailed developer training is required for anyone who will maintain custom code for external calls and connectors.

Some of your users, particularly any who have been deeply involved in the prior stages of the project, may have already received sufficient training. Otherwise, if ultimate owner of your solution has not been an active participant in project, now is the time to get them trained on the product.

Others who are occasional users of the identity management solution will need a limited amount of focused “end-user” training, and on-demand access to targeted information.

To accomplish a smooth transition from your existing identity management solution to your new one, you need to address training for your users. This might include short training session like the following:

Deployment into Production

It is now time to deploy your solution, and put HP OpenView Select Identity to work in your organization.

There are four major steps in deployment:

- 1 Install and set up Select Identity in the production environment. This is fully covered in the *HP OpenView Select Identity Installation and Configuration Guide*, so this section does not touch on it.
- 2 Replicate the development configuration in the production environment
- 3 Set up the actual users of the solution
- 4 Conduct final system check and “Go Live!”

Configuration Replication

Once you have HP OpenView Select Identity installed and ready in your production environment, you can replicate your development configuration there.



During deployment, follow these tips:

- Make sure any load balancing and failover mechanisms do not adversely affect your solution.
- Be sure to harden the systems according to the security policies of your organization.
- Firewalls are likely to cause problems and require reconfiguration to support Select Identity.

Select Identity lets you replicate the following configuration types from one environment into another:

- Attributes
- Notification templates
- Request instance reports
- Resources
- Services
- Workflow application definitions
- Workflow demplates
- Administrative roles
- Configuration approval setup
- Connectors
- External calls
- Rules

Use the **Export / Import Configuration** functionality to export the configuration to your production environment using XML files. All data is imported and exported through XML files.



You can replicate your system configuration securely by using the Select Identity secure object migration feature. This assures that the destination environment can import configurations only from a trusted source.

See [Secure Migration](#) on page 91, the *HP OpenView Select Identity Installation and Configuration Guide*, and the Select Identity online help for details.

At a high level, the process of replicating a Select Identity configuration follows these steps:

- 1 Export the XML configuration file.
- 2 Edit the file.
- 3 Import the XML file into the target application.
- 4 Verify that all dependent configurations have been imported.
- 5 Test the import to make sure that the data behaves as expected.

This section covers the following:

- Exporting Configurations
- Importing Configurations
- Secure Migration
- Troubleshooting Configurations

Exporting Configurations

Configurations can be exported in any order. However, to make sure that you don't miss any dependent configurations, you should export configurations in the order your services were created.



Create a directory to store your exported XML files so you can easily validate that you captured all configurations necessary.

To export a configuration, perform steps provided in the online help.

Editing an XML file

HP OpenView Select Identity exports all configurations as XML files. These files may be edited and changed following conventional XML editing techniques.



To prevent the need for manual edits in the import files, avoid different naming conventions in the two environments. However, resource-specific information, such as IP addresses, needs to be changed to match the production environment.

For example, you may export a service configuration that matches the service you are creating exactly, except for the service name. Change all references to the service name in the XML file, then import the configuration file into the target application.

Errors in editing an XML file, such as changing a service name in one reference but not in all, will cause the imported data to generate application errors once imported into the target HP OpenView Select Identity application.

Importing Configurations

All configurations can easily be imported from one instance to another using the instructions in the online help.

Understand Dependencies

Exported configuration files that do not contain errors may import into HP OpenView Select Identity without issues. However, the successful import of data does not necessarily mean that the functionality associated will work as expected.

All configurations and any other supporting files on which the function depends must be present in order for the imported functionality to work as planned.


For example, resource configurations can depend on attribute configurations when a resource attribute is mapped to an HP OpenView Select Identity attribute. Take great care to export all dependent configurations. The dependencies listed here are what these objects reference, not what references them:

Functional Area	Dependencies
Attributes	No dependencies
Connectors	No dependencies
External Calls	No dependencies
Administrative Roles	No dependencies
Notification Templates	No dependencies
Request Instance Report	No dependencies. Note: There is a default report, called <code>DefaultReport</code> , that may be version specific. It should not be imported from other versions of Select Identity. Request assistance from HP if you need to import this configuration from a different version.
Workflow Application Definitions	No dependencies. Note: There is a default, called <code>wfAppDefine</code> , that may be version specific. It should not be imported from other versions of Select Identity. Request assistance from HP if you need to import this configuration from a different version.
Resources	Attribute configurations associated with the resource. Rules used to provision the resource during reconciliation.
Services	<ul style="list-style-type: none">• All resource configurations associated with the service.• All attribute configurations associated with any associated resource.• Any notification templates associated with workflows used to provisioning the service.• Workflows used to provision the service.• External calls on which any associated workflows depend.
Rules	Resources provisioned by the reconciliation rules imported.

Functional Area	Dependencies
Workflow Template	<ul style="list-style-type: none"> • Workflows application definition associated with the workflow. • Notification templates required by the workflow. • External calls on which the workflow depends. • Request Instance Report associated with the workflow.
Workflows	<ul style="list-style-type: none"> • External calls • Workflow application definitions • Notification templates • Request Instance Reports
Configuration Approval Setup	<ul style="list-style-type: none"> • Workflows

Importing a Configuration

To import a configuration, perform steps provided in the online help. HP recommends that you import configurations in this order:

- 1 Import configurations with no dependencies:
 - Rules
 - Connectors
 - External Calls
 - Notification Templates
 - Request Instance Reports
 - Workflow Application Definitions
 - Administrative Roles
- 2 Import configurations that are dependent on external calls¹:
 - Attributes
- 3 Import configurations that are dependent on external calls, workflow application definitions, notification templates and Request Instance Reports:
 - Workflows
- 4 Import configurations that are dependent on workflows¹:
 - Configuration Approval Setup
- 5 Import configurations that are dependent on attributes, rules and connectors:
 - Resources
 -  Exported attributes depend on resources and vice versa. Make sure that attribute mappings have been migrated!
- 6 Import configurations that are dependent on resources¹, attributes¹, notification templates¹, and workflows¹

— Services

- 🚩 Migrate one service at a time, and test each as you go.

¹ Dependency that, if missing, will cause the import to fail.

To import a configuration, perform steps provided in the online help.

- 🚩 Read any warnings that are given after the import. For example, a warning message might indicate missing dependencies that need to be added. See [Understand Dependencies](#) on page 89 for more information.

Secure Migration

In some circumstances, you may want assurance that only configurations from a trusted source can be imported. For example, imagine that you have a development environment, a test environment, and a production environment for your Select Identity solution. It is essential that your production environment import *only* a tested configuration.

One way to assure that is through Select Identity’s **secure object migration** feature. This feature of Select Identity makes use of industry-standard paired-key technologies to assure that your production system will accept a new configuration only from the testing group, and not accidentally from the development lab.

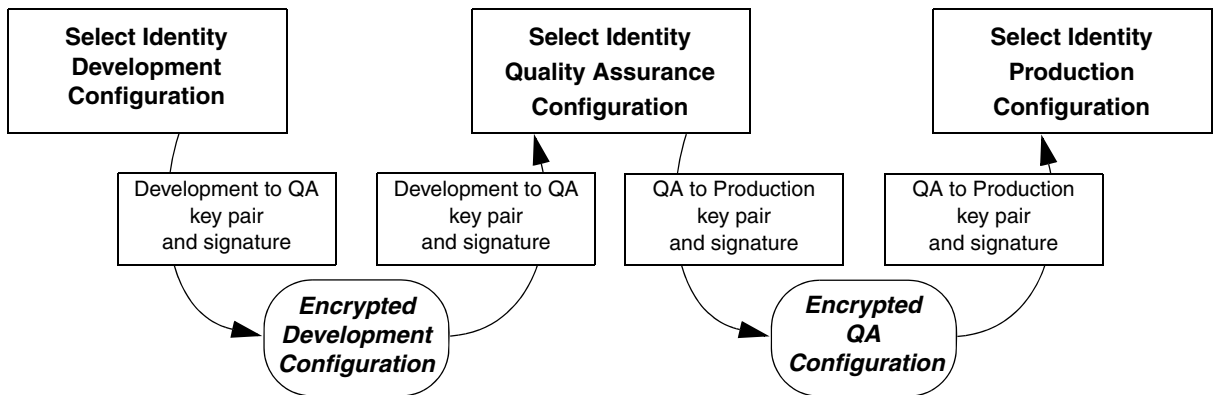
Secure object migration encrypts and signs your configuration upon export. A different Select Identity installation can import this configuration only if it has the correct decryption and signature keys. The encryption keys are created outside of Select Identity, using third-party tools.

At a high level, secure object migration works as follows:

- 1 The source system exports the configuration using its private signature key, and the destination system’s public encryption key.
- 2 The destination system imports the configuration using its private encryption key, and the source system’s public signature key.

As a result, the destination system is guaranteed that the received configuration is legitimate. The earlier example is illustrated in [Figure 9](#), where the production environment is protected against accidental import of a development configuration:

Figure 9 Secure Object Migration



Secure object migration depends upon both the source and destination installations of Select Identity to have the necessary security framework, which you set up during installation.

See the *HP OpenView Select Identity Installation and Configuration Guide* for details about setting up the `keystore` and `truststore` files, which are part of the security framework required for secure migration.

See also the Select Identity online help for detailed procedures on configuring and using secure object migration.

Troubleshooting Configurations

troubleshooting Listed below are common errors and ways to resolve them.

Service can not be found.

Make sure that all dependent configurations were imported when you imported your service. If not, import the required configurations now.

Validate any changes made to the service exported XML file. If errors exist, correct them and import the file.

Password not found.

Review the imported resource to determine if you imported the dependent Password attribute. If not, import the password attribute now.

Resource not found.

Check to make sure that all dependent configurations were imported when you imported your resource. Attributes fields required by the resource must exist in the target application. Import any missing attribute configurations now.

Validate any changes made to the resource exported XML file. If errors exist, correct them and import the file.

Import Failed

Verify that all known dependent configurations have been imported into the target application. If not, import the missing configuration now.

Validate any changes made to any exported XML files. If changes were verify that the changes are correct. Make sure that any name change was made consistently in each instance the name is used. Correct any errors and import the corrected configuration.

Set Up Users

Now that the system is fully configured, you need to populate it with the users whose provisioning will be managed by Select Identity.

Typically, you will use user import to load users from all authoritative sources into Select Identity to build the initial production user population. Follow the methodology in [Data Processing](#) on page 62.



Do a change freeze on all resource systems until the user import process is complete. Afterwards, start any automated reconciliation jobs, so that ongoing authoritative changes will be received by Select Identity.

Conduct Final System Check

Use a checklist to ensure all systems and processes are addressed, such as the following:

- Check that all services are defined.
- Spot check that various users have been imported correctly.
- Spot check that various services have the right number of users assigned as members.
- Check that logging is enabled and working.
- Check that provisioning is working correctly by requesting a new account, testing it, and then requesting it be deleted.
- Check that modifications of imported users also works.
- Check that reconciliation is working by sending through a native production change and then reversing it natively.
- Check that all client support teams are in place and ready.

Congratulations! Your HP OpenView Select Identity identity management solution is in place and functioning.

5 Configuring and Testing User Account Reconciliation

Reconciliation is the process whereby changes to a user account on a resource (for example, an LDAP server) are propagated back into Select Identity so that a user's Select Identity account can be updated with the changes, thus keeping Select Identity and the resource synchronized.

You can configure Select Identity to automatically reconcile account data at pre-determined intervals. These intervals can be monthly, weekly, daily, or hourly, or even every few minutes.

Select Identity can also reconcile multiple-user-ID accounts. For more information, see [Set Up Multiple-User-ID Accounts](#) on page 68.

Select Identity also supports reconciliation with encrypted attributes. For more information, see [Web Service Reconciliation SOAP Request with Encrypted Attributes](#) on page 114, and the *HP OpenView Configuration and Installation Guide*.

Before you start the reconciliation process, you should optimize Select Identity for best performance. To perform reconciliation tasks, determine the method used to reconcile changes. Reconciliation can be executed through either of the following methods:

- Changes to a resource are captured in an SPML file and then uploaded into Select Identity. This is the most common method used. Much of this chapter focuses on this method.
- An agent or utility captures changes and sends the changes to Select Identity through a Web Service interface.
- A two-way connector is used to periodically poll a resource and return any changes directly to Select Identity.

This chapter covers the following:

- [Reconciliation Overview](#)
- [Reconciling with Authoritative and Non-Authoritative Resources](#)
- [Designating Attributes as Sync In or Sync Out](#)
- [Writing Reconciliation Rules](#)
- [Defining Your Reconciliation Policy](#)
- [Scheduling and Managing Reconciliation Jobs](#)[Testing Reconciliation Prior to Production](#)
- [Evaluating Reconciliation Job Results](#)
- [Troubleshooting Reconciliation Jobs](#)

Reconciliation Overview



Before you start the reconciliation process you should optimize Select Identity for best performance. For more information, refer to the *HP OpenView Select Identity Installation and Configuration Guide*.

When resources are created, they can be designated as authoritative resources. These resources generally have the most up-to-date account information and are often used to store the master account records for an identity. For example, an enterprise may have 20 different applications that contain user or account data. However, the human resources application only updates the user's personal data on one resource. This resource would be considered the *authoritative resource*. Once an authoritative resource is designated, you can begin adding or changing user accounts through reconciliation.

Add requests from authoritative resources typically add new accounts in Select Identity. If an account exists and an add request comes in for a user, then the user's attribute values will be updated. As a result, the user's service membership will be re-evaluated to determine which, if any, services are assigned or removed.

Take the example of an add request coming in from reconciliation for a user who was previously disabled, and who, as a result lost all of their service memberships. During reconciliation, this user could be reassigned to the services they had before. During reconciliation with an authoritative resource, any existing rules associated with the resource are checked to determine if service access should be given. Rules are explored later in this chapter.

In most cases, Select Identity's reconciliation capability primarily relies on an SPML data file as its input from a resource. This SPML file should reflect changes to both attribute values and entitlements for a specific period. Typically, the SPML file is automatically uploaded to Select Identity. Once the file is uploaded, Select Identity uses it to check for any user account changes, and all services that rely on the specific resource are updated accordingly. However, if Select Identity determines during reconciliation that changes, such as adding fixed entitlements, might also affect the source resource, then the source resource will be provisioned back. Usually, however, the source resource is not provisioned back.


Service Membership Requirements

When a user account is added or modified through reconciliation, the user's service memberships are evaluated. Based on the changes to the user's attributes, a service membership can be assigned or removed. For a user to be assigned to a service through reconciliation, the user account must have all of the following:

- Access to all the resources contained in the service.
- All required service attribute properties defined by the service.
- A matching context value defined in one of the service's service roles.

All fixed entitlements are defined in the service's service roles, and are related to the user's context value. For example, if a user's context is associated to the third level in a service role hierarchy, then the user must also have all fixed entitlements assigned to the first and second levels of the service role.

- All optional entitlement values for the service based on the user's context value
- Matching attributes field values present must also match all field properties such as length and pattern

- Matching attribute values for any attributes in the service that have a constraint list.
 - A matching attribute value for any attributes in the service included in a constraint list. Constraints may be part of the static attribute definition contained in Select Identity or they may be a part of an external call if a call has been defined for the attribute.
-  These entitlements and attributes are defined as part of implementing your service. See [Chapter 3, Developing and Building Your Solution](#) for more information.

Resource Provisioning During Reconciliation

A user is removed from a service during reconciliation if the user's account no longer meets the requirements of the service. Changes to a user's attributes during reconciliation will cause provisioning in a resource if one of the following conditions is met:

- The attribute being changed is mapped to other resources and the user belongs to the service containing the attribute.
- A reconciliation rule is executed to add a user.
- When adding user to a service pro-actively through rule, fixed attributes are assigned to the user, thus provisioning the user in a new resource.

If a user attribute is mapped to an authoritative resource, or if the user attribute sync-in property is enabled, the user attribute is updated in Select Identity and is synchronized with all other resources to which the attribute is mapped. Synchronization across resources will only occur if the following two conditions are met:

- The user being changed contains attributes with the sync-out property enabled and mapped to a resource.
- The user has a service membership that contains the attribute and resource.

System Configuration Prior to Reconciliation

Before running a reconciliation job, make sure the following activities have been completed:

- All connectors and resources are deployed for all the target applications and resources you want to reconcile.
- Validate that all necessary resource and Select Identity attributes are mapped within Select Identity.
- Verify all Select Identity attributes are mapped to the appropriate resource attributes using the **Attributes** functionality or the **Modify Resource Attributes Mapping** feature in the **Resources** functionality.
- Make sure that at least one service has been created to use the resources for which you want to reconcile data.
- Make sure that at least one workflow template for reconciliation has been assigned to each service. You can use the default reconciliation template, `ReconciliationDefaultProcess`, or create and specify a custom template.
- Be sure that reconciliation events have been configured in each service.
- Define reconciliation policies for each resource.

Reconciling with Authoritative and Non-Authoritative Resources

When a resource is created, it can be designated as an authoritative resource. An authoritative resource generally has the most up-to-date account information and is often used to store the master account records for identities. For example, an enterprise may have twenty different applications that contain user or account data. However, the human resources application only stores the user's personal data on one resource. This human resources system would be considered the authoritative resource. All individuals must exist on this resource before they can be added to other applications or resources in the enterprise. See [About Resource Classes](#) on page 38 on for more information on authoritative and non-authoritative resources.

You must identify and designate an authoritative resource before performing reconciliation in Select Identity. Once an authoritative resource is designated, you can begin adding user accounts through reconciliation.

- ▶ An account on Select Identity cannot be added or updated from any other resource unless it is first added from an authoritative resource. After the account is added from the authoritative resource, then you can begin adding entitlements and other attributes from other resources.

Reconciling with an Authoritative Resource

Add requests from authoritative resources typically add new accounts in Select Identity. (An exception to this is a multi user add request.) If an account exists and an add request comes in for the user, then the user's attribute values are replaced with the values in the new add request. As a result, the user's service membership will be re-evaluated to determine which services are assigned or removed. If an add request for a user comes in from reconciliation and the user was previously disabled on the resource, then the user will become enabled again after the add request has been processed.

During reconciliation with an authoritative source, any existing rules associated with the resource and request event are checked. See [Writing Reconciliation Rules](#) on page 102 and [Appendix A, Creating Select Identity Rules](#) for more information about rules..

- ▶ Select Identity does not support reconciliation from multiple authoritative resources.

Table 6 Resource Action Options - Authoritative Resource

Event	Resource Policy	New User	Request Action
Add	Ignore	N/A	Nothing happens
Add	Revert	N/A	Remove User
Add	Accept	Yes	Add, Add Service
Add	Accept	No	Add / Modify (without rules) Delete / Enable / Disable Services Enable / Disable / Terminate Users (with rules)
Modify	Ignore	N/A	Nothing happens

Event	Resource Policy	New User	Request Action
Modify	Revert	No	Replace modified attribute/ entitlement with Select Identity values
Modify	Accept	No	Add / Modify (without rules) Delete / Enable / Disable Services Enable / Disable / Terminate Users (with rules)
Delete	Ignore	N/A	Nothing happens
Delete	Revert	No	Add users back to resource with Select Identity values
Delete	Accept	No	Terminate

Reconciling with a Non-Authoritative Resource

Non-authoritative resources typically contain attributes and entitlements for user accounts. After adding user accounts from the authoritative resource, process non-authoritative changes to user accounts. Changes may include the addition or removal of entitlements. Resource entitlements are automatically considered authoritative and these are always updated in Select Identity even when coming from a non-authoritative resource.

In some cases, you may need to synchronize other attributes from non-authoritative resources. Select Identity allows you to set a resource attribute's property called *Sync In*, so that any changes to that attribute on the non-authoritative resource are also made in Select Identity. For more information, see [Attribute-Level Reconciliation Policy](#) on page 103.

If, however, an attribute value changes on a non-authoritative resource and the attribute is not designated as *Sync In*, no changes will be made.

When reconciling changes from a non-authoritative source, remember the following:

- Account changes from a non-authoritative resource require that the user account already exists in Select Identity. (An exception to this are multiple-user-ID reconciliations.)
- If the attribute exists and has a value in Select Identity, it needs to be defined as *Sync-In* to enable an override of an existing value.
- When attributes are set to *Sync Out*, changes inside Select Identity are pushed out to the resources. For example, assuming Attribute 1 exists in a Select Identity service, and the service contains Resources A and B, if Attribute 1 on Resource A is set to *Sync-In*, and Attribute 1 in Resource B is set to *Sync-Out*, then a change in Attribute 1 on Resource A will be taken into Select Identity first and then pushed to Resource B.

Table 7 Resource Action Options - Non-Authoritative Resources

Event	Resource Action	New User	Request Action
Add	Ignore	N/A	Nothing happens
Add	Revert	N/A	Delete from Resource
Add	Accept	Yes	Attempt to Add Service

Event	Resource Action	New User	Request Action
Add	Accept	No	Add / Modify (without rules) Delete / Enable / Disable Services Enable / Disable / Terminate Users (with rules)
Modify	Ignore	N/A	Nothing happens
Modify	Revert	No	Undo changes
Modify	Accept	No	Add / Modify / Delete (without rules) Enable / Disable Services Enable / Disable / Terminate Users (with rules)
Delete	Ignore	N/A	Nothing happens
Delete	Revert	No	Add user back in resource
Delete	Accept	No	Removes resource from user (may delete service)

Designating Attributes as Sync In or Sync Out

Reconciliation takes place between Select Identity and one resource — called the **source resource** — at a time. During reconciliation with a given source resource, an attribute’s value for some user on the source resource may be found to differ from the same attribute’s value for that user in Select Identity. At that point, several independent actions can occur:

- 1 Select Identity may or may not update its data to conform with the source resource’s data.
- 2 Select Identity may or may not update the source resource so that its data conforms with Select Identity’s data.
- 3 Select Identity may or may not update other resources who know that user and that attribute — called **destination resources** — to conform with the new value.

What actually happens depends upon how the attribute’s synchronization properties — called Sync-In and Sync-Out — are set:

- **Sync-In:** If the Sync-In property is on (check marked) for the attribute on the source resource, then the new attribute value can be accepted and updated in Select Identity.
- **Sync-Out:** If the Sync-Out property is on (check marked) for the attribute on a destination resource, then Select Identity propagates its value for the attribute to update the resource.

▶ An attribute’s Sync-In and Sync-Out properties are distinct and independent; the effects of each are not linked in any way.

That is, the Sync-Out property for an attribute does not pertain to, and has no effect on, the source resource during reconciliation.

Similarly, the Sync-In property for an attribute does not pertain to, and has no effect on, a destination resource during reconciliation.

You can arrange to have a non-authoritative resource be the effective “authority” for a specific attribute. For example, suppose your authoritative resource is your Human Resources system. However, your non-authoritative Facilities Management system has the most up-to-date information regarding a `location` attribute.

In that case, you should make two changes regarding the `location` attribute:

- 1 For the Facilities Management resource, turn this attribute’s Sync-Out property **off**, unless you want to have Select Identity to be able to update the Facilities Management resource.
- 2 For the Facilities Management resource, turn this attribute’s Sync-In property **on**.
- 3 For the Human Resources resource, turn this attribute’s Sync-In property **off or on**, depending on whether you want to have it also be authoritative or not.

The `location` attribute for all other non-authoritative resources that use it should remain set to Sync-Out. These steps ensure that a change to the value of the `location` attribute, made on the Facilities Management resource, gets propagated to other resources.

▶ The synchronization properties described here only pertain to changes made during reconciliation. Changing an attribute value for a user within Select Identity always updates that person’s resources or services regardless of the synchronization properties for the attribute.

Take care to be methodical when setting synchronization properties. Reconciliation occurs with respect to one resource at a time. Conflicting synchronization properties for an attribute can result in inconsistent data.

For information about the synchronization properties of attributes, refer to [Attribute-Level Reconciliation Policy](#) on page 103

Writing Reconciliation Rules

When adding user accounts through reconciliation, you can control how accounts are added within the enterprise by using a reconciliation rule.

Reconciliation rules are associated with reconciliation events in the resource reconciliation policy. They specify the attributes, entitlements and services that users are eligible for based on particular criteria defined in the rule. Rules make assignments after determining if the request meets the rule criteria.

Rules are created outside of Select Identity and then uploaded to the system. You must create an XML file that adheres to the rules DTD used by the Workflow Template. When you add the rule in Select Identity's **Rules** capability, the XML rule file is uploaded to the Select Identity database. A sample rule and overview of the DTD can be found at `\SampleXML\ReconciliationRules` directory on HP OpenView Select Identity product CD.

If a reconciliation rule is configured for a resource, when Select Identity receives a reconciliation request for a resource, the system applies the reconciliation rule. If the user meets the criteria specified in the rule, then the corresponding actions and/or events associated with the criteria is applied.

Here is a typical scenario in which a rule would be used: A business rule states that all employees hired into the Information Technology (IT) department are added to Active Directory. In order to implement this business rule in Select Identity, a reconciliation rule is created that examines a new employee's `department` attribute value. If the value equals "IT", then the rule provisions the new employee to the Active Directory resource.

In addition, services can be added to a user using an external call to read the rule from a workflow. Using this method, separate requests are spawned for each additional service.

Users will only be added to services stated in the rule if they meet the criteria for the rule.

If the user action for a resource reconciliation policy for an add request is set to `Auto` or `Rule` and a rule *is* defined, users that do not meet the requirements of the rule will still be assigned those services for which they are otherwise qualified.


If the user action for a resource reconciliation policy for an add request is set to `Auto`, users will still be assigned those services for which they are qualified.

Defining Your Reconciliation Policy

Reconciliation policies let you define for each resource a set of actions for each reconciliation activity. If you have a two-way polling connector installed for a resource, you can also configure your reconciliation to poll at pre-determined intervals for changes to account attributes made outside of Select Identity. When Select Identity receives a reconciliation event from a resource, it evaluates the policy of the resource before taking any further action. When writing your XML reconciliation rule files, consider the way policies are evaluated in Select Identity. Each reconciliation request is built based on the evaluation results. One request per user is created. Multiple operations will be set as different targets in the request. Policies are evaluated as shown in [Table 8](#).

Table 8 Evaluation of Policies

Actions	Use
Actions exist at the Attribute Level	Select Identity uses the action with the most coverage. This precedes the resource level action.
No Attribute level Action global policies are configured	Select Identity uses actions defined at the resource level.
No Resource level Action polices are configured	Select Identity applies system defaults.

 If you need to add additional services to a user’s account when changing a user’s context, refer to the Workflow Studio online help. This process lets you build rules and external calls to add new services to an account while modifying the account.

Attribute-Level Reconciliation Policy

To set the reconciliation policy for an attribute, you must perform two tasks:

- Choose a resource action option from the **Modify Attribute** page, This section describes the resource action options available from this page.
- Set the **Resource Action** option to **Accept** in the Modify section of the **User Reconciliation Policy** page for *each resource* that contains the attribute. See the Resource-Level Reconciliation Policy section below for more information.

Procedural details for both tasks are available in the Select Identity online help.

Here are the Resource Action options available for each attribute from the **Modify Attribute** page:

- **Auto** - (Automatic Service Assignment) Apply basic action and evaluate existing/new services for the user based on attribute values.
- **Sync Resource** - Synchronize changes with other resources.
- **Basic** - Modify attribute in Select Identity without any service operations.
- **Rule** - Use actions specified in the XML rule. Apply additional user operations given in the rule if the user qualifies. If you select this option, make sure you have first added a rule from the **Modify Resource>User Reconciliation Policy** page.
- **Rule or Auto** - Attempt to apply a rule first. If the user is not qualified, apply the Auto action as described above.
- **Auto and Rule** - Apply rule-specific action, and then perform automatic service assignment.

Resource-Level Reconciliation Policy

To set the reconciliation policy for a resource, you must modify the resource. From the **Modify Resource** page, click the **User Reconciliation Policy** link to display the User Reconciliation Policy page for the resource you selected. This section explains the resource-level reconciliation policy options available from this page. Procedural details are available in the Select Identity online help.

Reconciliation Filter

The first section on the User Reconciliation Policy Page is **Recon Filter**. Choose one of the available filters to convert non-standard SPML requests (for example converting an add request to a modify request) into a standard SPML request. More information about creating and uploading SPML filters into Select Identity see the HP OpenView Select Identity *Configuration and Installation Guide*.

User Polling

The next section is User Polling. User Polling must be used in conjunction with a connector for the resource that allows two-way polling. Once the Polling Enabled option is selected, you can set the polling interval frequency values to indicate when Select Identity should poll this resource to check for any changes.

Add, Modify, Delete

Below the User Polling section, there are three sections of the policy page corresponding to each type of user request: Add, Modify and Delete. They all contain the same fields, but some of the options for each field are different. You can set a special reconciliation policy for each of these user requests.



Detailed descriptions of each option can be found in the online help.

Report Policy allows you to specify how you want the records in the SPML file upload to be treated when there is no operation that can be performed on them. Options are as follows:

- Brief: Ineligible Op As Fail (default) - Brief format; ineligible records will counted in the reconciliation report as failed requests.
- Detail: Ineligible Op As Fail - Detailed format; ineligible records will counted in the reconciliation report as failed requests.
- Brief: Ineligible Op As No Op - Brief format; ineligible records will be counted in the reconciliation report as no operation occurred.
- Detail: Ineligible Op As No Op - Detailed format; ineligible records will counted in the reconciliation report as no operation occurred.

Audit Enabled allows the reconciliation process to be audited and logged when records are added.

Resource Action determines what Select Identity should do with a change coming from this resource. Options are Accept (default), AcceptNew, Revert, or Ignore.

- Accept means Select Identity will just accept the changes, and apply user actions, if any, plus the user attribute-level policies.
- AcceptNew means Select Identity will allow creation of secondary accounts from this resource.

Make sure you have configured the following attributes correctly for Multiple-User-ID accounts:

- Resource Action
- Secondary User Membership Control
- Identity Propagation
- Propagation Workflow

- **Revert** means that Select Identity rejects the request and undoes the change on the source resource.
- **Ignore** means Select Identity ignores or does not process the change.

See [Table 6](#) on page 98 and [Table 7](#) on page 99 for a complete list of Resource Action options

User Action only applies if Resource Action is set to `Accept`. The default value for User Action is `Auto or Rule`. This means that if there is a rule, Select Identity attempts to apply the rule first to the user request. For an Add User request with no rule, the user is automatically added to Select Identity. If it is a Modify user request with no rule, the changes are applied to the user. In both cases the user is assigned to the services he or she qualifies for, without granting entitlements. Here are the User Action options:

- **Basic** - Create or modify user in SI without any service operations
- **Auto** - (Automatic Service Assignment) Apply basic action and evaluate existing/new services for the user based on attribute values (Automatic service assignment)
- **Rule** - Use actions specified in the XML rule. Apply additional user operations given in the rule if the user qualifies
- **Rule or Auto** - Attempt to apply a rule first. If the user is not qualified, apply the Auto action as described above.
- **SyncResource** (Modify event type only) - Synchronize changes with other resources
- **Rule and Auto** - (Modify event type only) - Apply rule specific action and then perform automatic service assignment.
- **Terminate User** - (Delete only) - Terminate user and associated secondary users from **Select Identity** and all resources.
- **Terminate if Last, x Secondary** - (Delete only) - If the deleted user is a normal or *secondary* user, and if this is the last authoritative resource the user is on, then the user is terminated from Select Identity and all of its resources. If this is not the last authoritative resource the user is on, then the source resource-specific attributes of the user are removed, and the user is deleted from all services containing this source resource.

If the deleted user is a *primary* user, and if this is the last authoritative resource the user is on, then the user and associated secondary users are terminated from Select Identity and all of its resources. The operations on secondary users may provision back to the source resource. If this is not the last authoritative resource the primary user is on, the source resource-specific attributes of the user are removed, and the user is deleted from all services containing this source resource. All secondary users that are associated with this source resource are found and terminated from Select Identity and all of its resources. The operations on secondary users may provision back to the source resource.

- **Terminate if Last** - (Delete only) - Similar to *Terminate if Last, x Secondary*, except that secondary users are not deleted if the user is not terminated.

Reconciliation Workflow - Select the approval workflow you wish to use for reconciling this resource. The workflow must have a reconciliation event associated with it.

Rule Name - Select the rule you wish to apply.

Serial Process - Selecting `Yes` means that all records for the same user will be processed sequentially. This should be set uniformly for add, modify, or delete. Serial processing is more accurate but slower. The serial process user action prevents a modify request from occurring before an add event.

Secondary User Membership Control - When adding service(s) through rules to a secondary user, the following policy options can be configured to control assignment of services. Please note that Secondary User Membership control is applicable only with Rule, Rule Or Auto and Rule And Auto options.

- **No Rule Services** - The rule will not be applied to secondary users.
- **Rule Services from Defined List** - The common set of services between what is specified in the rule and this list will be assigned to the user.
- **Rule Services On Source Resource** - Only attempts to assign the set of services that are specified in the rule and also contains the source resource as one of its resource
- **Any Rule Service** - No restriction on or filtering of the services in the rule.

Identity Propagation - Changes made to primary user attributes can be propagated to corresponding secondary accounts provided the Identity propagation flag is enabled. Attribute level propagation property has to be enabled for this. Profile attributes are by default propagated to secondary accounts. Select **Enable for Primary User** if you want modifications to flow to secondary users; otherwise select **Disable**.

Propagation Workflow - If the Identity Propagation option is enabled, then choose the workflow to use to propagate changes to secondary users. Changes made to primary accounts through reconciliation can be propagated, or flow, to secondary accounts. The distribution of changes from the primary account to the associated secondary accounts can be controlled by setting the propagation policy at the attribute and resource levels.

Multiple-User-ID Accounts and User Reconciliation

Multiple accounts for a person on one resource can be consolidated and managed in Select Identity using reconciliation or user import.

- ▶ The section titled [Importing and Updating Multiple-User-ID Information](#) on page 70 includes information about creating multiple-user-ID accounts via user import.

As a result of consolidation, multiple accounts for a person on one resource are represented in Select Identity using one primary and one or more secondary accounts.. Consolidation can happen when users are added to Select Identity using SPML `addRequests`.

During reconciliation, secondary accounts can be created using `primaryAcctKey` and `primaryAcctValue` tags. The primary account must exist prior to the addition of the secondary account. The primary account can already exist in Select Identity, or be included in the import file before the secondary account.



To create a secondary account during reconciliation, explicitly specify the `primaryAcctValue` in the operational attributes of secondary user in the input SPML. This explicitly creates a secondary account irrespective of whether it is the first account for the user on the resource or not.

- ▶ During user import, secondary accounts can be created *implicitly*, using `keyFields` information in the input SPML. However, this option is not available during reconciliation.

For an applicable SPML example, see [Import from a Non-Authoritative Resource](#) on page 73.

Authoritative and Non-Authoritative Resources and Multiple-User-ID Accounts

The major difference between reconciliation with authoritative or non-authoritative resources is the action when a primary account is not found. If the resource is authoritative, Select Identity adds the new user from the resource. If the resource is non-authoritative and a primary user is not found, the request is rejected.

-  Unlike user import, reconciliation does not support account consolidation with multiple authoritative resources.
-  When importing users from multiple authoritative resources, do not use reconciliation to create secondary accounts. Use user import and implicit creation of secondary accounts, instead. This will avoid creating inappropriate secondary accounts.

Workflow and Multiple-User-ID Accounts

Primary and secondary resource accounts are subject to the same workflow approvals as standard user accounts. However, if a request for a primary account is rejected, all associated secondary accounts are rejected.

Accounts with the same context are grouped into one request for approval purposes. Accounts being added or associated to different services with different context attributes require separate workflow requests.

Scheduling and Managing Reconciliation Jobs

To make sure that all user account changes are frequently reconciled in Select Identity, you should schedule an automatic reconciliation job. See the online help for details on how to schedule and execute an automatic reconciliation job.

Preparations

If you are planning to use reconciliation through the Select Identity interface, the following tasks must be completed first:

- Add all resources that are part of the reconciliation.
- Designate an authoritative resource. See [About Resource Classes](#) on page 38 for details.
- Identify and map the attributes to be updated in Select Identity accounts by mapping specific attributes to all resource attributes included in your identity management solution. Do this by using the **Modify Resource Attributes Mapping** feature in the **Resources** function or by using the **Attributes** functions. See [step 2](#) on page 41 for details.
- Identify any attributes that need to be set to Sync-In or Sync-Out. See [Attribute-Level Reconciliation Policy](#) on page 103 for details. Determine which attributes must be updated in Select Identity, and set these attributes to Sync-In.
- Identify the user account unique ID and attributes from a resource participating in reconciliation.
- Specify the attributes to be synchronized across various resources.

- Create and upload any reconciliation rules associated with the addition of accounts from authoritative resources.
- Define reconciliation policies for each resource that set the parameters used during reconciliation. See [step 3](#) on page 41 for details.

Using an Agent or Web Service Interface

Reconciliation can also be done through an agent or Web Service Interface. The format of the requests in Web Services is different from the format accepted using the **Reconciliation** pages. If you are planning to use reconciliation through an agent or Web Service interface, you will need to perform the following tasks:

- Create resources and attributes as described in [Defining Resources and Attributes](#) on page 37.
- Create batch requests or single requests compatible with Select Identity's Web Services Interface. See the `SampleXML\Reconciliation\Web Service` folder on the Select Identity product CD for examples of Web Services requests.
 - ▶ See the Web Services section of the *HP OpenView Select Identity Web Services Developer Guide* for more details.
- Invoke the Web Service from an agent, utility, or another web service.

Task Status

Each time a reconciliation request is scheduled, Select Identity creates a corresponding task. Tasks are generated from multiple sources:

- A one-time scheduled task
- A scheduled task repeated at specified intervals
- A Web Service reconciliation request
- A resource change polling request

Tasks are listed in the **Reconciliation Task List** and can be viewed at any time. If a task is marked complete, the Select Identity job has completed successfully. View the status of the request on the Request Status List page. The system also provides additional information in detailed reports you can generate for each task.

The Select Identity online help can show you how to use the **Reconciliation** item in the **Tools** menu to check your task status.

Testing Reconciliation Prior to Production

Before going to production, your reconciliation policy should be tested, so that you are sure that all changes to authoritative and non-authoritative resources are corrected reconciled by Select Identity. Assuming that you are inputting user data from SPML files, here are the steps you should take to perform testing:

- 1 Process the SPML Data Files (see the following section)
- 2 Upload the user data file for the authoritative resource and run the reconciliation

- 3 View, validate and troubleshoot results
- 4 Upload the user data file for the non-authoritative resource and run the reconciliation
- 5 View, validate and troubleshoot results

If you are using automatic polling with a two-way connector or a web services agent, simply run your reconciliation job and view, validate and troubleshoot results.

Processing the SPML Data File

The reconciliation function uses an SPML data file type to make account changes. This file should reflect changes from a specified resource, including entitlement and attribute changes. The file is then uploaded to Select Identity.

All SPML data files for automated jobs are stored in the reconciliation root directory as specified in the `TruAccess.properties` file. See the *HP OpenView Select Identity Installation and Configuration Guide* for information about this properties file and [Application Server Properties](#) on page 116 for reconciliation specific settings.

Understanding Dependencies

SPML files must be imported to handle reconciliation events. However, the successful import of a file does not mean that the functionality associated will work as expected. All dependencies must be met and any other supporting files on which the function depends must be present in order for the SPML functionality to work as planned. Dependencies are detailed in [Table 9](#).

Table 9 SPML File Dependencies:

Area	Dependencies
KeyFields	KeyFields must be defined. If KeyFields are not defined then the default value is UserName.
Attributes	Attributes must exist and be properly mapped in Select Identity. Attribute names in file are resource attribute names.
Resource	Resources used in reconciliation should be defined in Select Identity and appropriate policy should be configured for each of the resources.
Service	Services must exist and be accurately defined.
Rules	Rules used during reconciliation must exist. See Writing Reconciliation Rules on page 102.
User Account	User accounts must be set up on Select Identity unless the accounts are being added through reconciliation.
Connector	Connector must exist and be accurately defined.
Roles	The system administrator should have the necessary permissions to submit a reconciliation job. The system administrator also needs to have access to all services and contexts

Resource Names

Resources may follow any naming convention. If, however, resource names contain underscores then you must change the `com.hp.ovsi.spml.resourcename.separator` setting in `TruAccess.properties` to another character, such as `com.hp.ovsi.spml.resourcename.separator = +` to define the separator string in the reconciliation SPML file name between the resource name and the date/time portion. For example the file name will be `Resource_Name+yyy_mm_dd_hh_mm`. This allows reconciliation job files to be automatically uploaded when an underscore “_” is included in the names.

- ▶ One-time jobs submitted through the Select Identity user interface can use any naming convention.

Creating an SPML File Containing Users and Attributes

Many resources today have a utility or mechanism for exporting user data to an Extensible Markup Language (XML) or Service Provision Markup Language (SPML) format. To create the SPML format needed for reconciliation, perform one of the following:

- Export your data in the resource to LDIF format and use a parser to convert the data to SPML.
- Export your data in the resource to XML or DSML format. Convert it to SPML using an XML parser and XSLT style sheet.
- Use a third-party mapping tool to convert your data to SPML format.
- Use the SPML Generator tool provided in your Select Identity product CD. Learn more about the SPML Generator Utility in [Appendix B, The SPML Generator Utility](#).
- Programmatically build the file by reading through your resource and writing out changed records for each user.
 - ▶ Some connectors support a change detection utility called reverse synchronization. This utility enables the resource to send changes to the Select Identity server. See the *HP OpenView Select Identity Connector Developer Guide* for a complete description of reverse synchronization.

To see an example of SPML conversion files, view the sample files located in the `\SampleXML\Reconciliation` directory on the HP OpenView Select Identity product CD.

- ▶ When specifying attributes in the SPML file, be sure to use the attribute name in the mapped resource. This name often differs from the Select Identity attribute name.

Writing SPML

When creating the input file containing the user attributes, specify a unique identifier attribute associated with each user. The identifier is specified in the `<operationalAttributes xmlns=>` section of the SPML file and is designated as a value in the `keyFields` attribute. Select Identity’s default attribute for identifying accounts is `UserName`. The following is a sample of this section of the SPML file:

```
<operationalAttributes xmlns="">
  <attr name="urn:hp:selectidentity#keyFields">
    <value>UserName</value>
  </attr>
```

```
</operationalAttributes>
```



The former attribute name `urn:trulogica:concero:2.0` is backwards compatible to the new name `urn:hp:selectidentity`

This is the attribute name and value that Select Identity uses to determine if the account exists. In addition to specifying the user ID operational attribute in the header of the file, you must specify two other operational attribute values for each add user request:

```
<addRequest requestID="1">
  <operationalAttributes xmlns="">
    <attr name="urn:hp:selectidentity#taUserName">
      <value>avaughan</value>
    </attr>
    <attr name="urn:hp:selectidentity#taResourceKey">
      <value>AQ4100</value>
    </attr>
  </operationalAttributes>
```

The `taUserName` field value defines the unique value used to identify each account in Select Identity. When a new user is created in OVSI, the value specified in the `taUserName` attribute will be used `UserName` in OVSI. The `taResourceKey` defines the corresponding key used to identify the account on the resource from which the user originates. For resources where there is generation, use another value for Key Fields. For more information refer to [Creating an SPML File Containing Users and Attributes](#) on page 64.

If the `UserName` attribute is set up with a value generation function, and a reconciliation request is made from an authoritative resource, the `taUserName` should not be specified in the SPML file. Select Identity will invoke the value generation function to create the `UserName`. The user will be provisioned in Select Identity with this generated `UserName`.

Sensitive fields, such as one-way or two-way attributes, or other fields designated as confidential will not be included in reports with cleartext. They will be encrypted in the generated report, and masked with asterisks in the HTML report.

SPML Tips

- Encoded fields in SPML can be parsed properly according to their security settings.
- Each account to be added begins and ends with `<addRequest></addRequest>`. The operational attributes and values listed for each add request are required by Select Identity. An account cannot be added without these attributes and values.
- The keywords required by reconciliation processing must present in operational attributes.
- The administrator user name/password, source resource, resource business key fields, and reverse sync flag are required for all operations in the SOAP section for Web Service SPML. The reverse sync flag should always be set to true for Web Service reconciliation.
- `taUserName` is required for add operation from an authoritative resource unless `UserName` is associated with an `UserID` generation function
- Identifier (not an operational attribute) is required for Modify/Delete operations in sub request section.
- If you need to have multiple values for an attribute within Select Identity, use the following syntax:

```
<attr name="name">
```

```

    <value>value1</value>
    <value>value2</value>
  </attr>

```

Where name is the resource attribute name mapped to a Select Identity attribute.

SPML Examples

SPML File without taUserName

Following is a sample SPML file without the taUserName (using user name generation):

```

<batchRequest xmlns:countries="countries.uri" xmlns:cities="cities.uri"
xmlns:dsml="urn:oasis:names:tc:DSML:2:0:core"
xmlns:spml="urn:oasis:names:tc:SPML:1:0"
xmlns="urn:oasis:names:tc:SPML:1:0" requestID="1085774668899">
  <operationalAttributes xmlns="">
    <attr name="urn:hp:selectidentity#keyFields">
      <value>Email</value>
    </attr>
  </operationalAttributes>
<addRequest requestID="1">
  <operationalAttributes xmlns="">
    <attr name="urn:hp:selectidentity#taResourceKey">
      <value>ResAD051801</value></attr>
  </operationalAttributes>
  <attributes xmlns="">
    <attr name="State">
      <value>TX</value></attr>
    <attr name="LastName">
      <value>Smith</value></attr>
    <attr name="Email">
      <value>john.smith@abc.com</value></attr>
    <attr name="FirstName">
      <value>John</value></attr>
  </attributes>
</addRequest>
</batchRequest>

```

SPML File Identifying an Account with Two Fields

It is possible to identify an account using two different fields. In the example below, LastName and FirstName are used to search for a unique account by specifying them in the keyFields section of the operational attributes.



When using multiple fields, there should not be multiple occurrences in Select Identity. The fields combined must ensure a unique occurrence when searching for a user in Select Identity.

Use the most distinct key as the first value in the file. In the following example, LastName is specified before FirstName since it is the more unique of the two fields:

```

<batchRequest xmlns:countries="countries.uri" xmlns:cities="cities.uri"
xmlns:dsml="urn:oasis:names:tc:DSML:2:0:core"
xmlns:spml="urn:oasis:names:tc:SPML:1:0"
xmlns="urn:oasis:names:tc:SPML:1:0" requestID="1085774668899">
  <operationalAttributes xmlns="">
    <attr name="urn:hp:selectidentity#keyFields">

```



```

        <value>LastName</value>
        <value>FirstName</value></attr>
    </operationalAttributes>
<addRequest requestID="1">
    <operationalAttributes xmlns="">
        <attr name="urn:hp:selectidentity#taUserName">
            <value>chU54500</value></attr>
        <attr name="urn:hp:selectidentity#taResourceKey">
            <value>ch54500</value></attr>
    </operationalAttributes>
    <attributes xmlns="">
        <attr name="Employee ID">
            <value>HP</value></attr>
        <attr name="LastName">
            <value>roberts</value></attr>
        <attr name="Email">
            <value>bob.roberts@yourcom.com</value></attr>
        <attr name="FirstName">
            <value>Bob</value></attr>
        <attr name="State">
            <value>TX</value></attr>
    </attributes>
</addRequest>
</batchRequest>

```

Single Operation SOAP Request

```

<!--
Single SPML Add request to add users from a recon auth resource through webservice.
-->
<soap:Envelope xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/">
  <soap:Body>
    <addRequest requestID="12345" execution="urn:oasis:names:tc:SPML:1:0#asynchronous">
      <operationalAttributes>
        <attr name="urn:oasis:names:tc:SPML:1:0#UserIDAndOrDomainName">
          <value>sis</value>
        </attr>
        <attr name="urn:hp:selectidentity#password"><value>abc123</value></attr>
        <attr name="urn:hp:selectidentity#resourceId"><value>Auth_LDAP_1</value></attr>
        <attr name="urn:hp:selectidentity#keyFields"><value>UserName</value></attr>
        <attr name="urn:hp:selectidentity#reverseSync"><value>true</value></attr>
        <attr name="urn:hp:selectidentity#taUserName"><value>webrecon1</value></attr>
        <attr name="urn:hp:selectidentity#taResourceKey"><value>webrecon1</value></attr>
      </operationalAttributes>
      <attributes>
        <attr name="UserName"><value>webrecon1</value></attr>
        <attr name="Email"><value>QA1@somecom.com</value></attr>
        <attr name="State"><value>TX</value></attr>
        <attr name="urn:hp:selectidentity#groups">
          <value>HR Managers</value>
          <value>PD Managers</value>
        </attr>
      </attributes>
    </addRequest>
  </soap:Body>
</soap:Envelope>

```

Batch Operation SOAP Request

```

<!--
SPML batch request to add multiple users from a recon auth resource through webservice

```

```

-->
<soap:Envelope xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/">
  <soap:Body>
    <batchRequest xmlns:countries="countries.uri" xmlns:cities="cities.uri"
      xmlns:dsm1="urn:oasis:names:tc:DSML:2:0:core" xmlns:spml="urn:oasis:names:tc:SPML:1:0"
      xmlns="urn:oasis:names:tc:SPML:1:0" requestID="1085774668899">
      <operationalAttributes xmlns="">
        <attr name="urn:oasis:names:tc:SPML:1:0#UserIDAndOrDomainName">
          <value>sisas</value>
        </attr>
        <attr name="urn:hp:selectidentity#password"><value>abc123</value></attr>
        <attr name="urn:hp:selectidentity#resourceId"><value>Auth_LDAP_1</value></attr>
        <attr name="urn:hp:selectidentity#keyFields"><value>UserName</value></attr>
        <attr name="urn:hp:selectidentity#reverseSync"><value>true</value></attr>
      </operationalAttributes>
      <addRequest requestID="2" execution="urn:oasis:names:tc:SPML:1:0#asynchronous">
        <operationalAttributes>
          <attr name="urn:hp:selectidentity#taUserName"><value>webrecon2</value></attr>
          <attr name="urn:hp:selectidentity#taResourceKey"><value>webrecon2</value></attr>
        </operationalAttributes>
        <attributes>
          <attr name="UserName"><value>webrecon2</value></attr>
          <attr name="Email"><value>QA2@somecome.com</value></attr>
          <attr name="State"><value>TX</value></attr>
          <attr name="urn:hp:selectidentity#groups">
            <value>HR Managers</value>
            <value>PD Managers</value>
          </attr>
        </attributes>
      </addRequest>
      <addRequest requestID="3" execution="urn:oasis:names:tc:SPML:1:0#asynchronous">
        <operationalAttributes>
          <attr name="urn:hp:selectidentity#taUserName"><value>webrecon3</value></attr>
          <attr name="urn:hp:selectidentity#taResourceKey"><value>webrecon3</value></attr>
        </operationalAttributes>
        <attributes>
          <attr name="UserName"><value>webrecon3</value></attr>
          <attr name="Email"><value>QA3@somecom.com</value></attr>
          <attr name="State"><value>TX</value></attr>
          <attr name="urn:hp:selectidentity#groups">
            <value>HR Managers</value>
            <value>PD Managers</value>
          </attr>
        </attributes>
      </addRequest>
    </batchRequest>
  </soap:Body>
</soap:Envelope>

```

Web Service Reconciliation SOAP Request with Encrypted Attributes



Please refer to the *HP OpenView Select Identity Installation and Configuration Guide* for more information on generating encrypted values.

```

<!-- WS recon add user sample with encrypted attributes -->
<soap:Envelope xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/">
  <soap:Body>
    <batchRequest xmlns:countries="countries.uri" xmlns:cities="cities.uri"
      xmlns:dsm1="urn:oasis:names:tc:DSML:2:0:core" xmlns:spml="urn:oasis:names:tc:SPML:1:0"
      xmlns="urn:oasis:names:tc:SPML:1:0" requestID="1085774668899">
      <operationalAttributes xmlns="">
        <attr name="urn:oasis:names:tc:SPML:1:0#UserIDAndOrDomainName">
          <value>sisas</value>
        </attr>

```

```

    <attr name="urn:hp:selectidentity#password">
      <value>ENC:1:O+Zzu7gXT+muEFDE04lCGLs07dZZifGSE6w/Walp8SVtykzzSzL0FQESaIAMm8pNouL
        SildIa8Q9z6N1gYwjwuF7LSHiWlTl091csT/66CrwYwTIxvgrpfhDJTidDJuYjLCKTDBZy5Hyj
        pNdK/JLYnamEMFnkSoVxkiu/1Vx/t1c=}</value>
    </attr>
    <attr name="urn:hp:selectidentity#resourceId"><value>SensitiveAttrRes</value></attr>
    <attr name="urn:hp:selectidentity#keyFields"><value>UserName</value></attr>
    <attr name="urn:hp:selectidentity#reverseSync"><value>>true</value></attr>
  </operationalAttributes>

<addRequest requestID="1">
  <operationalAttributes xmlns="">
    <attr name="urn:hp:selectidentity#taUserName"><value>recon_sauser1</value></attr>
    <attr name="urn:hp:selectidentity#taResourceKey"><value>recon_sauser1</value></attr>
  </operationalAttributes>

  <attributes xmlns="">
<!-- password (abc123) encrypted using OVSI sample encryption script -->
    <attr name="Password">
      <value>{ENC:1:OG06wm4aNQA9tuUxSe1wd3j8AZhcc2+BXJBNO0eV58hc1cZthSY4wkqx8C1ThEx3NP80D
        n+ucigx+wkvC9a/wm03icc6qA0SzMsdQRSe9RO+BpJf198QoEAXUXkzZF6weB+VKqfxkYcromK9
        PPmlHvfXNfrCThr8v8dG8+kYiJA=}</value></attr>
    <attr name="mailAlternateAddress"><value>recon_sauser1@hp.com</value></attr>
    <attr name="FirstName"><value>recon_sauser1</value></attr>
    <attr name="LastName"><value>recon_sauser1</value></attr>
    <attr name="State"><value>TX</value></attr>
    <attr name="mailHost"><value>txp01</value></attr>
    <attr name="UserName"><value>recon_sauser1</value></attr>
    <attr name="Employee ID"><value>HP</value></attr>
    <attr name="Address 1">
      <value>Address 1</value>
      <value>Address 2</value>
      <value>Address 3</value>
    </attr>
  </attributes>
</addRequest>
</batchRequest>
</soap:Body>
</soap:Envelope>

```

Specifying Attributes in SPML

When specifying attributes in the SPML file, be sure to use the mapped resource attribute's name. This may differ from the Select Identity attribute name. Attributes uploaded to Select Identity must be mapped to a resource. For information related to attribute mapping, see [Adding New Select Identity Attributes](#) on page 37.

Creating an SPML File Containing Entitlements

After building the SPML file containing your list of users and associated attributes, you may need to add attributes and entitlements from other resources to your users. Users may have entitlements from multiple resources. To upload these entitlements, a separate SPML file containing the entitlements must be created for each resource.



Entitlement additions/changes from a non-authoritative resource can only be added if the entitlements were added/changed on the resource. You do not need to create a reconciliation file with entitlements when a user is added to an authoritative resource.

For each resource file created, determine the unique identifier on the resource that links the entitlement to the designated user. This unique identifier is specified in the SPML file as the `taResourceKey` field. If the `keyField` appears in the data section, use the `keyField` value;

otherwise use the value of the identifier. Specify the `userId` or user name so that you can associate the entitlements to the correct Select Identity account. This is designated in the identifier tag as follows:

```
<identifier xmlns=""
type="urn:oasis:names:tc:SPML:1:0#UserIDAndOrDomainName">
  <id>AEE200</id>
</identifier>
```

When specifying the entitlement, the identifier type `UserIDAndOrDomainName` is used to specify the username or account in Select Identity associated with the entitlement. In the example above, the entitlement is associated with an account called `AEE200` in Select Identity.

The operational attributes `keyFields`, and `taResourceKey` are required for assigning entitlements. These are specified in the file that you created to add users to the system. The attribute `keyFields` is only listed once at the beginning of each file. The attribute `taResourceKey` is listed for each user account.

Here is an example file for adding entitlement to an existing user.

```
<?xml version="1.0" encoding="ISO-8859-1"?>
  <batchRequest xmlns:countries="countries.uri" xmlns:cities="cities.uri"
xmlns:dsm1="urn:oasis:names:tc:DSML:2:0:core" xmlns:spml="urn:oasis:names:tc:SPML:1:0"
xmlns="urn:oasis:names:tc:SPML:1:0" requestID="1485774668899">
  <operationalAttributes xmlns="">
    <attr name="urn:hp:selectidentity#keyFields"><value>UserName</value></attr>
  </operationalAttributes>
  <modifyRequest requestID="1">
    <operationalAttributes xmlns="">
      <attr name="urn:hp:selectidentity#taResourceKey">
        <value>rKey_jmercado</value>
      </attr>
    </operationalAttributes>
    <identifier xmlns="" type="urn:oasis:names:tc:SPML:1:0#UserIDAndOrDomainName">
      <id>jmercado</id>
    </identifier>
    <modifications xmlns="">
      <modification name="urn:hp:selectidentity#groups" operation="add">
        <value>RetirementAcct</value>
      </modification>
    </modifications>
  </modifyRequest>
</batchRequest>
```



For other SPML samples, refer to the Select Identity product CD in the `\SampleXML\Reconciliation` directory.

Application Server Properties

Reconciliation relies on settings defined on the web application server. You can configure the necessary parameters in the `TruAccess.properties` file and create relative directories on the application server host.

The `TruAccess.properties` file is described in detail in the *HP OpenView Select Identity Installation and Configuration Guide*. The following is a sample section of the reconciliation-related property entries in the file:

```
#The attributes for reconciliation
truaccess.recon.rootdir=c:/temp/reconroot
truaccess.recon.stagingdir=c:/temp/reconstaging
truaccess.recon.backupdir=c:/temp/reconbackup
truaccess.recon.filename.timeformat=yyyy_MM_dd_H_mm
```

```
#The attributes for batch processing
truaccess.batch.inprogresstimeout=18000000
truaccess.fixedtemplate.reconciliation=ReconciliationDefaultProcess
si.recon.webservice.report.generate=2
```

The following table explains properties supporting reconciliation in Select Identity

Table 10 Properties Used for Reconciliation.

Property	Req'd	Default	Description
truaccess.recon.rootdir	Yes	None	The root directory for reconciliation data files. If you add a sub directory to this directory, you must add the same to the truaccess.recon.staging directory
truaccess.recon.stagingdir	Yes	None	The working directory for reconciliation.
truaccess.recon.backupdir	Yes	None	The backup directory for completed automated job data files.
truaccess.recon.filename.timeformat	Yes	yyyy_MM_dd_H_mm	The format of the time section within the reconciliation filename.
truaccess.recon.task.check.threshold	Yes	Default = 120 (60 minutes) (Setting this limit to less than 20 is not recommended.)	Controls the number of jobs that stay in "Prepare Data" state. If this limit is exceeded, a new job will stay in "Scheduled" state until one "Prepare Data" job moves on to the next state, or the blocking period expires. The default blocking limit is 120, which equates to about 60 minutes, as the threshold check occurs every 30 seconds.
truaccess.batch.inprogresstimeout	Yes	18000000 seconds	The time-out for when an in-progress batch can be processed again.
truaccess.batch.reportdir	Yes	None	The directory to store XML reports for reconciliation, user import, and service assignment. Apart from generating report files in this directory, User Import, Bulk Add, and Service Assignment reports are also e-mailed to the admin who created the job. For reconciliation, a summary e-mail is sent to the admin.
com.hp.ovsi.connector.changeLogField	No	c:\\Temp\\recon_modify.xml	Directory where the log file is stored.
com.hp.ovsi.default.workflowtemplate.service.change.recon	Yes	SI\\Provisioning\\Only	Workflow template used to make changes to services

Property	Req'd	Default	Description
si.recon.webservice. report.generate	No	2	Determines whether to generate and send a report each time Web Service reconciliation completes 0=Never 1= Only at the initial request when no request is processed 2=Always
si.recon.server.num	No	3	Number of servers in the cluster. This information is used to schedule number of concurrent authoritative and non-authoritative tasks. It is strongly recommended that this property is set based on the actual number of servers in the cluster.
truaccess.recon.task. simul.auth	No	twice the si.recon. server.num default	The value for concurrent authoritative tasks is calculated as (si.recon.server.num * truaccess.recon.task.simul.auth)
truaccess.recon.task. simul.nonauth	No	twice the si.recon. server.num default	The value for concurrent non-authoritative tasks is calculated as (si.recon.server.num * truaccess.recon.task.simul.nonauth)

Evaluating Reconciliation Job Results

To effectively evaluate your reconciliation job results, you must understand how Select Identity evaluates reconciliation policies.

After each reconciliation job completes, an e-mail is sent with the report details such as whether the job was successful, total user accounts processed, how many user accounts passed, and how many failed.

You can generate a final report to see the detailed results if necessary, using the **Reconciliation** item of the tools menu. See the online help for details on how to generate this report.

Using this report, and armed with an understanding of how Select Identity evaluates reconciliation policies, you can make any needed corrections to your SPML file and resubmit the file with only those accounts that failed.

Use the expected policy responses listed below to evaluate the whether your reconciliation job succeeded.

Priority of Policies

Policies are evaluated and acted upon according to the priorities listed below:

- **Attribute level policy** actions are evaluated and applied first.
- **Global level policy** actions are evaluated and applied when no attribute level policies exist.
- **Resource level actions** are applied when no policies exist.

Expected Policy Responses for an Authoritative Resource

Expect the following results for each action when the default is set to `Rule` and the `Resource Action` is set to `Accept`:

- **Action = add**
 - If user account does not exist in Select Identity, and it is listed in the data file, the account is created with all specified attributes and values.
 - If an account already exists in Select Identity but is disabled, the workflow process is started to enable the account. The account attributes are then modified based on the authoritative resource.
 - If there are rules enabled for this action, Select Identity checks the rule to see if new service actions should be performed on the user. Each resource can only have one rule for reconciliation. This action can only take place for new accounts. Only one resource can be authoritative.
 - If a rule is not assigned, user accounts are evaluated against all services that contain the source resource (resource where the change came from). Based on service qualification rules, new services are assigned.
 - Provisioning occurs for resources other than the source resource managed through the new service additions. The workflow requests should be encapsulated in a single request per user.

- Select Identity attributes used for storage are changed first, such as key fields mentioned in [Creating an SPML File Containing Users and Attributes](#) on [page 110](#). service mapping and additional attributes (for example, fixed attributes) are saved based on the provisioning result and post provisioning policy.

- Action = **modify**

Expect the following results when ResourceAction is set to Accept and no policies can be located from change attributes causing reconciliation to default to Rule:

- If a user account does not exist in Select Identity, the account is skipped and it is listed as an error in the job results report.
- If there is no attribute or value change for an account, the action is rejected.
- A modify request is submitted for each user. Provisioning will determine which resources need to be synchronized.
- When the workflow returns to the reconciliation post provisioning, the attribute values in Select Identity are updated based on the provisioning result and post provisioning policy.
- The user profile is checked against service mapping (no fixed attributes will be assigned to the user), to see if any new service assignments are gained or lost. The services to be checked should meet these conditions:
 - The service uses this resource or the user is already assigned to the service.
 - The user has access to all the resources required by the service.
 - The service has attributes that are changed.

If the user can be assigned to a new service, the account is added to or modified in the service.

If the modification makes the user ineligible for a service, the user will be removed from the service and the appropriate entitlements are deleted. The Authoritative Resource Key attribute will never be deleted. Non-Authoritative Resource Keys may be deleted if the user is removed from the last service on the resource, depending on the resource “Delete User” setting.

- Action = **delete**

Expect the following results when ResourceAction is set to Accept:

- If user is not on Select Identity, the action is skipped and reported.
- The assigned workflow is started to terminate the user. Depending on the termination policy, the user may just be disabled for a period of time.
- When the workflow returns to the reconciliation post provisioning, the attributes and attribute values on Select Identity are updated based on the provisioning result and post provisioning policy.

Expected Policy Responses for a Non-Authoritative Results

- Action = **add**

Expect the following results for each action when the default is set to Rule and the ResourceAction is set to Accept or Accept New.

- If the user account does not exist on Select Identity, the action is skipped and reported.

- The `Resource_KEY` attribute is added.
- If Modify Attributes has the Sync-in property set to true then Select Identity is updated and appropriate resources are synchronized.
- An add request is submitted for each user. Provisioning will determine which resources need to be synchronized.
- When the workflow returns to the reconciliation post provisioning, the attribute values in Select Identity are updated based on the provisioning result and post provisioning policy.
- The user profile is checked against service mapping (no fixed attributes will be assigned to the user), to see if any new service assignments are gained or lost. The services to be checked should meet these conditions:
 - The service uses this resource or the user is already assigned to the service.
 - The user has access to all the resources required by the service.
 - The service has attributes that are changed.
- Action = **modify**
 - If the user account does not exist on Select Identity, the action is skipped and reported.
 - The action will be rejected if one of the following occurs:
 - there are no entitlement changes
 - sync-in property is not enabled for the attribute
 - The `Resource_KEY` attribute must be present in the user profile or the request fails.
 - A modify request is submitted for each user. Provisioning will determine which resources need to be synchronized.
 - When the workflow returns to the reconciliation post provisioning, the attribute values in Select Identity are updated based on the provisioning result and post provisioning policy.
 - The user profile is checked against service mapping (no fixed attributes will be assigned to the user), to see if any new service assignments are gained or lost. The services to be checked should meet these conditions:
 - The service uses this resource or the user is already assigned to the service.
 - The user has access to all the resources required by the service.
 - The service has attributes that are changed.
- Action = **delete**
 - If the user account does not exist on Select Identity, the action is skipped and reported.
 - Run this user profile against the service mapping, delete user from all assigned services that use this resource and delete the entitlements and `Resource_KEY` attribute.
 - There is single request per user. When the workflow returns to the reconciliation post provisioning stage, the attributes and values are updated on Select Identity based on the provisioning result and post provisioning policy.

Troubleshooting Reconciliation Jobs

This section contains the following

- Issues common to all reconciliation methods
- Job Submission and Execution
- Request Processing
- Reporting
- Performance and Configuration Tips

Issues common to all reconciliation methods

Failed requests using reconciliation to add thousands of users from Oracle

Consider the following options:

- Use user import to add new users to the system when you plan to add thousands of users to the system.
- Ask your DBA to schedule more frequent jobs to gather statistics.
- Run the following script manually every 50K to 100K users:

```
exec dbms_stats.gather_schema_stats ('OVSI_SCHEMA_NAME',  
estimate_percent=>DBMS_STATS.AUTO_SAMPLE_SIZE, method_opt=>'FOR ALL  
COLUMNS SIZE SKEWONLY', cascade=>TRUE);
```

- Make sure old execution plans are flushed from the system. Flush all execution plans, using this command:

```
alter system flush shared_pool;
```

➤ Other smart databases may encounter this same issue.

Why does the job not run?

- The start date and time of job is not properly set.
- The clocks are not synchronized for all nodes in the cluster.
- There are too many batch jobs stuck in the "Process" state and the automatic Select Identity system recovery can not restore them. Ask the database administrator to update the BATCH table.
- There are too many reconciliation tasks stuck in the "Prepare Data" state and the automatic Select Identity system recovery can not restore them. Ask the database administrator to update the RECONCILIATIONDATA table.

What is the job execution order

- 1 Automatic file upload executes jobs for authoritative resources prior to non- authoritative resources. For the same resource, the order is in alphanumeric order on date time portion of the filename.

- The number of simultaneous authoritative jobs can be controlled through `TruAccess.properties` items `si.recon.server.num` (default to 3) and `truaccess.recon.task.simul.auth` (default to 2 times of reconciliation server number)
- The number of simultaneous non-authoritative jobs can be controlled through `TruAccess.properties` items `si.recon.server.num` (default to 3) and `truaccess.recon.task.simul.nonauth` (default to 2 times of reconciliation server number)
- The number of simultaneous jobs controls the number of jobs staying in "Prepare Data" state. If the number exceeds this limit, new jobs stay in "Scheduled" state until one "Prepare Data" job moves on to the next state or the blocking period expires. The blocking period is specified through the `TruAccess.properties` item `truaccess.recon.task.check.threshold`, default to 120 (one hour since the check is done every 30 seconds).

Job Submission and Execution

Administrative Role and Context

Who is the reconciliation administrator?

- For File Upload: the sign-on administrator creating the reconciliation job.
- For Web Service: the administrator specified in the SPML operational attributes.
- For Polling: the resource owner.

Administrative role error or context error during process?

- The administrator who submits the reconciliation job must have the reconciliation functionality assigned.
- The administrator must have all service/all context access.

File Upload

Why can't I upload the file?

Check that the file upload directory specified in `TruAccess.properties` file has been created with proper read/write permission. The file size should be less than 5 MB.

Why does the job not pick up the file?

- Check that the reconciliation root, staging and backup directories have been created with proper read/write permission. In a cluster environment, these directories should be shared across nodes.
- If a sub-directory is specified for an automatic reconciliation job, sub-directories with the same name must be created under the root, staging and backup directories with proper read/write permission.

What if the Resource Name is incorrect?

- Resource Name and Separator (`TruAccess.properties` item `com.hp.ovsi.spml.resourcename.separator`, default is "_") is not a prefix of the filename.

- The file extension is not .spml nor .xml.

What if the job picks up an incomplete file?

This usually happens in a cluster environment when reconciliation directories are shared.

The utility that puts files into the job root directories should be from the physical node where the directory is shared. The “Move” here is almost an instant operation while the “Move” from logical directories on other nodes may cause a “Copy” and delay.

Web Service and Polling

How do I associate a web service reconciliation request with a task?

If the Web Service request is accepted, a positive SPML response is sent back to the Web Service submission client. The task id is part of the response.

How many sub SPML requests (records) should be in a Web Service SPML batch request?

- If a SPML batch contains too few records, it increases the time for task switching and slows down the overall throughput.
- Too many records in a batch increases the memory used for parsing and pushes back the response acknowledgement. It also slows down the report generation for the task.
- According to the system configuration, the recommended most common batch size is between 100 and 1000 records.

Request Processing

SPML Data: XML Format

Does Select Identity have SPML schema validation?

- No. The syntax is standard SPML format; invalid syntax is rejected.
- The majority of the variation resides in the data value portion; that is, the semantic explanation of the tags.

Why does the parser report bad XML format even though the file can be opened by a Web Browser or other XML validating tool without error?

- There might be hidden invalid characters at the beginning of the file that the SPML parser used by Select Identity cannot accept. These characters, called a byte-order-mark (BOM), are placed at the beginning of a Unicode character encoded text file by some editors (including Windows Notepad). The BOM should be removed using a hex editor, designed to display file contents as bytes.
- The hidden characters could be viewed through advanced text editors.



All XML and SPML files used by HP OpenView Select Identity should be encoded using the Unicode character set UTF8.

Is there any tool provided by Select Identity to generate reconciliation SPML?

- Select Identity provides a utility that converts LDAP diff to SPML.
- Select Identity provides a utility that polls LDAP Change Log to SPML.

- Some Select Identity agent-based connectors can generate SPML for Web Service reconciliation.

SPML Data: Fields

What are the operation attributes in SPML used for?

Operational attributes are used to control the policy and access of SPML operations. Select Identity reconciliation SPML operational attributes are used mainly for authentication and user search in Select Identity storage.

Assume each task has a batch and the single request is just a batch with one record. The operational attributes at batch level apply to all records (sub requests). The operation attributes at record level only applies to this record.

What are the required fields for the reconciliation SPML used in file upload?

- At batch level, the required fields for all types of request are:
Operation Attribute Key fields: Field names used to search user in Select Identity.
- At record level, the required for add requests are:
 - Operation Attribute Resource Key: used to establish user resource relationship in Select Identity.
 - For an authoritative resource, the attribute fields for search fields must exist. The Select Identity user name must exist (Operational Attribute `taUserName` or mapped attribute field), or the fields used by user ID generation through generator or SPML filter external call must exist.
 - For non-authoritative resource, either the attribute fields for search fields or the identifier must exist. Search fields take precedence and identifier can only be used to search one field.
- At record level, the required for modify and delete requests are:
Either the attribute fields for search fields or the identifier must exist. Search fields take precedence and identifier can only be used to search one field.

What are the required fields for the reconciliation SPML used in Web Service?

In addition to required fields used by file upload, the following operational attributes must be provided:

- Administrator username
- Administrator password
- Resource name
- Reverse Sync Flag set to `true`

What are the attribute or modification fields in SPML?

- These are resource attribute names, not the Select Identity attributes, nor the actual attribute names on the resource. There is an XML mapping file for each resource that maps the Select Identity resource attribute to actual resource attribute.
- Resource attributes should be mapped to Select Identity attributes in order to import the data correctly. If there are unmapped attributes in SPML, the `TruAccess.properties` item `truaccess.spml.fieldmismatchpolicy` controls the process policy:
 - `=1`: An exception is thrown on this record

- =2 (default): The unmapped fields are skipped and displayed in the report.

SPML Data: Operations

What are the available Modification Operation Tags for single value attribute?

- add: add or modify
- replace (default): modify
- delete: delete

What are the available Modification Operation Tags for multi value attribute?

- add/delete: delta to the current values
- replace (default): the final value set

What the operations for the attributes with <attribute> tag?

Add or replace

What are the extended requests supported by reconciliation?

Reconciliation only supports standard add/modify/delete requests. An extended request must be converted to add/modify/delete request by the SPML Filter which is actually a preprocessor. The sample SPML filter provided converts enable/disable and reset password extended request to modify requests.

Rules: General

What are reconciliation rules used for?

- A reconciliation rule is an XML file used to specify the actions to take based on user properties during the reconciliation process.
- The reconciliation rule can be configured per resource/event combination.

What are the main elements in a rule?

- A unique rule name, which does not need to match the XML filename.
- Two input user properties map: current and future.
- Condition and Operations putting desired actions into an output action map.

What is the limitation of the current reconciliation rule?

- No schema validation.
- No support for multi value user attribute condition judge.

Rule: Actions

What are the desired Select Identity actions supported by the rule?

- User Level: Enable/Disable/Terminate
- All Services related to the source reconciliation resource: Add/Delete/Enable/Disable
- Specific Services: Add/Delete/Enable/Disable

- If the user properties can trigger multiple actions, what is the execution order?
 - User Level Action; only one action can be taken.
 - All Service Action (when there is no User Level Action, add before enable | disable or just delete).
 - Individual Service Action (when there is no User or All Service level action, ordered by add, delete, enable, disable).

Why are the actions in the rule not taken?

- There might be an error in the rule XML itself or runtime error.
- The user properties do not satisfy the conditions for the action.
 - ▶ Rule Actions are only candidates. For example, if the action says the user could get service S1 if the location is TX, but S1 has some constraints disqualifying the user from getting the service, the user will not get the service S1. Use detailed reporting to help identify the problem.

Engine Process: User Attribute Analysis

How do I manipulate incoming attributes?

Use an SPML filter to pre-process the incoming SPML request.

The interface of the filter is defined in the Select Identity interfaces package distributed along with the core application. Customers can develop their own filter. This is also a sample filter shipped with the core product that transforms some connector specific extended SPML requests (enabled/disabled, reset password etc.) to standard SPML modify request.

There are several requirements for the filter implementation:

- The reconciliation engine expects the same request to return back to the engine after the filtering. For example, an add request can not be returned as modify request.
- Each invocation to the filter is passed with a single SPML request, not a batch request.
- The attributes in the SPML request are all Select Identity resource attributes.

Is there an example case when the incoming attributes should be manipulated?

Yes

If source resource A's last name change triggers a full name in Select Identity and other resources (B), full name is not an attribute on the actual resource.

The resolution steps are:

- 1 Add a field (FullNameA) in resource A's mapping file and then modify resource A.
- 2 In Select Identity, create a FullName attribute mapping to A's FullNameA and B's FullName. For FullNameA, allow sync-in but no sync-out.
- 3 Develop an SPML filter that checks if the Modify request contains the FirstName, Middle Name or LastName, then add a FullNameA in the SPML Modify Request and pass it back to the reconciliation engine.

What attribute changes from the source resource are taken by Select Identity?

All the resource attributes mapped to Select Identity attributes that have the sync-in flag checked.

Why do I get "No valid attribute change" error even if the SPML request contains attributes/modifications?

- The sync-in flag may not be checked.
- There is no difference between the values in SPML request and the value in Select Identity system storage. The real change check is performed as a safety measure to avoid circular update between resource and Select Identity.

Can a disabled user be modified?

This is determined by the `TruAccess.properties` item `com.hp.ovsi.modify.disableduser`

Policy Evaluation

What happens if there is no policy configured for a resource/event combination?

The system default policy (R3.x compatible behavior) is used.

When does the User Action have any effect?

When the Resource Action is "Accept"

What is the order for User Action?

- **Add Event:** Basic < Auto < Rule < Rule or Auto. Basic is included in all other actions.
- **Modify Event:** Basic < Sync < Auto < Rule < Rule or Auto < Auto and Rule. Basic and Sync are included in all other actions. Basic is included in Sync action also.

What is the User Action picked for a Modify Event?

- 1 A user action can be specified for each Select Identity attribute in the attribute configuration page.
- 2 When a reconciliation modify request is received, the real changed attributes are collected.
- 3 For each changed attribute, its **Reconcile User Action** is checked. If there are user actions configured, the one has the highest order is selected.
- 4 If there is no attribute level policy found, the resource level policy is used.
 - ▶ For example, customer could configure the resource level modify policy to be "Auto" and last name attribute reconciliation policy to be "Sync." If the modify request contains last name change, only a sync resource action is executed without service level operations.

When do I have to select a rule in policy configuration?

If the user level action is configured to involve rule.

If not selected, only the include basic and/or sync actions are executed or an exception is thrown.

Why there is no "Auto and Rule" user action for Add event?

For the add event, there are usually only "Add Service" actions.

Since the rule added services may assign additional attributes to the user that makes the auto services ineligible, it is usually not proper to execute auto actions and then rule actions.

Operation Analysis and Select Identity Request Composition

How many requests are composed for a reconciliation record?

- 0 or 1
- For a modify event, if the user action is set to “Basic”, then the modification is saved directly to the Select Identity database without a request.
- For a delete event from a non-authoritative resource, if the user does not have any service related to this resource assigned, the user's resource related is removed from the Select Identity database without a request.

Is the new user always added to Select Identity for an add event from the authoritative resource?

No. This is different than the behavior in R3.x

A request is built and may be rejected by approvers.

What operations does a Select Identity reconciliation add/modify request contain?

- Reconciliation add/modify request is a compound request containing many targets. Each target is for one operation.
- The first target is always the pure attribute change target that does not orient for any service.
- The follow up targets are oriented for operations (add/modify/delete) for different services. Each involved service has a target.
- The execution order is the order of the targets.

What is the difference between Rule and Auto service assignment?

Auto service assignment is a passive operation. Whether the user can be assigned to a service is based on the existing user properties and the following conditions to be satisfied, during the process (new attributes can be granted to the user and the source resource will NOT be provisioned back):

- User has access to all resources a service requires.
- User has all the required attribute values for a service.
- User has a matching value for the service's context attribute.
- User has matching fixed attribute values for a service based on context value. The set of all fixed attribute values starting from the current context/business role level all the way up to the root is a subset of the corresponding user attribute set. If there is no fixed attribute value defined, this evaluation is skipped.
- User has matching optional attribute values for a service based on context value. The optional attribute value set is gathered starting from the current context/business role. If there is a set defined, the lookup stops. Otherwise, the search goes all the way up to the root. The optional value set is a super set of the corresponding user attribute set. If there is no optional attribute value defined, this evaluation is skipped.
- If an attribute is present, the length and pattern must match the field definition.
- If an attribute is present and the field has constrained value set, the value must be one of the constrained values. The constrained values are either from the static attribute definition or external call is such a function is defined for the attribute

Rule service assignment is an active operation. It is just like assigning a service to the user through Select Identity's Web UI, and the validation is the same as Add Service through Web UI. New attributes may be granted to the user (fixed attributes) and the source *may* be provisioned back.

What services are selected as Auto service assignment candidates for add/modify/delete?

- All enabled services related to the source resource
- All services the users have been assigned
- New Admin Service and Aggregate (Composite) Services are excluded. There should be more strict control on adding users to administrator services. Aggregate service may contain administrator services and also needs special handling in delegated situation.

How do I perform reconciliation Move User?

- In Select Identity, the “Move User” means if the service context attribute value of a user is changed that should affect the service assignment change and fixed attribute change.
- The normal “Move” from UI takes care of these changes. It is an active operation.
- The reconciliation service operation is more of a passive operation which does not grant new attributes to the user.
- To simulate “Move” in reconciliation situation, there are two ways:
 - Set the reconciliation modify policy as “Auto” and then use the reconciliation Default Move workflow or a similar workflow to add users back to services through the delegated mode in the Add2Services block of the workflow.
 - Set the reconciliation modify policy as “Auto and Rule,” use reconciliation Default workflow and a rule to add user back to the services.

Workflow: Design and Selection

What is the basic requirement for a reconciliation workflow?

Look at the packaged **ReconciliationDefaultProcess** workflow.

The workflow must have Provisioning and Post Provisioning Block with Reconciliation Post Provision as the application, Exception Handling and state transition

Must a reconciliation request have a workflow?

- No. The reconciliation process can run without a workflow.
- No workflow processing speeds up the performance, it performs the provisioning and post provisioning, but misses the flexibility and flow control provided by the workflow.

What is the workflow to run for a reconciliation request?

It is the workflow specified in the resource's user reconciliation policy.

If there is no policy configured for the resource, it is the ReconciliationDefaultProcess or something specified through the `TruAccess.properties` item `truaccess.fixedtemplate.reconciliation`.

- ▶ Note: the workflow template in service event handler “Reconciliation - Add” is NOT used. Event handler is required to perform validation for rule service add action in the view. The workflow template is not needed.

Workflow: Execution

My reconciliation request shows as in process but when I look at it, there is no workflow assigned. Why is this?

There is a possible backlog on the workflow JMS queue. If the workflow queue has pending items, then likely the item will be processed; it is just waiting because of current high volume of requests being processed.

It also may be due to the possible workflow JMS message loss due to application server problems. Please refer to [Appendix 6, Recovering from Abnormal Server Shutdown](#) for more details.

How do I retry a reconciliation request?

Currently there is no UI interface in 4.0 MR to issue a request retry. The reconciliation request retry brings up more complexity to the task control and reporting. Terminate the request and reprocess the reconciliation file to recover.

Approval

Can reconciliation approval update attributes?

No. The approvers can only accept or reject a reconciliation request.

Resource and Select Identity Provisioning

What attributes are synchronized to other resources?

- The resource attributes mapped to the changed Select Identity attributes and whose sync-out flags are checked.
- Required attributes by the resource.

If some of the multiple resources are down or have a provisioning error, what is the effect on reconciliation?

- Auth Add: The first target is to add the user in Select Identity. The other targets are normally for service operations. Provisioning failure only keeps the user from adding to the associated resources.
- Modify (Include Non-Auth Add and Auth Add again for existing user): The first target is the user profile update. The other targets are normally for service operations. Since the first target pushes the changes to all mapped resources, the behavior depends on the workflow variable settings:
 - A workflow provisioning block variable named "stopRequest" was introduced to control reconciliation modify or move user requests. Default value is false.
 - If this variable is set to true, then any resource down or provisioning failure stops the reconciliation process for this user and marks the record as "failed". Neither other resources nor Select Identity data are updated.
 - If this variable is set to false, the normal resources and Select Identity are updated.
- Auth Delete: The user will not be deleted from any other resource and Select Identity.
- Non-Auth Delete: The user will not be deleted from down resources and lose the Select Identity service/resource attributes.

What are the common causes for discrepancies between resources and Select Identity?

- Provisioning failure may cause the discrepancies between source resource and other resources and Select Identity. See above.
- Post-Provisioning failure may cause the discrepancies between resources and Select Identity.
- The Select Identity database is down after the provisioning completes and the auto recovery is unable to resume.

Reporting

Format and Contents

What is the format of the reconciliation report?

The base report format is XML. XML can be easily transformed into other formats. It is also easy for find the error segment of the file when debugging.

Select Identity provides a default transformation to HTML through XSL. Customer can provide their own XSL template through `TruAccess.properties` item `truaccess.recon.report.htmlxsl` (full path of XSL file)

If the customer only wants the original XML, set the above XSL path to an invalid one.

What is in a reconciliation report?

- A reconciliation report has a task summary section that includes the overall process status and counters for all records in the task.
- The reconciliation report includes detailed section for each record:
 - The user profile operation
 - Service operation
 - Attaches the original SPML data if there is error.
 - The reason for the error or no-processing may or may not be included. If the report style is set to detailed, then the details are saved for more operations.
 - More important, prints the request ID of the record with the error.

How do I determine the status of one record?

A record can end in one of the following status:

- No Op: no operation is performed
- Complete: all operations completed successfully
- Fail: all operations have failed
- Partial Success: some operations have failed
- Ineligible Rule Operations: since rule operations are only candidates, the customer can set the option to mark ineligible operations after validation as Failed or No Op

How do I determine the status of the task?

- Successful: If all records are Complete or No Op

- Failed: Every record failed
- Partially Successful: Some records failed or partially failed

Generation and Email Notification: Type and Time

How many reconciliation reports are normally sent and when are they sent?

- An initial report email is sent after the task is analyzed and moves into submit status with only the summary. This is not always sent because the job can be completed at that time.
- A final report email with only the summary is sent after the task finishes.
- An on demand report is sent when the customer clicks the reporting button in the reconciliation task list. The detailed report is sent as the attachment of the email. If the `TruAccess.properties` item `com.hp.ovsi.volumedata.report.compressed` is set (default = true), the attachment is compressed in the Zip format.

Who gets email notification?

- The job submitter
- CC email set when submitting a file upload job
- Web service e-mails authenticated user
- Polling resource owner


Why do I get a summary email but no attachment for an on demand report?

The file size is over the limit set for email attachment. This is set in the `TruAccess.properties` item `truaccess.batch.report.file.maxsize` (bytes, default to 1M)

If the file is too big, the locations of the files are included in the email body.

Why are the reports not generated?

- Check the reporting parameters in `TruAccess.properties`
- Check the file directory parameters in `TruAccess.properties`
- The task has not started
- There is a report generation exception

 Large reports may take some time to complete.

Generation and Email Notification: Web Service Report

When will the Web Service reconciliation have a report?

Due to the amount of Web Service tasks, it is up to the system administrator to set up the report options in the `TruAccess.properties` file as follows:

`si.recon.webservice.report.generate=`

- 0 - Never
- 1 - Only Initial Report when no request is processed

- 2 - always

General Error Handling

What are the reconciliation error categories?

- System Configuration
- SPML Data
- Select Identity Runtime Processing

My reconciliation report lists the request as completed with no errors but viewing the request shows that it failed, or the record has an error but the request status shows OK.

The error may happen in the workflow blocks after the post provisioning when the report data is prepared.

Check the Ineligible Rule Operations settings.

A large number of my reconciliation requests failed for whatever reason. How do I recover?

- Identify the type of error.
- Analyze the processing error
 - If Select Identity does not take the changes, the request can be resubmitted
 - If Select Identity saves the changes, the request can not be resubmitted. In this case, delegated operations may be required or the database administrator has to help to correct the data in the database.
 - In Process Requests: Please refer to [Appendix 6, Recovering from Abnormal Server Shutdown](#) for more details.
- Correct the configuration or SPML data error and resubmit

Can I resend all my reconciliation request including the successful ones?

Yes. The successfully processed ones are ignored as there are no valid changes.

Where can I pull the file to resend it?

The reconciliation back up directory specified in the `TruAccess.properties` file

Where can I get the data sent via Web Service?

Select Identity currently does not provide the capability to pull the Web Service reconciliation SPML data from database storage.

We recommend the Web Service client keep a copy of the data.

Performance and Configuration Tips

Configuration

Select Identity Resource Settings

The following Select Identity resource settings all affect the overall reconciliation performance:

- Entitlements Caching
- User Reconciliation Policy:
 - Serial Process or Not (requests are processed in the receiving order; needs to scan database to establish dependencies)
 - Audit or Not
 - User Action (More operations to perform, slower)
 - Workflow Selected (Simpler or No workflow runs faster)

User Search Fields

- To improve the user search speed, have the search fields created as the extended columns in the TAUSER table and indexed.
- If there are multiple search fields, use the more unique one as the first search field.

Workflow External Call vs. SPML Filter

- Both can be used to manipulate the data. An SPML filter focuses on the incoming resource data, while a workflow external call focuses on Select Identity attributes and outgoing data.
- A SPML filter is executed per resource, while a workflow external call is executed per workflow.
- A SPML filter class loading is faster. Execution is also faster since a workflow external call is invoked for every request target.
- A workflow external call can check the service. An SPML filter can not.

Data Caching

Properly set up, the data caching can avoid the time spent on loading data from the Select Identity database and reduce the risk of having inconsistent data in the processor memory and versus the Select Identity database.

Select Identity provides caching for the following types of most frequently used data:

- Resources
- Attribute Definitions
- Services
- Rules

There are cache controlling parameters in `TruAccess.properties` file for each category. Please read the configuration document for more details.

Application Server/Cluster and Database Server

What is the recommended configuration for a server or servers in my cluster if I want to optimize some for reconciliation and others for UI Requests?

Refer to the *HP OpenView Select Identity Installation and Configuration Guide* for detailed configuration settings and other information.

6 Recovering from Abnormal Server Shutdown

The Server Management function lets you view the status of requests within any designated application server that are running the Select Identity application. Use this functionality to track down all the request instances that are affected by an ungraceful sever shutdown of HP OpenView Select Identity.

This chapter covers the following:

- [Understanding the Server Work List](#)
- [Managing Requests](#)
- [Recovering Requests in the Event of Server Failure](#)

Understanding the Server Work List

The Server Worklist lets you view the status of one or more servers. Use this page to view the status of each server instance that is running or has run the Select Identity application.

The Server Name column identifies the server running (or that has run) the Select Identity application. It contains identification information about the server-like host, port, and so on. This identification information varies among different application servers.

The Server Status is updated periodically by each server running Select Identity. Select Identity updates the status of all of its active Select Identity servers as `Running`. When a server is shutdown gracefully, the server status indicates `Not Running`. If a server is terminated abruptly, its status remains `Running`; however, after a time-out period, its status is changed to `Error`. Thus, due to the length of the time-out period, there may be a delay before an abruptly terminated server reports an `Error` status.

The Start Time indicates when the server started. The Modified Time indicates when server last updated the status. The End Time indicates when the server was gracefully shut down.

See the online help for procedures to view the server work list or individual request status.

Managing Requests

HP OpenView Select Identity generates requests that are sent to the servers that store the supported resources through a connector interface. When requests fail, you have the flexibility to manage the request and determine the next best course of action.

Terminate requests that error or are incorrect. Terminating a request stops the request where it is in the workflow process. Requests that have already completed successfully cannot be terminated.

Refer to the online help for specific details of how to terminate a request.

Recovering Requests in the Event of Server Failure

If a database or application server failure causes an unexpected shutdown, Select Identity requests can become suspended in the In Process state due to loss of the database connection, JMS message store failure, or other abnormal conditions.

Select Identity can recover the majority of these requests without intervention, using the **Terminate and Retry** option of the Server Management section. In addition, the following procedures can help you recover any requests that Select Identity does not recover automatically.

Locating Requests for Recovery

To locate requests for recovery, perform the following steps:

- 1 Perform the following steps to determine the time period during which the database connection and/or Select Identity application server was non-operational. This assists you in isolating the requests that need to be recovered.
 - a Use the database management interface to check the database server downtime.
 - b In Select Identity, open the **Server Management / Service Instance List** page and check the Select Identity server downtime.
- 2 Locate the requests that are in an incomplete state, which were started before or during the server failure. If a request should have been completed by now, given the typical performance of your environment, then it is likely a request that needs to be recovered.

Recovery Procedures

The recovery procedures you need to follow depend upon the type of request to be recovered:

- [Recovering Delegated and Self-Service Requests](#)
- [Recovering User Reconciliation Requests](#)
- [Recovering Bulk Add and Move Requests](#)
- [Recovering Service Reconciliation Requests](#)

Recovering Delegated and Self-Service Requests

To recover delegated and self service requests, perform the following steps:

- 1 If the failure caused a Select Identity server to shut down in an ungraceful manner, navigate to the **Server Management** section.
 - a If there is any server listed in error status that has links to request IDs in the Request ID column, view that server and inspect each request.
 - b If the request is a Delegated or Self-Service request, you can use the Terminate and Retry actions to resubmit the request.
- 2 Next, open the **User Request Status List**

Use the search filters to locate requests created within the downtime period and which remain in the “In Process” state.

- 3 Copy and save the request IDs.

- 4 Select all of the requests using the check boxes in the list.
- 5 Click **Terminate**, and then click **Retry**.

Recovering User Reconciliation Requests

User Reconciliation requests cannot be retried directly from the Select Identity request interface as Delegated and Self-Service requests can. To recover User Reconciliation requests, perform the following steps:

- 1 Navigate to the Reconciliation Task List.
- 2 Locate tasks during the downtime period remaining in the “Submitted” or “In Progress” states.
- 3 Copy and save the IDs of the tasks.
- 4 Query the Reconciliation Task report and save the reports.
- 5 Navigate to the Request Status List.
- 6 Search for requests created within the downtime period that remain in the “In Process” state.
- 7 Select all affected requests and click **Terminate**.
- 8 After all requests are terminated, return to the Reconciliation Task List.
- 9 Verify the Submitted or In Process tasks have completed after a suitable period of time.
- 10 Query the reconciliation task reports again to locate any tasks needing additional intervention.
- 11 If incomplete tasks were submitted through File Upload, prepare a new file with the records that are not processed properly and submit the file again.
- 12 If incomplete tasks were submitted through Web Service or resource polling, prepare Web Service reconciliation SPML files with the data from the report and submit the Web Service requests again through Web Service client.

Recovering Bulk Add and Move Requests

Bulk add and move requests cannot be recovered directly from the Select Identity user interface. To recover requests of this type, perform the following steps:

- 1 Save bulk job initial reports, if they are sent.
- 2 Query the **Bulk Add** report from the **Bulk Task List** interface.
- 3 Navigate to the **Request Status List**.
- 4 Use the search filter to search for requests created within the downtime period and that remain in the “In Process” state.
- 5 Select all requests that fit the search criteria and click **Terminate**.
- 6 After all the requests are terminated, upload the bulk add job again, or create a new bulk move job.

Recovering Service Reconciliation Requests

This type of request cannot be retried directly from the request interface. To recover requests of this type, perform the following steps:

- 1 Open the **Request Status List**.
- 2 Use the search filters to locate any service reconciliation requests that were initiated during the server downtime period and that remain in the “in process” state.
- 3 Select each request by checking the box on the left of the request entry in the list.
- 4 Click **Terminate**.
- 5 After the request has terminated, you must submit a new Service Reconciliation job, using the settings for each affected request, from the **Service** tab.

A Creating Select Identity Rules

Reconciliation rules specify the operations required to add and maintain a user account based on user properties. Rules control user assignment and provisioning during request processing.

You add and manage rules through the **Rules** pages. You must create an XML file that adheres to the Document Type Definition (DTD) to build a rule. See [Complete DTD Rule Definition](#) on page 143 for details. Once the file is complete, upload it to the Rule List. Rules on the list are available for assignment to resources and can be used in workflows and external calls.

The following types of operations are supported:

- Add Service
- Delete Service
- Enable Service
- Disable Service
- Enable User
- Disable User
- Terminate User

This appendix covers the following topics:

- [Reconciliation Rules](#)
- [Exclusion Rules](#)
- [Managing Rules](#)

Understanding User Status Dependencies

The following limitations exist based on the status of a user account:

- New user accounts may only be added
- Disabled user accounts may have services deleted, but not added.
- Disabled users may only be enabled with the authority to add
- Enabled user accounts may receive any action
- Terminated user accounts may not be enabled

New Users

When a reconciliation request is received by Select Identity from an authoritative resource and all required conditions are met, the system makes the change. The event must be supported based on the status of the user, the rule definition, and the rule policy. (See [About Resource Classes](#) on page 38 for a description of authoritative resources.)

When a rule is attached to a resource, if the user meets the criteria specified in the rule and qualifies for a service, the user account is added or changed based on criteria in the rule. Rules can also be used to assign additional services to a user during **Move User** using an external call to read the rule from a designated workflow.

In an external call, the rules are defined in the workflow external call type. `Rule` is defined as a parameter in this workflow external call. The rule defined in the external call should be defined with the add rule functionality. When you add the rule, the XML rule file is uploaded to the Select Identity database.

Reconciliation Rules

Select Identity allows you the flexibility to create an XML file that adheres to the DTD rule to manage reconciliation events. No complete reconciliation rules exist when you install the system although a template is included. Each rule must be built and customized to meet your company's specific needs.

The XML rule files are loaded into Select Identity. Once available they are triggered by workflows and external calls. Each policy is evaluated in a logical order.

This section covers the complete DTD for reconciliation rules.

Tips

Here are some tips to consider while writing XML reconciliation rules:

- Make sure that your parameters are consistent throughout the file.
- Do not nest your code. Select Identity does not support nesting.
- Review your key field (`keyField`). Make sure the attribute field selected is a unique identifier in Select Identity.
- Keep in mind that requests may be audited. The audit flag is set based on the resource policy.
- Remember that the processor class should implement a `process(int request id)` method which the request broker calls back. An integer identifies a processor class that will be loaded dynamically to process the request without a workflow.
- Review the event types you used to make sure they are spelled correctly.
- Remember that event types are case sensitive.

Complete DTD Rule Definition

Following is the complete Select Identity DTD rule. Each element contains an explanation of its children and attributes:

```
<!-- Rules are scripts that are executed with reference to specific events
  @title TruAccess Rule Language
  @root Rule
-->
<!--
A Rule has a collection of InputObjects and one or more scripts that work on the input
object
-->

<!ELEMENT Rule (InputObject*,Script+)>
<!--
RuleId is an identifier that is used to identify the rule Comment is a piece of information
associated with a rule Under debug mode the Comment is printed in the log
-->
<!ATTLIST Rule
RuleId ID #REQUIRED
Comment CDATA #IMPLIED>
<!--
InputObject is an object that is used in the rule. Most of the time the InputObject will be
created and passed to the rule. Optionally the input object will be created if specified.
Please note that one should not specify variables as InputObjects. Variables are treated
differently.
-->
<!ELEMENT InputObject EMPTY>
<!--
type is the type of the object. It can be any valid fully qualified Java type name. The
primitive types can be declared as int, String and boolean. The actual type when passed
needs to be Integer, String and Boolean.
name is the name of the object
create specifies whether the object has to be created. The assumption is that the object
supports a no-argument constructor
-->
<!ATTLIST InputObject
type CDATA #REQUIRED
name CDATA #REQUIRED
create (yes|no) #IMPLIED>
<!--
Script is the body of a Rule, where conditions are checked and actions taken
-->
<!ELEMENT Script (ConditionScript | ActionScript | AssertScript | PlainText |
PrintScript)*>
<!--
Comment in a script can be used to trace the execution for debugging
-->
<!ATTLIST Script
Comment CDATA #IMPLIED>
<!--
ConditionScript is a condition statement
-->
<!ELEMENT ConditionScript (Condition,TrueAction,FalseAction?)>
<!--
Comment in a script can be used to trace the execution for debugging
-->
<!ATTLIST ConditionScript
Comment CDATA #IMPLIED>
<!--
ActionScript models action. currently only one type of action is specified
-->
<!ELEMENT ActionScript (AssignStmt)>
<!--
```

```

Comment in a script can be used to trace the execution for debugging
-->
<!ATTLIST ActionScript
Comment CDATA #IMPLIED>
<!--
AssignStmt allows assignment of values to fields or variables
-->
<!ELEMENT AssignStmt (Field, Expression)>
<!--
Condition is a boolean expression
-->
<!ELEMENT Condition (Not?, (OrCondition | AndCondition | UnitCondition)>
<!--
OrCondition models logical or
-->
<!ELEMENT OrCondition (UnitCondition+)>
<!--
AndCondition models logical and
-->
<!ELEMENT AndCondition (UnitCondition+)>
<!--
UnitCondition is a nested condition or a relation
-->
<!ELEMENT UnitCondition (Condition | Relation)>
<!--
Relation is between two expression
-->
<!ELEMENT Relation (Expression, Expression?)>
<!--
Relation supports the following operations:
<ul>
<li><b>eq</b> equal</li>
<li><b>ne</b> not equal</li>
<li><b>gt</b> greater than</li>
<li><b>lt</b> less than</li>
<li><b>ge</b> greater than or equal</li>
<li><b>le</b> less than or equal</li>
<li><b>contains</b> contains</li>
<li><b>startswith</b> startswith</li>
<li><b>endswith</b> endswith</li>
<li><b>matches</b> matches</li>
<li><b>eqic</b> equals ignore case</li>
</ul>
contains is a special operation. It can be applied to String, Map and Collection types.
startswith, endswith, matches, equal can be applied to String only. The semantics are the
same as that of java string class.
<!ATTLIST Relation
op ( eq | ne | gt | lt | ge | le | contains ) #REQUIRED>
<!--
TrueAction is executed when the condition is true
-->
<!ELEMENT TrueAction (Script*)>
<!--
FalseAction is executed when the condition is false
-->
<!ELEMENT FalseAction (Script*)>
<!--
Field represents a field in a bean or a variable
-->
<!ELEMENT Field EMPTY>
<!--
name is the name of the field. If the field has a . then it is assumed that it is an
attribute of an InputObject otherwise it is a temporary variable.
type is the type of the variable. The following types are supported:
<ul>
<li><b>int</b> integer</li>

```



```

<li><b>boolean</b> boolean</li>
<li><b>java.lang.String</b> Java String</li>
<li><b>java.util.Collection</b> Java Collection</li>
<li><b>java.util.Map</b> Java Map</li>
</ul>
For variables, the collection is implemented as an ArrayList and Map is implemented as a
HashMap
fieldKey is used to access the object in the collection/map
hasG(S)etter and setter is used to generate accessing functions
Y => has a function get<Name> and set<Name>
N => no function ... direct access assuming that it is public
D => has generic function: get(<name>) and set(<name>) of string type
-->
<!-- ATTLIST Field
name CDATA #REQUIRED
type ( int | boolean | java.lang.String | java.util.Map | java.util.Collection ) #REQUIRED
fieldKey CDATA #IMPLIED
hasGetter ( Y | N | D ) "D"
hasSetter ( Y | N | D ) "D" >
-->
Expression is an expression
-->
<!-- ELEMENT Expression (Field | FixedValue | ArithExp | BoolExp)>
-->
BoolExp is a boolean expression
-->
<!-- ELEMENT BoolExp (Field | True | False | Relation | Condition)>
-->
True represents a true value
-->
<!-- ELEMENT True EMPTY>
-->
False is a false value
-->
<!-- ELEMENT False EMPTY>
-->
ArithExp is an Arithmetic Expression
-->
<!-- ELEMENT ArithExp (AddExp | MultExp | Field | FixedValue)>
-->
AddExp is an additive expression
-->
<!-- ELEMENT AddExp (ArithExp,ArithExp)>
-->
AddExp supports two operations
<ul>
<li><b>plus</b> addition</li>
<li><b>minus</b> subtraction</li>
</ul>
AddExp support concatenation of two Strings
-->
<!-- ATTLIST AddExp
op ( plus | minus ) #REQUIRED>
-->
MultExp supports two operations
<ul>
<li><b>mult</b> multiplication</li>
<li><b>div</b> division</li>
</ul>
-->
<!-- ELEMENT MultExp (ArithExp,ArithExp)>
-->
<!-- ATTLIST MultExp
op ( mult | div ) #REQUIRED>
-->
FixedValue is a literal which can be a quoted string or an integer
-->

```

```

<!ELEMENT FixedValue (#PCDATA)>
<!--
AssertScript is an assertion
Condition is checked and if it is false then an exception is generated with the Message
-->
<!ELEMENT AssertScript (Condition,ExceptionName?,Message?)>
<!--
Comment in a script can be used to trace the execution for debugging
-->
<!ATTLIST AssertScript
Comment CDATA #IMPLIED>
<!--
This signifies a not condition
-->
<!ELEMENT Not EMPTY>
<!--
A message for assertion failure
-->
<!ELEMENT Message (#PCDATA)>
<!--
Any BeanShell script
-->
<!ELEMENT PlainText (#PCDATA)>
<!--
class name of the assertion failed exception
-->
<!ELEMENT ExceptionName (#PCDATA)>
<!--
Statement to print an expression
-->
<!ELEMENT PrintScript (Expression)>

```

Managing Rules

Create your own rule XML files, or edit and use the XML rule template provided in Select Identity to add rules. Use existing rules by copying the rule, renaming the file, and making modifications when you need a new rule similar to an existing rule. Maintain rules by downloading the file, making required changes and uploading the file again. All XML rule files must be uploaded for the most recent version to appear in the rule list.

Rules on the list are available to those functions that are supported by the reconciliation process. Your imported rules can be attached to resources, used in external calls, or triggered by workflows as required.

For details on creating, adding, modifying, viewing or deleting rules, see the Select Identity online help.

Troubleshooting Reconciliation Rules

The list that follows contains answers to frequently asked questions:

[The initial AddRequest creates the user in Select IdentitySelect Identity, however it does not create a Workflow instance for the request.](#)

Make sure the user attributes are stored correctly.

Validate that the context attribute value is specified as required. If the context attribute value is not specified or is not provided at all the user is created, but not assigned to any service. As this point the request is created only when the user is assigned a service of any kind.

How do I enable and disable all services for a specific user?

Send a modify request to Select Identity specifying the correct attribute values based on the resources the target services provision to disable or enable all services for a specific user.

The `<value>code:urn:trulogica:concerro:2.0#generalError,desc:Exception in processing request: null,</value>` error appears.

Review your key field (`keyField`). Make sure the attribute field selected is a unique identifier in Select Identity.

Validate the name of the field. The key field should be a Select Identity attribute field name. Look in your XML file and compare the resource attribute name with the Select Identity name (**Service Studio > Resources > Modify Resources**). View the Select Identity name on the right in the `Attribute` column. If the `keyField` name is incorrect, then change it now.

Exclusion Rules

There are three types of exclusion rules:

- [Service Exclusion Rule](#)
- [Attribute Exclusion Rule](#)
- [Entitlement Exclusion Rules](#)

Service Exclusion Rule

This is a conflict rule used in workflows. It compares rules. For example, if a user is in Service 1, then the user cannot also have Service 2. Replace Service 1 and `gvA1` with the name of the service to check for. Also replace `gvL3` with the name of the service to exclude.

```
//When one of these values exists on user valueList.add("Service 1");
valueList.add("gvA1");
//This value is excluded
valueMap.put("gvL3", valueList);
```

Attribute Exclusion Rule

This rule specifies excluded values:

```
//Entitlement name goes here
String entitlementName = "LDAP72_ENTITLEMENTS;
String attributeName = "FirstName";
```

Replace `LDAP72_ENTITLEMENTS` with the name of the entitlement attribute to exclude. And replace `"FirstName"` with the single value attribute.

```
//These values are excluded
excludedList.add("HR Managers");
excludedList.add("QA Managers");
//When this value is used
valueMap.put("Greg", excludedList);
```

Replace `"HR Managers"` and `"QA Managers"` with the entitlement values to exclude, and replace `"Greg"` with the attribute name to check.

Entitlement Exclusion Rules

Similar to service rules these rules state if the user belongs to Entitlement 1, the user cannot also belong to Entitlement 2, or if the user does belong to both Entitlement 1 and Entitlement 2, the user cannot belong to Entitlement 3.

EntitlementExclusion

```
//Entitlement name goes here  
String EntitlementName = "LDAP72_ENTITLEMENTS";
```

Replace LDAP72_ENTITLEMENTS with the name of the entitlement attribute to exclude.

```
//These values are excluded  
valueList.add("HR Managers");  
valueList.add("QA Managers");  
//When this value is used  
valueMap.put("PD Managers", valueList);
```

Replace "HR Managers" and "QA Managers" with the entitlement values to exclude, and replace "PD Managers" with the entitlement value to check for.

EntitlementAndExclusion

```
//Entitlement name goes here  
String EntitlementName = "LDAP72_ENTITLEMENTS";
```

Replace LDAP72_ENTITLEMENTS with the name of the entitlement attribute to exclude.

```
//If all of these condition values are used  
valueList.add("HR Managers");  
valueList.add("QA Managers");  
//Then this value is not allowed  
valueMap.put("PD Managers", valueList);
```

Replace "HR Managers" and "QA Managers" with the entitlement values to exclude, and replace "PD Managers" with the entitlement value to check for.

Sample Rules

This section provides sample rules supporting a variety of actions. These are samples only. Many of the actions would not be included in the same rule. For example, you would not add a user and delete the same user moments later.

The `RuleId` MUST match the rule name defined in the reconciliation policy. The conditions are based on the user attribute map whose key is the Select Identity attribute name and value is the string value.

Multi-value attribute conditions are NOT supported.

This section includes the following:

- [Rule Standards and ServiceNameMap](#)
- [Sample Rule One](#)
- [Sample Rule Two](#)

Rule Standards and ServiceNameMap

This rule provides examples of multiple actions for both single service and all services scenarios. The user actions are specified in the `ServiceNameMap`. The following table is the list of tags for user actions.

Key	Meaning	Value	Meaning
.*-	User Level	Disable	Disable User
.*-	User Level	Enable	Enable User
.*-	User Level	Terminate	Terminate User
*	All Services (for specified resource)	+OK	Add All Services
*	All Services (for specified resource)	-OK	Delete All Services
**	All Services (for specified resource)	Disable	Disable All Services
**	All Services (for specified resource)	Enable	Enable All Services
Service Name	String name of a specific service	+OK	Add to this Service
Service Name	String name of a specific service	-OK	Delete from this Service
Service Name	String name of a specific service	Disable	Disable this Service
Service Name	String name of a specific service	Enable	Enable in this Service

Sample Rule One

```
<?xml version="1.0" standalone="no"?>
<!--
<!DOCTYPE Rule PUBLIC "http://www.truologica.com/truaccess/rule" "file:///C:/
sanjoy/TruAccess/scriptengine/src/rule/Rule.dtd">
-->
<Rule RuleId="Resource_ReconRule" Comment="Reconciliation Resource Rules">
  <!-- name of the resource, input parameter -->
  <InputObject name="ResourceName" type="java.lang.String"/>
  <!-- input parameter, user "final" attribute set -->
  <InputObject name="AttributeMap" type="java.util.HashMap"/>
  <!-- input parameter, user "current" attribute set -->
  <InputObject name="OldAttrMap" type="java.util.HashMap"/>
  <!-- service name -->
  <InputObject name="ServiceNameMap" type="java.util.HashMap"/>
  <Script>
    <ConditionScript Comment="Service Operation if conditions meet, Pro-
```

```

cess=SvcAssign">
    <Condition>
        <AndCondition>
            <UnitCondition>
                <Relation op="contains">
                    <Expression>
                        <Field name="AttributeMap" type="java.util.Map"
fieldKey="0"/>
                    </Expression>
                    <Expression>
                        <FixedValue>&quot;Company&quot;</FixedValue>
                    </Expression>
                </Relation>
            </UnitCondition>
            <UnitCondition>
                <Relation op="eq">
                    <Expression>
                        <Field name="AttributeMap" type="java.util.Map"
fieldKey="&quot;Company&quot;"/>
                    </Expression>
                    <Expression>
                        <FixedValue>&quot;GE&quot;</FixedValue>
                    </Expression>
                </Relation>
            </UnitCondition>
        </AndCondition>
    </Condition>

    <TrueAction>
        <ActionScript Comment="Assign Services">
            <AssignStmt>
                <Field name="ServiceNameMap" type="java.util.Map"
fieldKey="&quot;AcctManagers&quot;"/>
                <Expression>
                    <FixedValue>&quot;+OK&quot;</FixedValue>
                </Expression>
            </AssignStmt>
            <AssignStmt>
                <Field name="ServiceNameMap" type="java.util.Map"
fieldKey="&quot;PayrollManagers&quot;"/>
                <Expression>
                    <FixedValue>&quot;-OK&quot;</FixedValue>
                </Expression>
            </AssignStmt>
            <AssignStmt>
                <Field name="ServiceNameMap" type="java.util.Map"
fieldKey="&quot;AllManagers&quot;"/>
                <Expression>
                    <FixedValue>&quot;Enable&quot;</FixedValue>
                </Expression>
            </AssignStmt>
            <AssignStmt>
                <Field name="ServiceNameMap" type="java.util.Map"

```

```

fieldKey="&quot;HRManagers&quot;"/>
    <Expression>
        <FixedValue>&quot;Disable&quot;</FixedValue>
    </Expression>
</AssignStmt>
</ActionScript>
</TrueAction>
</ConditionScript>
<ConditionScript Comment="All Service Operation if conditions meet,
Process=AllSvcAssign">
    <Condition>
        <AndCondition>
            <UnitCondition>
                <Relation op="contains">
                    <Expression>
                        <Field name="AttributeMap" type="java.util.Map"
fieldKey="0"/>
                    </Expression>
                    <Expression>
                        <FixedValue>&quot;Company&quot;</FixedValue>
                    </Expression>
                </Relation>
            </UnitCondition>
            <UnitCondition>
                <Relation op="eq">
                    <Expression>
                        <Field name="AttributeMap" type="java.util.Map"
fieldKey="&quot;Company&quot;"/>
                    </Expression>
                    <Expression>
                        <FixedValue>&quot;AM&quot;</FixedValue>
                    </Expression>
                </Relation>
            </UnitCondition>
        </AndCondition>
    </Condition>

    <TrueAction>
        <ActionScript Comment="Assign Services">

            <AssignStmt>
                <Field name="ServiceNameMap" type="java.util.Map"
fieldKey="&quot;*&quot;"/>
            <Expression>
                <FixedValue>&quot;+OK&quot;</FixedValue>
            </Expression>
        </AssignStmt>
        <AssignStmt>
            <Field name="ServiceNameMap" type="java.util.Map"
fieldKey="&quot;*&quot;"/>
            <Expression>
                <FixedValue>&quot;-OK&quot;</FixedValue>
            </Expression>

```

```

        </AssignStmt>
        <AssignStmt>
            <Field name="ServiceNameMap" type="java.util.Map"
fieldKey="&quot;**&quot;"/>
            <Expression>
                <FixedValue>&quot;Enable&quot;</FixedValue>
            </Expression>
        </AssignStmt>
        <AssignStmt>
            <Field name="ServiceNameMap" type="java.util.Map"
fieldKey="&quot;**&quot;"/>
            <Expression>
                <FixedValue>&quot;Disable&quot;</FixedValue>
            </Expression>
        </AssignStmt>
    </ActionScript>
</TrueAction>
</ConditionScript>
<ConditionScript Comment="User level Operation if conditions meet,
Process=uerLevelOperation">
    <Condition>
        <AndCondition>
            <UnitCondition>
                <Relation op="contains">
                    <Expression>
                        <Field name="AttributeMap" type="java.util.Map"
fieldKey="0"/>
                    </Expression>
                    <Expression>
                        <FixedValue>&quot;Company&quot;</FixedValue>
                    </Expression>
                </Relation>
            </UnitCondition>
            <UnitCondition>
                <Relation op="eq">
                    <Expression>
                        <Field name="AttributeMap" type="java.util.Map"
fieldKey="&quot;Company&quot;"/>
                    </Expression>
                    <Expression>
                        <FixedValue>&quot;HP&quot;</FixedValue>
                    </Expression>
                </Relation>
            </UnitCondition>
        </AndCondition>
    </Condition>

    <TrueAction>
        <ActionScript Comment="Assign Services">
            <AssignStmt>
                <Field name="ServiceNameMap" type="java.util.Map"
fieldKey="&quot;*-&quot;"/>
                <Expression>

```



```

        <FixedValue>&quot;Enable&quot;</FixedValue>
    </Expression>
</AssignStmt>
<AssignStmt>
    <Field name="ServiceNameMap" type="java.util.Map"
fieldKey="&quot;*-&quot;"/>
    <Expression>
        <FixedValue>&quot;Disable&quot;</FixedValue>
    </Expression>
</AssignStmt>
<AssignStmt>
    <Field name="ServiceNameMap" type="java.util.Map"
fieldKey="&quot;*-&quot;"/>
    <Expression>
        <FixedValue>&quot;Terminate&quot;</FixedValue>
    </Expression>
</AssignStmt>
</ActionScript>
</TrueAction>
</ConditionScript>
</Script>
</Rule>

```

Sample Rule Two

The following is a sample of an XML rule for reconciliation. This checks to see if a user exists on an authoritative resource called LDAPv3_Auth and has an attribute or field called Company with a value of ABCCorp. If so, the rule adds new user to the services reconSvc1 and reconSvc2.

```

<?xml version="1.0" standalone="no"?>
<!--
<!DOCTYPE Rule PUBLIC "http://www.trulogica.com/truaccess/rule" "file:///C:/
sanjoy/TruAccess/scriptengine/src/rule/Rule.dtd">
-->
<Rule RuleId="LDAPv3_Auth_ReconRule" Comment="Reconciliation Authoritative
Resource Service Assignment Rules">
    <InputObject name="ResourceName" type="java.lang.String"/>
    <InputObject name="AttributeMap" type="java.util.HashMap"/>
    <InputObject name="ServiceNameMap" type="java.util.HashMap"/>
    <Script>
        <ConditionScript Comment="Check Resource Name and Company Name">
            <Condition>
                <AndCondition>
                    <UnitCondition>
                        <Relation op="eq">
                            <Expression>
                                <Field name="ResourceName" type="java.lang.String"/>
                            </Expression>
                            <Expression>
                                <FixedValue>&quot;LDAPv3_Auth&quot;</FixedValue>
                            </Expression>
                        </Relation>
                    </UnitCondition>

```

```

<UnitCondition>
  <Relation op="contains">
    <Expression>
      <Field name="AttributeMap" type="java.util.Map" fieldKey="0"/>
    </Expression>
    <Expression>
      <FixedValue>&quot;;Company&quot;;</FixedValue>
    </Expression>
  </Relation>
</UnitCondition>
<UnitCondition>
  <Relation op="eq">
    <Expression>
      <Field name="AttributeMap" type="java.util.Map"
fieldKey="&quot;;Company&quot;;"/>
    </Expression>
    <Expression>
      <FixedValue>&quot;;ABCCorp&quot;;</FixedValue>
    </Expression>
  </Relation>
</UnitCondition>
</AndCondition>
</Condition>
<TrueAction>
  <ActionScript Comment="Assign Services">
    <AssignStmt>
      <Field name="ServiceNameMap" type="java.util.Map"
fieldKey="&quot;;reconSvc1&quot;;"/>
      <Expression>
        <FixedValue>&quot;;+OK&quot;;</FixedValue>
      </Expression>
    </AssignStmt>
    <AssignStmt>
      <Field name="ServiceNameMap" type="java.util.Map"
fieldKey="&quot;;reconSvc2&quot;;"/>
      <Expression>

```

B The SPML Generator Utility

HP OpenView Select Identity provides a utility called the SPML Generator, which converts CSV and XML file formats into Service Provisioning Markup Language (SPML). While there are many ways to create SPML files, this is a convenient way for developers who are more familiar with CSV and XML to create SPML files for user import and reconciliation.

This chapter covers the following:

- [SPML Generator Utility Package](#)
- [Understanding Dependencies](#)
- [Running the Utility](#)
- [Properties Descriptions for the Properties Configuration Files](#)

SPML Generator Utility Package

The SPML Generator package includes all the files you need to get started. Copy the files from the `utilities` directory on the HP OpenView Select Identity product CD to any directory convenient for you.

File	Description
<code>runGenerator.bat</code>	Windows DOS batch file used to run the SPML Generator utility. This batch file is based on one of the <code>spmlgenerator*.properties</code> configuration files listed in this table. See Running the Utility on page 156 for details. You must change the batch file to show the location of the <code>JAVA_HOME</code> environment variable according to the installation on the target host.
<code>runGenerator.sh</code>	UNIX shell script used to run the SPML Generator utility. This script is based on one of the <code>spmlgenerator*.properties</code> configuration files listed in this table. See Running the Utility on page 156 for details. You must change the <code>@shell</code> script file to show the location of <code>JAVA_HOME</code> according to the installation on the target host.
<code>SPMLGenerator.jar</code>	SPML Generator utility executable files

File	Description
<code>spmlgeneratorcsv.properties</code>	Sample configuration file for CSV-based input for user import. Edit this file according to your requirements.
<code>spmlgeneratorreconcsv.properties</code>	Sample configuration file for CSV-based input for Reconciliation. Edit this file according to your requirements.
<code>spmlgeneratorxml.properties</code>	Sample configuration file for XML-based input for user import. Edit this file according to your requirements.
<code>spmlgeneratorreconxml.properties</code>	Sample configuration file for XML-based input for Reconciliation. Edit this file according to your requirements.

Understanding Dependencies

The following files must be on your system for the utility to work:

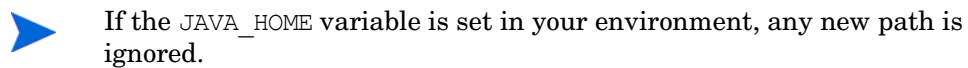
- `commons-logging.jar` — for logger and log factory
- `ovsii18n.jar` — for internationalization

Running the Utility

Complete the following steps to run the SPML Generator utility:

- 1 Copy the package to a single directory.
- 2 Be sure JRE (≥ 1.4) is installed properly and included in the path.
- 3 Select one of the following sample files to use as a template based on your requirements.
 - `spmlgeneratorcsv.properties` — Used to convert CSV-format to SPML for user import.
 - `spmlgeneratorreconcsv.properties` — Used to convert CSV-format to SPML for Reconciliation.
 - `spmlgeneratorxml.properties` — Used to convert XML-format to SPML for user import.
 - `spmlgeneratorreconxml.properties` — Used to convert XML-format to SPML for Reconciliation.
- 4 Edit the batch file or shell script to do the following:
 - Enter the `JAVA_HOME` location according to the installation on the target host.

Enter the appropriate properties configuration file name as a command line argument. Do not use the dot and the extension



- 5 Run the batch file or shell script.

Example

Following is an example of the `runGenerator.bat` file that is ready to be run to convert a CSV format to SPML for Reconciliation:

```
set JAVA_HOME=C:\bea\jdk142_05

SET CLASSPATH=./SPMLGenerator.jar;./commons-logging.jar;./
ovsiil8n.jar;

%JAVA_HOME%\bin\java -classpath %CLASSPATH%
com.ovsi.spmlgenerator.core.SPMLGenerator spmlgeneratorreconcsv
```

Properties Descriptions for the Properties Configuration Files

Following are the properties and their descriptions by category as they appear in any of the properties configuration files. The categories are listed in the order of relevance to users.

#csvplugin parameters

- `reader.filename`

File name with the full path of the CSV input file.



Note that back-slashes (\) in a Windows path name must be specified as double back-slashes (\\).

For example:

```
reader.filename=C:\\tools\\eclipse\\workspace\\SPMLGenerator\\bin\\sample
files\\reconcsv.csv
```

- `reader.colnames`

List of column names for the data to be specified in the CSV input file. Be sure the name and order of the columns are not altered. The order is referenced in the properties file and in the SPML Generator utility.

For example:

```
reader.colnames={username,ssn,firstname,lastname,email,profile,role,accou
nttype,primaryAcctValue,recontype}
```

- `reader.coldelimiter`

Separator character that is used between two columns in the CSV file.

For example:

```
reader.coldelimiter=#
```

- `reader.datadelimiter`

Separator character that is used between multiple values to be given in one column. This is typically used in the case of multi-valued attributes.

For example:

```
reader.datadelimiter=,
```

- `reader.quotechar`

Escape sequence to represent a double quote within the data in the CSV file.

For example:

```
reader.quotechar=\"
```

- `reader.encoding`
Encoding of the characters in the CSV file.

For example:

```
reader.encoding=UTF-8
```

Reconciliation or User Import Flag

- `writer.isrecon`
Flag to specify whether the conversion run is for Reconciliation or user import.
For Reconciliation, the value is **Yes**. For user import the value is **no**.

For example:

```
writer.isrecon=no
```



- If the reconciliation value in `writer.isrecon` is **yes**, then specify the following reconciliation values.
- If the reconciliation value in `writer.isrecon` is **no** then specify the **# User Import global operational attributes** values.

- `writer.reconcol`
Field that specifies which column in the CSV format represents the Reconciliation Action Type. Refer to `reader.colnames` above.

The value specified for this property refers to the Reconciliation Type column in `reader.colnames`. The CSV data records typically have a value of Add, Modify or Delete as the Reconciliation type.

For example:

```
writer.reconcol=recontype
```

- `writer.reconopattr`
Specifies which item key in **writer.map** below (SPML file properties) maps to the Reconciliation Operational Attributes.

For example:

```
writer.reconopattr=operation
```

User Import global operational attributes

- `writer.globalop.keyFields`
Specifies the key fields under the global operational attributes section of the SPML file.

For example:

```
writer.globalop.keyFields=UserName,Email
```

- `writer.globalop.isprimary`
Specifies whether the SPML file has a global operational attribute entry for the primary user.

For example:

```
writer.globalop.isprimary=yes
```

- `writer.globalop.primaryAcctKey`

Specifies which attribute represents the Primary Account Key in the global operational attributes.

For example:

```
writer.globalop.primaryAcctKey=Email
```

Specify adduser operational attributes

- `writer.requestop.isUIDGenerated`

Set to **yes** if the `userId` needs to be generated.

For example:

```
writer.requestop.isUIDGenerated=yes
```

- `writer.requestop.taUserNameAttr`

Refers to the column corresponding to the `UserName` attribute.

For example:

```
writer.requestop.taUserNameAttr=username
```

- `writer.requestop.taResourceKey`

Refers to the column corresponding to the `Resource Key`.

For example:

```
writer.requestop.taResourceKey=ssn
```

- `writer.requestop.primaryAcctKey`

Specifies the string representation of `primaryAcctKey`.

For example:

```
writer.requestop.primaryAcctKey=urn:hp:selectidentity#taResourceKey
```

- `writer.requestop.primaryAcctValue`

Specifies the string representation of `primaryAcctValue`.

For example:

```
writer.requestop.primaryAcctValue=primaryAcctValue
```

- `writer.requestop.typecolumn`

Specifies the string representation of `typecolumn`.

For example:

```
writer.requestop.typecolumn=accounttype
```

- `writer.requestop.multientcolnames`

Provides column names corresponding to the multi-valued columns.

For example:

```
writer.requestop.multientcolnames=profile,role
```

#SPML file parameters

- `writer.spmlfile`

Specifies the full path and name of the output SPML file.



Note that slashes (\) in a Windows path name must be specified as double slashes (\\).

For example:

```
writer.spmlfile=C:\\tools\\eclipse\\workspace\\SPMLGenerator
\\bin\\samplefiles\\spmlfromxml.xml
```

- `writer.usersperfile`

Specifies the number of users to include in a file in order to limit the file size. The file size should not grow larger than 10 MB in order to avoid degradation of performance. Follow the suggestions below or experiment to find out what works best for you.

— For ten or fewer attributes, set the value at 500 users per file.

— For eleven or more attributes, set the value at 250 users per file.

For example, if the value is set to 2, the file is closed after writing the SPML entries corresponding to two user requests. If there are more users in the CSV input file (or other input file), multiple files will be written with the same name. Each file will be given a suffix integer numbered from 1 to $n-1$, where n is the total number of files required to represent the whole input as SPML.

For example:

```
writer.usersperfile=2
```

- `writer.isauthoritative`

Specifies whether the resource represented is authoritative or non-authoritative. Enter **yes** for authoritative.

For example:

```
writer.isauthoritative=yes
```

- `writer.map1`

Specifies one of the several maps you can specify. These maps provide a way to map the column from the input file under `reader.colnames` to the SPML output.

`map1` maps `username` in `reader.colnames` to the SPML output.

For example:

```
writer.map1=username|UserName
```

- `writer.map2`

`map2` maps `ssn` in `reader.colnames` to the SPML output.

For example:

```
writer.map2=ssn|SSN
```

- `writer.map3`

`map3` maps `firstname` in `reader.colnames` to the SPML output.

For example:

```
writer.map3=firstname|FirstName
```


- `writer.map4`
`map4` maps `lastname` in `reader.colnames` to the SPML output.
For example:
`writer.map4=lastname|LastName`
- `writer.map5`
`map5` maps `email` in `reader.colnames` to the SPML output
For example:
`writer.map5=email|Email`
- `writer.map6`
`map6` maps `profile` in `reader.colnames` to the SPML output
For example:
`writer.map6=profile|Profile`
- `writer.map7`
`map7` maps `role` in `reader.colnames` to the SPML output
For example:
`writer.map7=role|Role`
- `writer.map8`
`map8` maps `operation` in `reader.colnames` to the SPML output
For example:
`writer.map8=operation|Operation`

C Attribute Mapping Utility

The attribute mapping utility helps you create or modify XML and XSL mapping files of the connectors. This feature is supported on database connectors and bi-directional LDAP connectors. For other connectors, the resource typically defines a fixed set of attributes. Hence, the mapping files cannot be modified.

In the case of database connectors, you can provision users and entitlements into a database. Since the database schema can be defined in a number of ways, the mapping files can also vary widely, and can be modified. For instance, the attribute mapping utility gives you the flexibility to map table columns and stored procedures to HP OpenView Select Identity attributes.

After you map the attributes and save, an XML file for forward mapping and an XSL file for reverse synchronization are generated. In case of bi-directional LDAP connectors, the attribute mapper retrieves the groups and the corresponding attributes under the directory server. You are allowed to map these attributes and generate the schema mapping file. Learn more about deploying connectors in [Installing Connectors](#) on page 37.

Attribute Mapping Utility Overview

The attribute mapping utility lets you load the resource schema directly from the resource and map its attributes onto Select Identity attributes, thereby creating an XML mapping file that is used by the Select Identity connectors. The connectors are supplied with a default XML mapping file and this is usable in most cases. You can use the attribute mapping utility to create a new mapping file or edit an existing mapping file. The attribute mapping utility can also let you provision entitlements into the resource.

Accessing the Attribute Mapping Utility

To access the attribute mapping utility from the Select Identity home page, do the following:

- 1 While deploying a new connector on Select Identity using the **Manage Connectors** option, select the **Yes** radio button under the **Mapper Available** section to make the attribute mapping utility available for that particular connector.

Connector Name:	Pool Name:	Mapper Available:
GenSQLConnector	eis/Gen-SQL2000Connector	<input checked="" type="radio"/> Yes <input type="radio"/> No

- 2 On the Select Identity home page, click **Service Studio -> Resource**. The **Resources** List appears.
- 3 Click **Add New Resource**. The **Add New Resource: Basic Information** page appears.
- 4 Enter the necessary values into each field (refer to the connector’s installation guide for more information on the values needed to deploy the resource), and then click **Next**.
- 5 On the Access Info page, enter the necessary connection credentials, which depend on how the database connector and agent are installed and configured:
 - Using a JDBC data source without an agent installed:
In this configuration, the connector performs operations on the database directly through JDBC calls.

Table 11 Connection Credentials

Field	Value
Mapping File	The name of the XML file that will be generated.
JDBC Datasource String	The JNDI name of the JDBC data source that was created on the Select Identity server to connect to the target database.

Make sure all of the other fields are empty.

Using a JDBC driver without an agent installed

The connector uses the JDBC driver to communicate with the database. You must specify all parameters except the agent port and JDBC data source

Table 12 Connector Parameters

Field	Value
SQL URL	The name of the JDBC driver to use to connect to the database.
Mapping File	The name of the XML file that will be generated.
Server Name	The name of the database server.
Server Port	The database server’s listening port.
Username	The database user ID.

Field	Value
Password	The password of the specified user.
Database/Service Name	The name of the database.
Database Driver String	The JDBC driver being used.

Using a JDBC driver with an agent installed

If the agent is installed and a JDBC driver is used to communicate with the database, you must specify all parameters except the JDBC data source.

Table 13 Agent Parameters

Field	Value
SQL URL	The name of the JDBC driver to use to connect to the database.
Mapping File	The name of the XML file that will be generated.
Server Name	The name of the database server.
Server Port	The database server's listening port.
Username	The database user ID.
Password	The password of the specified user.
Database/Service Name	The name of the database.
Database Driver String	The JDBC driver being used.
Agent Port	The port of the listening agent.

- 6 Click the **Edit** link next to the Mapping File field.

To create or modify a mapping file of an existing resource, resource from the **Resource List** page, click **Modify**, and then click **Resource Access Information** in the left pane. In the **Resource Access Information** page, click **Edit** next to the Mapping field.

When you click **Edit**, the attribute mapping utility page appears and connects to the database using the values entered on the Access Info page. If you are creating a mapping file, a new XML file is created inside the folder path `com/truologica/truaccess/connector/schema/spml` under the directory specified as the `com.hp.ovsi.connector.schema.dir` property of the `TruAccess.properties` file, which resides in the `install_dir/sysArchive` directory on the Select Identity server. You should set this property to the top-level directory where the mapping files reside.

If no value was specified for the property, the `com/truologica/truaccess/connector/schema/spml` directory structure is created in the application server's home directory.

If the specified mapping file exists, the attribute mapping utility appears, connects to the database, and loads the existing settings in the mapping file.

Configuring the JDBC Datasource for a Connector

Ensure that for creating the JDBC Datasource on WebLogic, you perform the following configuration:

- Uncheck the **Honor Global Transactions** option.

- Check the **Emulate Two-Phase Commit for non-XA Driver** option.

This configuration must be done to allow the newly created datasource to co-exist with the Select Identity JDBC datasource. For WebSphere, you have to create the connection pool using an XA Driver.

- ▶ In order to connect to MS SQL Server 2000 using an XA Driver, JTA related stored procedures need to be installed on the database. If they are not available, you can use the JDBC-driver-based method to connect to database.

Attribute Mapping Utility Menus and Pages

If you access the attribute mapping utility by loading its URL in a browser, the page, which appears first, prompts you to enter connection information.

If you are editing an existing XML file, the mapped attributes are listed in the **Attribute Mappings** section of the page.

The attribute mapping utility home page contains the following:

- Menus
- Mapping pages

Attribute Mapping Utility Menus

The following menus and options are available.

File Menu

The **File** menu has the following options:

- **Save Mapping File**
Saves the XML and XSL file in the directory specified in the Base Directory field on the Select Identity server. If the base directory is not specified while logging in, Select Identity uses the value of `com.hp.ovsi.connector.schema.dir` property. If none of these values are available, the files are stored in the home directory of the application server.
- **Save As**
Saves the XML and XSL files with another name, other than the one specified when you logged in to the utility. This displays a pop-up dialog where you need to enter base directory and file name.
- **Download Mapping File**
Lets you download the XML file from the Select Identity server if you are running the utility from a remote client. Displays a download dialog allowing you to save the XML file locally.
- **Download Reverse Synchronization File**
It lets you download the XSL file from the Select Identity server if you are running the utility from a remote client. Displays a download dialog, which enables you to save the XSL file locally on the client.
- **Reload**
Reloads the XML file in the attribute mapping utility.

- **Disconnect**
Disconnects the connection from the utility to the database.

Entity Menu

An entity is a logical grouping of attributes for users or groups (entitlements). By default, a user entity exists. If you want to provision entitlements, you need to create an additional entity. Each new entity is a group entity that represents user entitlements.

These are the options on this menu:

- **Select Entity**
Select the entity to edit.
- **Add Entity**
Create a new entity in the utility. Allows you to map attributes for entities other than users. Displays a dialog prompting you to name the new entity.
- **Edit Entity**
Edit an entity name using this option. The default entity (user) cannot be edited.
- **Delete Entity**
Deletes the currently selected entity. You cannot delete the default entity (user).

Mapping Operations Menu

This menu lists all of the operations that can be performed using the utility, which are the same operations through which you are guided if you use the wizard (see [Mapping Pages](#) on page 167). The following shows the menu when the user entity is selected:

These are the options on this menu:

- **Attributes Home**
Displays the **Attributes** page, allowing you to map database columns to Select Identity attributes.
- **User Enable/Disable Attribute Configuration**
Displays the Enable/Disable page, allowing you to set values that are assigned when a user is enabled or disabled during provisioning.
- **Define Entity Operations**
Displays the Specify Supported Operations page, allowing you to define the operations the connector can perform on the schema.
- **Define Relationships**
Displays the Define Relationship Definitions page, allowing you to define how tables in the schema relate.
- **Reverse Synchronization Attributes**
Displays the Reverse Synchronization Attributes page, allowing you to map key fields used during reverse synchronization.

Mapping Pages

Mapping pages are the interface through which you can modify mapping files, define operations, define relationship between tables, and so on. You can access the mapping pages either by user entity or by group entity. By default, you can access these pages by user entity.

When you use user entity, you can find five mapping pages. You can navigate to them either by **Previous** or **Next** button at the bottom of the window, or using the **Mapping Operations Menu**.

Attributes Home (Page 1)

This page lets you add or delete attributes from a database to Select Identity.

If you create a new mapping file, no mappings are listed on this page in the beginning. If you load an existing mapping file, the page lists the mappings currently defined in the file.

Each database column listed under Resource Field has a link. If you click the link, the properties of that mapping are displayed, which you can modify.

Mapping Attributes to Select Identity

The attribute mapping utility maps the attributes of a database resource to Select Identity. To achieve this mapping, perform the following steps:

- 1 Load the mapping page of the connector, through the Select Identity user interface (see [Usage Scenarios](#) on page 170).
- 2 Click **Add Attribute Mappings** to open the Filter Schema page, which reads the database schema and provides a listing in a pop-up dialog. This displays the database schema in a hierarchical tree.

Select the schema, and then click one of the action menus at the bottom:

- Click **Filter Attributes** to display the next level of database schema. For example, if you select a schema, and then click **Filter Attributes**, the Tables and Stored Procedures under the Database Schema will be displayed.
 - Click **Map Attributes** to display the **Map Attributes** page where the Database schema can be mapped into the mapping file. When a table schema is selected and **Map Attributes** is clicked, all the columns of the table become available for mapping.
- 3 Click **Map Attributes** to display the Map Attributes window, which reads the database schema of the resource and provides a listing on the left side of the window of the Selected Items from the Filter Schema page.



If you try to edit a mapping file without connecting to the database, the schema is not displayed.

- 4 Select the schema items (table, columns or stored procedure parameters) from the left frame, and then click **Add Attribute Mappings** to map them into the mapping file.

After mapping the required attributes, click **Finish** to return to the **Attribute** home page that displays the attribute mappings and the associated properties.

- 5 Save the modified or newly created mapping file by clicking **File** -> **Save Mapping File**.

For advanced operations, you can navigate to other pages.

User Enable/Disable (Page2)

Lets you select the Status attribute from the **Database** menu and set the values assigned when a user is enabled or disabled during provisioning.

Specify Supported Operations (Page 3)

Lets you define the operations that the connector can perform on the entity.

Define Relationship Definitions (Page 4)

The Define Relationship Definitions page defines how tables in the schema relate and lets you specify how user and entitlement information is linked in the database schema. The Define Relationship Definitions page defines which table columns are used to store the LINK (user-entitlement relationship) information for User or Group Entities.

Reverse Synchronization Attributes (Page 5)

Lets you map key fields used during reverse synchronization. This information is used to generate the corresponding XSL file for the XML mapping file.

For additional operations, access mapping pages through group entity. The follow mapping pages are available:

- 1 Attributes Home Page
- 2 Specify Supported Operations Page
- 3 Define Relationship Definitions Page
- 4 Provisioning Information Page

Provisioning Information page

Displays the attributes mapped for the group entity, except for the attribute linked to the user attribute. You can provision entitlements directly in to the database. The attribute mapping utility provisions group entities only.

See [Mapping Pages](#) on page 167 for procedures that illustrate how to set properties in these sections of the window.

Usage Scenarios

The database schema that stores user and entitlement data can be created in a number of ways. Usage Scenarios describe the most common schema configurations. The scenarios used in this chapter are the most likely configurations and provide a broader view of the required steps to define mappings for XML and XSL files.

Table Scenario

For user provisioning and entitlement provisioning procedures, three tables are used to store provisioning data:

- A user table called `GEN_SQL_USERS` that contains six columns, which define user attributes. For example, the table may contain the `UserName`, `Password`, `Firstname`, `Lastname`, `Email`, and `City` columns, which store user-related information.
- An entitlement table called `GEN_SQL_ENTITLEMENTS` that contains one column, which stores the list of possible entitlements that can be assigned to users. For example, the table may contain the `Entitlement ID` and `Description` columns, which store entitlement information that can be assigned to users.
- A normalized mapping table called `GEN_SQL_USER_TO_ENT` that defines user-to-entitlement mappings and contains two columns, one to store user IDs and one to store the assigned entitlements. For example, the table may contain the `UserName` and `Entitlement ID` columns, which store user name-to-entitlement mappings.

The user column in the mapping table uses the user column in the user table as its foreign key. Similarly, the entitlements column in the mapping table uses the entitlement column in the user table as its foreign key. Finally, when users and entitlements are provisioned, there can be one user ID per user and zero or more entitlements per user.

Stored Procedure Scenario

The database connectors allow the usage of stored procedures for provisioning and other user-related operations. A set of stored procedures can be written to perform operations that are typically done by the connector. These operations include creating a user, modifying a user, deleting a user, and so on. Hence, a stored procedure provides a way to abstract connector operations.

Stored procedures can be customized for the necessary operations. For more information on the most common schema configurations, see [Usage Scenarios](#) on page 170. The scenario used in this chapter is one of the more likely configurations. Here is a description of the scenario:

Three stored procedures are created in the database:

- The `addUser` stored procedure can create a user in the database, in the `Users` table. Values are passed as parameters to the procedure.
- The `modifyUser` stored procedure can modify user attributes for an existing user in the database. This procedure modifies all columns in the `Users` table except the `UserName` and `Password` columns.
- The `deleteUser` stored procedure takes a user name as a parameter and deletes the user from the database table.

Defining User Mappings

The following procedure describes how to map user and entitlement attributes to Select Identity attributes. When you finish defining the mappings, the XML mapping file is generated in the directory specified under `com.hp.ovsi.connector.schema.dir` property in `TruAccess.properties`.

- 1 To invoke the attribute mapping utility, follow steps 1-6 in "Accessing the Attribute Mapping Utility" section for the connector being used.

If the utility successfully connects to the database, the **Attributes** page displays to let you map user attributes. By default, the user entity is selected.

- 2 Map each user attribute defined in the database schema to a Select Identity attribute:
 - a Click **Add Attribute Mappings** to display the Filter Schema page, which provides a listing of the database schema in a pop-up dialog. It displays the database schema in a hierarchical form.

From this page, you can filter the tables, views, and stored procedures that you wish to map from the rest of the database schema. This reduces the schema retrieval time and also provides a better view by showing only the selected schema when mapping attributes.

Note the following points when using attribute mapping utility for Oracle:

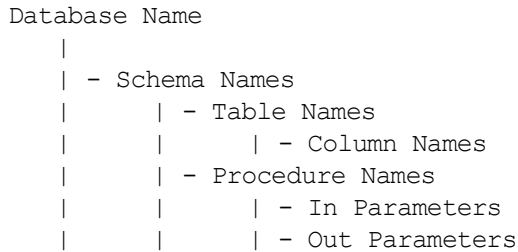
- The Filter Schema page shows the schema for all users in Oracle.
 - Select the schema where the required tables are present and click **Filter Attribute** to show only the tables or stored procedures in that schema.
 - Make sure that you have entitlements for the selected schema on the Filter Schema page. If not, the Map Attributes window will not display a schema. If more than one schema is selected and entitlements are available for a subset of the selected schemas, only the subset of schemas is displayed on the Map Attributes window.
- b To select part of the schema for mapping, select the **Schema** check box and click the **Filter Attributes** button to display the schema. (You can also click the **Map Attributes** button to view the entire schema while mapping attributes.)

The database schema is shown on the Filter Schema page in a tree that you can expand. Here is an explanation of the tree:

```
Database Name
|
| - Schema Names
|   | - Table Names
|   | - Procedure Names
|   | - View Names
```

- c Select the user tables that you wish to map. After you select tables, click the **Map Attributes** button. This displays the Map Attributes window, which displays the expanded schema items selected on Filter Schema page.

Here is the explanation of the tree:



- d From the left side of the window, select the database column(s) that you want to map to a Select Identity attribute:

Then, click **Add Attribute Mappings** button. The selected column is added to the table:

▶ If you map a stored procedure, another section named Stored Procedure Group is added to the page below the Table/Views Group section.

- e In the **SI Attribute** field, enter the name of the Select Identity attribute to which you want to map the selected database column. This value is the same as the value displayed in the **Name** column while creating attributes for the connector's resource on the Select Identity client.

Because **UserName** is the attribute name in this scenario, enter **UserName** in the **SI Attribute** field for the **USERID** column:

▶ The **Required** option is selected if the chosen database column is the schema's primary key.

- f Select the **SI Key** option if the specified attribute is a key field in Select Identity. If you add a stored procedure, you can select multiple attributes as key fields.
- g To treat an attribute as **password** type, which can be used in the Reset Password or Change Password operation in Select Identity, select the **Password Field** option.
- h If you wish to define the actions that the connector can perform for the attribute (not the entire entity), click the **Define Attribute Operations** button and select the allowed operations. A pop-up window appears in which you can select the operations, and then click **OK**.
- i Repeat **step d** through **step h** for each user column you wish to map. The columns are added to the right side of the page in alphabetical order. Here is a snapshot of the defined user mappings for this scenario:

<input type="checkbox"/>	Resource Field	SI Attribute *
Table/Views Group		
<input type="checkbox"/>	schema=dbo.table=Users.column=Email	Email
<input type="checkbox"/>	schema=dbo.table=Users.column=FirstDate	Firstname
<input type="checkbox"/>	schema=dbo.table=Users.column=Lastname	
<input type="checkbox"/>	schema=dbo.table=Users.column=Password1	Password
<input type="checkbox"/>	schema=dbo.table=Users.column=Username1	UserName
<input type="checkbox"/>	schema=dbo.table=Users.column=enabled	

The Select Identity attribute specified for the **STATUS** column is arbitrary. You must map the column but the Enable/Disable settings (**step 3**) control what is provisioned to this column.

- j If you provision entitlements and the database schema includes two tables (one for user data and one for entitlements), you must map the user column from the entitlement table. If the database schema includes at least three tables, including a user-to-entitlement mapping table, you must map the columns of this table accordingly.

- a Click **Next** under the Specify Supported Operations section of the page or select **Mapping Operations** → **Define Relationships**. The Define Relationship Definitions section is displayed.



If the entitlement entity has not been created in the attribute mapping utility, the entitlement entity is not displayed.

If you wish to provision entitlements, complete the steps in [Defining Entitlement Mappings](#) on page 175, and then return to this step.

- b For the user entity, select the column that stores the User entity's Key field for operations that involve entitlements. In this scenario, the user column in the mapping table is selected.



If there is no entitlement table and the connector will not perform group or entitlement operations, you can leave this field empty. If a column is selected from the **Link Field** list, it will not be available for provisioning user information.

- c For the entitlement entity, select the column that stores the user's entitlements. In this scenario, select the entitlement column in the mapping table.

Here is a snapshot of the relationships for the example scenario:

Entity Name	Link Field
User	schema=dbo,table=Users,column=FirstDate
Entitlement	schema=dbo,table=ENTITLEMENTS_1,column=DESCRIPTION

- 6 If you have configured the agent to perform reverse synchronization, you must define reverse mappings:
 - a Click **Next** under the Define Relationship Definitions section of the page or select **Mapping Operations** → **Reverse Synchronization Attributes**. The Reverse Synchronization Attributes section is displayed.
 - b From the Resource User Key drop-down list, select the database column that was mapped to the Select Identity attribute that is the Select Identity key.
 - c From the Resource Password field drop-down list, select the database column that was mapped to the Select Identity password attribute.
 - d From the SI User Key drop-down list, select the Select Identity attribute that is the Select Identity key for the user.
 - e From the SI Password field drop-down list, select the Select Identity attribute that stores the user password.

Here is a snapshot of the values specified for this scenario:

Reverse Synchronization Attributes		
Selected Entity: User	Selected Connector: Gen-SQL2000Connector	Currently Editing: <i>/space/si40/weblogic/schema/com/truilogic/truaccess/connector/schema/spm/attributemap.xml</i>
Page 5 of 5		
Specify the key attributes for reverse synchronization. The values of the key attributes will be used in the XSL file during reverse synchronization.		
XSL Attributes	Attribute Value	Description
Resource User Key	schema=dbo,table=Users,column=Username1	User Key field specified as "SI Key" for Mappings
Resource Password field	schema=dbo,table=Users,column>Password1	Password field specified as "Password Field" for Mappings
SI User Key	UserName	User Key field for SI
SI Password field	Password	SI Password field

- 7 Select **File** → **Save Mapping File** to generate the XML and XSL files in the location specified in the Base Directory field when you connected.

If you wish to download the XML and XSL files to the local system, select **File** → **Download Mapping File** or **File** → **Download Reverse Synchronization File**, respectively. To disconnect from the database or log out from the utility, click **Disconnect**.

Defining Entitlement Mappings

If the database schema contains entitlement data and you wish to provision entitlements, you must create an entitlement entity in the attribute mapping utility. Otherwise, performing the steps in [Defining User Mappings](#) on page 171 generates the XML and XSL files required to provision users only.

Complete the following steps to create an entitlement entity and map its attributes. This procedure assumes that you are logged in to the attribute mapping utility.

- 1 Add the entitlement entity to the attribute mapping utility:
 - a Select **Entity** → **Add Entity**.
 - b In the Entity Name field, specify a name for the entitlement entity.
 - c Click **Add Entity**. The entitlement entity is selected in the utility window.
- 2 Map each entitlement attribute defined in the database schema to a Select Identity attribute:
 - a Click **Add Attribute Mappings** to display the Filter Schema page, which provides a listing of the database schema in a pop-up dialog. This displays the database schema in a hierarchical form.

From this page, you can filter the tables, views, and stored procedures that you wish to map from the rest of the database schema. This reduces the schema retrieval time and also provides a better view by showing only the selected schema when mapping attributes.

- b To select part of the schema for mapping, select the **Schema** check box and click the **Filter Attributes** button to display the schema. (You can also simply click the **Map Attributes** button to view the entire schema when mapping attributes.)

The database schema is shown on the Filter Schema page in a tree that you can expand. Here is an explanation of the tree:

```

Database Name
|
| - Schema Names
  
```

```

|         | - Table Names
|         | - Procedure Names
|         | - View Names

```

- c Select the entitlement tables that you wish to map. After you select tables, click on the **Map Attributes** button. This displays the Map Attributes window, which displays the expanded schema items selected on Filter Schema page.

Here is the explanation of the tree:

```

Database Name
|
| - Schema Names
|     | - Table Names
|     |     | - Column Names
|     |     | - Procedure Names
|     |     | - In Parameters
|     |     | - Out Parameters

```

- d In the left side of the window, select the database column that you would like to map to a Select Identity attribute:

Then, click the **Add Attribute Mappings** button. The selected column is added to the table:

- e In the SI Attribute field, enter the name of the Select Identity attribute to which you want to map the selected database column. For this scenario, map the ENT_NAME column, and enter **Entitlements** in the SI Attribute field.

▶ The **Required** option is selected if the chosen database column is the schema's primary key.

- f Select the **SI Key** option if the specified attribute is a key field in Select Identity. This table column is the entitlement column that stores the entitlements that need to be retrieved using the connector.

- g To treat an attribute as password type, select the **Password Field** option.

▶ Typically, attributes of an entitlement or group entity do not require this password behavior. Hence, this option can be ignored unless required.

- h If you wish to define the actions that the connector can perform for the attribute (not the entire entity), click the **Define Attribute Operations** button and select the allowed operations. A window displays in which you can select the operations, and then click **OK**.

- i Repeat [step d](#) through [step h](#) for additional entitlement columns you wish to map to Select Identity attributes. For this scenario, only the ENT_NAME column is mapped to a Select Identity attribute.

- 3 *If a mapping table is defined in the database schema:*

Map the entitlement column in the mapping table:

- a Select the entitlement column in the mapping table and click **Add Attribute Mappings**. The selected column is added to the list of mapped columns and attributes:

- b In the SI Attribute field, enter an arbitrary value. You simply need to provide a value to enable you to include this column in the mappings.

- 4 *If an entitlements table (in addition to a user table) is defined in the database schema:*
Map the member column in the entitlements table to the entitlement column in the user table:

- a Select the member column in the entitlement table and click **Add Attribute Mappings**. The selected column is added to the list of mapped columns and attributes.
 - b In the `SI Attribute` field, enter an arbitrary value. You simply need to provide a value to enable you to include this column in the mappings.
- 5 Click **Finish** at the bottom of the Map Attributes window.
 - 6 If you have not done so, return to [step 5](#) on page 173 to define the relationship between the user entity and the entitlement entity.
 - 7 Select **File** → **Save Mapping File** to generate the XML and XSL files in the location specified in the Base Directory field when you connected.

If you wish to download the XML and XSL files to the local system, select **File** → **Download Mapping File** or **File** → **Download Reverse Synchronization File**. To disconnect from the database or log out of the utility, click **Disconnect**.

Provisioning Entitlements in the Database

The attribute mapping utility enables you to provision entitlements directly in to the database. Complete the following steps to provision entitlements using the utility:

- 1 Select the entitlement entity.
- 2 Select **Provision Entity** from the Mapping Operations menu. The Provisioning Information section displays and lists the attributes that are mapped for the entity.
- 3 Click **Add Attribute** to modify the values assigned to the attributes.
- 4 Change or enter a value in each Value field for the attributes you wish to provision in the database.
- 5 Click **OK** when you are done. The value is provisioned in the database.

Mapping Stored Procedure Parameters

The following procedure describes how to map stored procedure parameters to Select Identity attributes. When you finish defining the mappings, the XML mapping file is generated in the directory specified in the Base Directory field when you logged in to the attribute mapping utility.



These steps assume that you are logged in to the attribute mapping utility and connected to the database.

Map each parameter for the stored procedure defined in the database schema, as follows:

- 1 Click **Add Attribute Mappings** to display the Filter Schema page, which provides a listing of the database schema in a pop-up dialog. This displays the database schema in a hierarchical form.
- 2 To select part of the schema for mapping, select the **Schema** check box and click the **Filter Attributes** button to display the schema. (You can also simply click the **Map Attributes** button to view the entire schema when mapping attributes.)

The database schema is shown on the Filter Schema page in a tree that you can expand. Here is an explanation of the tree:

```

Database Name
|
| - Schema Names
|   | - Table Names
|   | - Procedure Names
|   | - View Names

```

- c Select the stored procedure tables that you wish to map. After you select tables, click on the **Map Attributes** button. This displays the Map Attributes window, which displays the expanded schema items selected on Filter Schema page. Here is the explanation of the tree:


```

Database Name
|
| - Schema Names
|   | - Table Names
|   |   | - Column Names
|   | - Procedure Names
|   |   | - In Parameters
|   |   | - Out Parameters

```

- 3 From the left side of the window, select the stored procedure parameters that you would like to map to Select Identity attributes. In the example below, all of the parameters for the addUser stored procedure (except the RETURN VALUE) are mapped:

Attribute Mappings

Please map all the required fields from the schema. Required fields are denoted with  symbol

<input type="checkbox"/>	Resource Field	SI Attribute *	Required	SI Key *	Password
Stored Procedure Group					
<input type="checkbox"/>	schema=dbo.procedure=add_Users.column=Email	Email	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
<input type="checkbox"/>	schema=dbo.procedure=add_Users.column=FirstName	FirstName	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
<input type="checkbox"/>	schema=dbo.procedure=add_Users.column=LastName	LastName	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
<input type="checkbox"/>	schema=dbo.procedure=add_Users.column>Password	Password	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
<input type="checkbox"/>	schema=dbo.procedure=add_Users.column=UserName	UserName	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>

- 4 Map the parameters of the modifyUser and deleteUser stored procedures.



Be sure to specify the Select Identity attributes exactly as they appear in the Select Identity client.

Note that similar parameters may occur multiple times in the mapping, such as in this scenario; the Username parameter is required by all stored procedures. HP recommends that all Username parameters and table columns specify the same Select Identity attribute name.

- 5 Map the key column of the table into which the stored procedures are provisioning data. This is the column used to uniquely identify a user in the database tables and is used as a field to verify that the user is available to the connector.

In this scenario, all the stored procedures use the Username column of the Users table as the key field. Hence, this is mapped in the utility.

Resource Field	SI Attribute *	Required	SI Key *	Password
Stored Procedure Group				
<input type="checkbox"/> schema=dbo.procedure=add_Users.column=@Email_4	Email	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
<input type="checkbox"/> schema=dbo.procedure=add_Users.column=@Firstname_2	FirstName	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
<input type="checkbox"/> schema=dbo.procedure=add_Users.column=@Lastname_3	LastName	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
<input type="checkbox"/> schema=dbo.procedure=add_Users.column=@Password_5	Password	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
<input checked="" type="checkbox"/> schema=dbo.procedure=add_Users.column=@Username_1	UserName	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>

- 6 Provide the Select Identity key information for the mapped attributes. There are two cases here:
 - The table column that is mapped is present in a separate group and the **SI Key** radio button is available. A radio button is available because only one **SI Key** attribute is supported for table column-related scenarios. Hence, the table column that is designated as the **SI Key** is the Select Identity key field.
 - The store procedures parameters are grouped separately and the **SI Key** check box is available for each. Check boxes are available because you may have multiple Select Identity keys for stored procedures. Thus, check the **SI Key** option for all of the parameters that are analogous to the Select Identity key attribute of the mapped table column.

In the example scenario, the Username column in the Users table is the key field in Select Identity. For this column, the **SI Key** checkbox is selected in the Table/View Group.

In the Stored Procedure group, the **SI Key** checkbox is selected for all of the parameters that correspond to this table column (mapped to the Username parameter of addUser, modifyUser, and deleteUser stored procedures).

Resource Field	SI Attribute *	Required	SI Key *	Password Field
Stored Procedure Group				
<input type="checkbox"/> schema=dbo.procedure=add_Users.column=@Email_4	Email	false	false	false
<input type="checkbox"/> schema=dbo.procedure=add_Users.column=@Firstname_2	FirstName	false	false	false
<input type="checkbox"/> schema=dbo.procedure=add_Users.column=@Lastname_3	LastName	false	false	false
<input type="checkbox"/> schema=dbo.procedure=add_Users.column=@Password_5	Password	false	false	false
<input checked="" type="checkbox"/> schema=dbo.procedure=add_Users.column=@Username_1	UserName	false	true	false
Table/Views Group				
<input type="checkbox"/> schema=dbo.table=Users.column=Email	Email	false	false	false
<input type="checkbox"/> schema=dbo.table=Users.column=FirstDate		false	false	false
<input type="checkbox"/> schema=dbo.table=Users.column=Lastname		false	false	false
<input type="checkbox"/> schema=dbo.table=Users.column>Password1	Password	false	false	true
<input type="checkbox"/> schema=dbo.table=Users.column=Username1	UserName	true	true	false
<input type="checkbox"/> schema=dbo.table=Users.column=enabled		false	false	false

- 7 Click the **Define Operations** button for the mapped table column that is used for verifying that the user exists in the database table. For this attribute, click **Select All** in the **Retrieval** section; deselect all other operations.
- 8 Click **Define Operations** for each of the stored procedure parameters and select only the relevant operations for which the parameter is used.

For example, the Username parameter of the addUser stored procedure will be used when performing Add operations only. Therefore, only the Create operation must be selected for this parameter.

Similarly, the Email parameter is required by the addUser and modifyUser procedures. The Email parameter of addUser should be enabled for Create operations only and the Update operation should be enabled for the Email parameter of the modifyUser stored procedure.

The Update operation must be enabled for the Username parameter of the modifyUser procedure.

Finally, the Delete operation must be enabled for the Username parameter of the deleteUser procedure.

- 9 Click on the **Finish** button to return to the **Attributes** page. This displays the mappings for the table columns and stored procedure parameters. You can click on the **Edit** link for any attribute to modify the details.
- 10 From the **Mapping Operations** menu, select **Define Entity Operations**.
- 11 On the page that appears, select the operations that are required to support the entity.
- 12 From the File menu, select the **Save Mapping File** option to save the XML file.

User Tables and Stored Procedures Scenarios

The scenarios provided in this section describe possible ways to provision identity information through the use of user tables, stored procedures, and a combination of user tables and stored procedures.

One User Table

A single database table is created and contains all user information. This table has columns for all the user attributes, such as the name, ID, password, email address, and so on.

When the resource is deployed in Select Identity, the connector loads all of the column names from the user table as resource attributes. You can then map Select Identity attributes to the table columns and provision users.

One User Table, One Entitlements Table

A single database table is created and contains all user information. This table has columns for all user attributes, such as the name, ID, password, email address, and so on. In addition, the user table provides a column that contains all of the user's entitlements (such as `memberOf`). This is a multi-valued attribute and the mapping could implement it as a CSV, such as "group1, group2, group3".

Another table is created to store the entitlements. This table has columns that define each entitlement, such as the name and description. In addition, this table has a column for listing users belonging to this entitlement (such as `members`). This is a multi-valued attribute and the mapping could implement it as a CSV, such as "user1, user2, user3, user4". Populate the entitlement table with entitlements that are to be associated and dissociated with the user.

When the resource is deployed in Select Identity, the connector loads all of the column names from the user table as resource attributes. You can then map Select Identity attributes to the table columns and provision users.

Select Identity also retrieves the entitlements during the service configuration or user creation, and connector loads all entries from the entitlement table.

The users are provisioned in the `members` column of the database.

For example, if you create a group entity named `Entitlement` for the entitlement table in the schema, and you provision a user with the administrator entitlement (in Select Identity), the `memberOf` column of that user will have a value "administrator". If more than one entitlement is assigned, they are comma separated, as in this example: `administrator, guest`. Similarly,

the user will be provisioned in the members column of the Entitlement table for each entitlement assigned to him or her. If more than one user is given the same administrator entitlement, the members column for the administrator row will have "user1,user2" as its value.

One User Table, One Entitlements Table, One Map Table

A single database table is created and contains all user information. This table has columns for all user attributes, such as the name, ID, password, email address, and so on.

Another table is created to store the entitlements. This table has columns that define each entitlement, such as the name and description. In addition, this table has a column for listing users belonging to this entitlement (such as members). This is a multi-valued attribute and the mapping could implement it as a CSV, such as "user1, user2, user3, user4". Populate the entitlement table with entitlements that are to be associated and dissociated with the user.

A third table is created to store the user-to-entitlement mappings. This table refers to the primary keys (PKs) of user table and entitlements table. This map table has columns such as user name and entitlements.

When the resource is deployed in Select Identity, the connector loads all of the column names from the user table as resource attributes. You can then map Select Identity attributes to the table columns and provision users.

Select Identity also retrieves the entitlements during the service configuration or user creation, and connector loads all entries from the entitlement table.

If more than one entitlement is assigned to a user, the values are provisioned in separate rows in the map table.

For example, if the user1 user is assigned the administrator and guest entitlements, the map table will have two rows, user1-administrator and user1-guest.

Multiple User Tables

A table is created to store some user information. This table can have columns to store such attributes as user name, ID, password, email address, and so.

Additional tables can be created to store additional user information. These tables can contain columns to store data such as company, department, cost center, address, phone number, and so on. The tables must have a foreign key constraint against the main user table.

When the resource is deployed in Select Identity, the connector loads all of the column names from the user tables as resource attributes. You can then map Select Identity attributes to the table columns and provision users. Provisioning should populate the user tables.

Two User Tables, One Entitlements Table

A table is created to store some user information. This table can have columns to store such attributes as user name, ID, password, email address, and so. In addition, the user table provides a column that contains all of the user's entitlements (such as memberOf). This is a multi-valued attribute and the mapping could implement it as a CSV, such as "group1, group2, group3".

Another table is created to store additional user information. This table can contain columns to store data such as company, department, cost center, address, phone number, and so on. This table has a foreign key constraint against the main user table.

A third table is created to store the entitlements. This table has columns that define each entitlement, such as the name and description. In addition, this table has a column for listing users belonging to this entitlement (such as members). This is a multi-valued attribute and the mapping could implement it as a CSV, such as "user1, user2, user3, user4". Populate the entitlement table with entitlements that are to be associated and dissociated with the user.

When the resource is deployed in Select Identity, the connector loads all of the column names from the user tables as resource attributes. You can then map Select Identity attributes to the table columns and provision users.

Select Identity also retrieves the entitlements during the service configuration or user creation, and connector loads all entries from the entitlement table.

Two User Tables, One Entitlements Table, One Map Table

A table is created to store some user information. This table can have columns to store such attributes as user name, ID, password, email address, and so.

Another table is created to store additional user information. This table can contain columns to store data such as company, department, cost center, address, phone number, and so on. This table has a foreign key constraint against the main user table.

A third table is created to store the entitlements. This table has columns that define each entitlement, such as the name and description. In addition, this table has a column for listing users belonging to this entitlement (such as members). This is a multi-valued attribute and the mapping could implement it as a CSV, such as "user1, user2, user3, user4". Populate the entitlement table with entitlements that are to be associated and dissociated with the user.

A fourth table is created to store the user-to-entitlement mappings. This table refers to the primary keys (PKs) of user table and entitlements table. This map table has columns such as user name and entitlements.

When the resource is deployed in Select Identity, the connector loads all of the column names from the user tables as resource attributes. You can then map Select Identity attributes to the table columns and provision users.

Select Identity also retrieves the entitlements during the service configuration or user creation, and connector loads all entries from the entitlement table.

Multiple User Tables, Multiple Entitlement Tables

Multiple user tables and entitlement tables can be created, where the user information is store in several tables. Each entitlement table has a specific type of entitlements. For example, there could be a table for user groups, one for roles, and one for access control levels.

Because there is more than one entitlement table, you must create more than one group entity for the connector's mapping file (using that attribute mapping utility). For example, if the Roles and ACL entitlement tables exist in schema, you must create two entities to map these two entitlement tables.

Single Stored Procedure

A stored procedure is created in the database, which will be called by the connector for all provisioning operations. When mapping attributes (such as using the attribute mapping utility), map all of the Select Identity user attributes to the arguments of the stored procedure. When the connector provisions users, the stored procedure is called with the argument values as attributes.

Multiple Stored Procedures

A separate stored procedure is created in the database for each of the user provisioning operations, such as to add a user, modify a user, delete a user, get a user, enable a user, disable a user, reset a password, and so on. When mapping attributes (such as using the attribute mapping utility), map all of the Select Identity operations to the stored procedures. Then, map all of the Select Identity user attributes to the arguments of the stored procedures. When the connector provisions users, the stored procedures are called with the argument values as attributes.

A mapping file for this scenario must map the parameters of the stored procedures and it must specify the attribute level operations (Define Operation) for each parameter. For example, the `UserID` parameter of the `createUser` stored procedure will have “create” in its defined operations because the parameter should be used only for the Add User operation.

Stored Procedure for Attributes

Certain attribute information must be encrypted before being stored in the database. The connector passes data to the stored procedure that it calls. You can create one or more stored procedures to be called for encryption of attribute information. Then, map the attributes to the stored procedures.

Tables and Stored Procedures

There may be scenarios in which a user is provisioned using a combination of updating table and invoking stored procedures in the database. For example, a user could be added to tables but a stored procedure is invoked to enable or disable the user. Review the table and stored procedure scenarios above to understand the necessary configuration.

D Auditing XML and Client Sample

HP OpenView Select Identity can pass event auditing data to third-party auditing tools (such as HP OpenView Select Audit) as an XML stream. An extensible schema definition (XSD) and a sample audit client are packaged with Select Identity.

The audit client is an example program that illustrates how to connect to an application server via JMS and subscribe to the audit XML stream.

The audit XML schema provided can be used to develop an application that interprets the Audit XML stream and presents it in a user-readable format such as a report. The XML stream can be processed into a database, for example, and analyzed using a reporting tool.

The XML stream can also be translated into a Java object hierarchy, which must be undertaken using a third party tool. Appropriate development tools are available from several sources, such as Sun Microsystems and Apache.

This appendix describes how to run the audit client and provides details of the sequences, types, and elements that are defined in the Select Identity audit XSD.

Processing the Audit XML Stream into a Database

The steps below provide a brief high-level outline of the process that must be used to translate the Audit XML stream into a format that can be used to build reports in a database:

- 1 Determine in advance how the database schema and tables will be set up and create these so that they are compatible with the converted Java objects.
- 2 Select a third-party tool to map XML to a java object hierarchy and insert the result into the database.
- 3 Map the java objects to the database schema.

Using the Audit Client

The audit client displays the audit stream in a terminal window in real-time. It is located in the `/utilities/auditclient` subdirectory.

The audit client consists of the component files listed in the table below:

Table 14 Audit Client Files

File or Directory	Purpose
/doc	Contains API documentation in HTML format.
runclient.bat	The batch file used to run the audit client
readme.txt	Release notes and other information
auditbroadcastlistener.java	Sample code demonstrating how to connect to the application server and subscribe to the audit stream
auditbroadcastlistener.class	compiled version of the AuditBroadcastListener class
new-audit.xsd	The XML schema definition for Select Identity audit XML
Various other files	To support your specific application server.

Configuring Connection Properties

The audit client is configured by default to connect to Select Identity on `localhost` to receive the audit stream. If you need to connect to a different host, edit the host name in the `jndi.properties` file before running the audit client.

Running the Audit Client

To run the audit client:

- 1 Copy the entire contents of the following directory from the HP OpenView Select Identity product CD into an appropriate subdirectory in your Select Identity install directory:
`/utilities/auditclient/<audit_client_dir>`
- 2 Ensure that the application server and HP OpenView Select Identity are running.
- 3 Start the audit client using the instructions in the `readme.txt` file appropriate for your application server.

The Select Identity Audit XSD

Select Identity Auditing data is output in XML form as set out in the Audit XML Schema Definition, named `new-audit.xsd`. This file is located on the HP OpenView Select Identity product CD, under `\utilities\auditClient`.

- [Table 15](#) on page 187 provides detailed information about each individual element, since some can belong to more than one complex type.
- [Table 16](#) on page 193 lists the possible event types, which indicate the action that occurred in an audit event.

Table 15 Element Definitions

Element Name	Type	Constraints	Definition
(ConfigChange) Type	Integer		The type of configuration change: TYPE_RESOURCE = 1 TYPE_ADMIN_ROLE = 4 TYPE_EXT_CALL = 5 TYPE_SERVICE = 6 TYPE_SERVICE_CTX = 7 TYPE_SERVICE_ROLE = 8 TYPE_SERVICE_VIEW = 9 TYPE_ATTRIBUTE = 10 TYPE_WORKFLOW = 11 TYPE_RULE = 12 TYPE_NOTIFICATION = 13 TYPE_CONNECTOR = 14
add	PropertyValueSeq	Optional	A property that was added
adminID	String	Required, once per event	The administrator account ID requesting the operation
adminName	String	Optional, once per event	The name of the administrator requesting the operation
adminRole	String	Required, once per event	The role name that authorized the operation
attrChangeData	OVSIAuditAttr ChangeDataSeq	Optional	Affected user attributes/ properties
attrId	Integer	Required, once per event	Attribute ID affected
attrName	String	Required, once per event	Attribute name affected
auditAdminRoles	ConfigChangeSeq	Optional, once per event	Any changes to administrator roles
auditAttrs	ConfigChangeSeq	Optional, once per event	Any attribute changes
auditConnectors	ConfigChangeSeq	Optional, once per event	Any changes to connectors

Table 15 Element Definitions

Element Name	Type	Constraints	Definition
auditExtCalls	ConfigChangeSeq	Optional, once per event	Any changes to external calls
auditNotifications	ConfigChangeSeq	Optional, once per event	Any changes to email templates
auditResourceChanges	ConfigchangeSeq	Optional, once per event	Any resource changes
auditRules	ConfigChangeSeq	Optional, once per event	Any changes to rules
auditServiceChanges	SvcConfigChangeSeq	Optional, once per event	Any service changes
auditTargets	Targetseq	Optional, once per event	The target of the request (e.g. a user, service, or resource)
auditType	Integer	Required, once per event	UNKNOWN_VAL = 0 APPROVAL_VAL = 1 PROVISIONING_VAL = 2 POST_PROVISIONING_VAL = 3 EXTERNAL_CALL_VAL = 4 RESERVED_1_VAL = 5 RESERVED_2_VAL = 6 RESERVED_3_VAL = 7 RESERVED_4_VAL = 8 RESERVED_5_VAL = 9
auditUsers	UserSeq	Optional, once per event	Users affected by the request
auditWorkflows	ConfigChangeSeq	Optional, once per event	Any changes to workflows
causeByRequestId	Integer	Optional, once per event	If an event was triggered by another event, the ID of the triggering event.
ConfigChangeType	Complex		Details of a configuration change

Table 15 Element Definitions

Element Name	Type	Constraints	Definition
ctxVarId	String	Optional, once per event	For service-specific requests, the context variable ID
ctxVarName	String	Optional, once per event	For service-specific requests, the context variable name
ctxVarValue	String	Optional, once per event	For service-specific requests, the context variable value
delete	PropertyValueSeq	Optional	A property that was deleted
entity		Optional, unbounded	When this change is relative to an entity change the entity for this property
entityChanges	EntityChangeSeq	Optional	A collection of changed items. If you change a resource attribute mapping, each attribute of the resource that change will be included in the entity changes for the property "attrs." Each entity change is similar to a property change.
EntityListType	Complex	Optional	A Group of changed entities
fieldId	Integer	Required	ID of configuration item that changed. i.e. ServiceRole ID, Context ID
fieldName	String	Required	The name of the configuration item that changed, i.e. resource name, service name, rule name
key	String		The key name of the entity
membershipId	Integer	Required, once per event	Affected service or resource IDs
membershipName	String	Required	The name of the service or resource in a membership operation
membershipOperation	Integer	Required	Whether the membership was added or deleted: ADD_VAL = 1 DEL_VAL = 2
memberships	MembershipSeq	Optional	Affected memberships

Table 15 Element Definitions

Element Name	Type	Constraints	Definition
membershipType	Integer	Required	Whether the membership was to a service or resource: RESOURCE_VAL = 1 SERVICE_VAL = 2
name	String	Required	Affected user name
name	String	Required	The name of a changed property
newValue	String	Required	The value to which the attribute was changed.
oldValue	String	Optional	The value that was changed.
opType	Integer	Required	A simple type that contains a value indicating the type of operation: Add_VAL = 1 Change_VAL = 2 Delete_VAL = 3
OVSIAuditAttrChange Data	Complex		Attribute change details.
OVSIAuditUser	UserType	Optional, unbounded	Represents the user affected by the event.
parentRequestId	Integer	Optional, once per event	The ID number assigned to a parent request.
primaryId	Integer	Optional	Primary affected user ID (if affected user ID is secondary).
primaryName	String	Optional	Primary affected user name (if affected user name is secondary).
properties	PropertySeq	Required	The Properties that changed as a result of an operation.
property	propertyType	Required	An individual property that changed as a result of an operation.
PropertyType	Complex	Optional	Property change type
requestID	Integer	Optional, once per event	The ID number assigned to a request

Table 15 Element Definitions

Element Name	Type	Constraints	Definition
requestMethod	Integer	Optional, once per event	The method via which the operation was performed, e.g. API, Web, WebService, File: DELEGATED_API = 1 all requests from UI have value. DELEGATED_WEB whether delegated or self service, name is not precise. DELEGATED_WEB = 2 all requests from web service have value. DELEGATED_WEBSERVICE no matter delegated or self service, name is not precise DELEGATED_WEBSERVICE = 3 RECONCILIATION_FILEUPLOAD = 10 RECONCILIATION_WEBSERVICE = 11 BULK_FILEUPLOAD = 12 BULK_WEBSERVICE = 13; BULK_MOVEUSER = 14; RECONCILIATION_POLLING = 15
requestType	Integer	Optional, once per event	The type of request DELEGATED_REGISTRATION = 1 SELF_REGISTRATION = 2 AUTO_DISCOVERY = 3 RECONCILIATION = 4 SYSTEM = 5 BULK_UPLOAD = 6 PROVISION = 7 SERVICECHANGE_RECONCILIATION = 8
sensitiveLevel	Integer	Optional	Indicates a field that is marked "sensitive."
serviceId	Integer	Required	Service ID of the item that changed
serviceName	String	Optional, once per event	For service-specific requests, the service that initiated the request
serviceName	String	Required	Service name of the item that changed

Table 15 Element Definitions

Element Name	Type	Constraints	Definition
status	Integer	Required, once per event	PENDING_VAL = 1 SUCCESS_VAL = 2 FAILURE_VAL = 3 PARTIAL_SUCCESS_VAL = 4 APPROVED_VAL = 5 APPROVED_CHANGES_VAL = 6 REJECTED_VAL = 7
SvcConfigChangeType	Complex		Specialized form of ConfigChangeType
targetId	Integer	Required	The ID of the target
targetName	String	Required	The name of the target
targetType	Integer	Required	The type of target: USER_NORMAL_VAL = 1 USER_PRIMARY_VAL = 2 USER_SECONDARY_VAL = 3 USER_CLUSTER_VAL = 4 RESOURCE_VAL = 5 SERVICE_VAL = 6 SERVICE_CONTEXT_VAL = 7 SERVICE_ROLE_VAL = 8 SERVICE_VIEW_VAL = 9 ATTRIBUTE_VAL = 10 WORKFLOW_VAL = 11 RULE_VAL = 12; NOTIFICATION_VAL = 13 CONNECTOR_VAL = 14 EXTERNAL_CALL_VAL = 15 ADMIN_ROLE_VAL = 16
TargetType	Complex		Audit operation target details
timestamp	Long Integer	Required, once per event	The time at which the event occurred, relative to server time, expressed as the number of milliseconds since January 1st, 1970.
type	Integer	Required	As for ConfigChangeType
userId	Integer	Required	Affected user ID

Table 16 Possible Event Types

Action Type	Action
User Request	ADD_NEW_USER = 1 MODIFY_USER = 2 DELETE_SERVICE_MEMBERSHIP = 3 ENABLE_ALL_SERVICES = 4 DISABLE_ALL_SERVICES = 5 RESET_PASSWORD = 6 COPY_USER = 7 ADD_SERVICE = 8 CHANGE_PASSWORD = 9 FORGET_PASSWORD = 10 ENABLE_SERVICE_MEMBERSHIP = 11 VIEW_SERVICE_MEMBERSHIP = 12 TERMINATE_USER = 13 MANAGE_USER_EXPIRATION = 14 DISABLE_SERVICE_MEMBERSHIP = 15 SECURITY_VIOLATION = 16 MODIFY_PROFILE = 17 PASSWORDEXPIRE_NOT = 18 MOVE_USER = 19 LOGIN = 20 LOGOUT = 21 IMPORT = 22 EXPIRE_PASSWORD = 24 HINTSETUP = 30 DISABLE_TERMINATE = 31 REVERT_MODIFY = 32 REVERT_ADD = 33 REVERT_DELETE = 34 IGNORE_ADD = 35 IGNORE_MODIFY = 36 IGNORE_DELETE = 37
Cluster Operations	CREATE_CLUSTER = 40 MODIFY_CLUSTER = 41 DELETE_CLUSTER = 42 ADD_SECONDARY = 43 REMOVE_SECONDARY = 44
Service Change Reconciliation	SVCCHG_RECON_MODIFY_USER = 51 SVCCHG_RECON_ADD_RESOURCE = 52 SVCCHG_RECON_DELETE_RESOURCE = 53

Table 16 Possible Event Types

Action Type	Action
Resource Reconciliation	RESOURCE_RECONCILIATION_DELETE = 56 RESOURCE_RECONCILIATION_MODIFY = 57 RESOURCE_RECONCILIATION_REPLACE = 58
User Role Delegation	USER_ROLE_DELEGATION_ACTIVATE = 54 USER_ROLE_DELEGATION_DEACTIVATE = 55
Service	SERVICE_CREATE = 2000 SERVICE_DELETE = 2001 SERVICE_MODIFY = 2002 SERVICE_COPY = 2003 SERVICE_SET_ATTR_VALUES = 2004 SERVICE_SET_ATTR_PROPS = 2005 SERVICE_VIEW_CREATE = 2006 SERVICE_VIEW_DELETE = 2007 SERVICE_VIEW_MODIFY = 2008 SERVICE_ROLE_CREATE = 2009 SERVICE_ROLE_DELETE = 2010 SERVICE_CONTEXT_CREATE = 2011 SERVICE_CONTEXT_DELETE = 2012 SERVICE_CONTEXT_MODIFY = 2013 SERVICE_IMPORT = 2014 SERVICE_ROLE_MODIFY = 2015 SERVICE_EXPORT = 2016
Resource	RESOURCE_CREATE = 3000 RESOURCE_DELETE = 3001 RESOURCE_MODIFY = 3002 RESOURCE_VIEW = 3003 RESOURCE_COPY = 3004 RESOURCE_ATTR_VIEW = 3005 RESOURCE_ATTR_MODIFY = 3006 RESOURCE_IMPORT = 3007 RESOURCE_EXPORT = 3008
Attribute	ATTRIBUTE_CREATE = 4000 ATTRIBUTE_DELETE = 4001 ATTRIBUTE_MODIFY = 4002 ATTRIBUTE_VIEW = 4003 ATTRIBUTE_COPY = 4004 ATTRIBUTE_IMPORT = 4005 ATTRIBUTE_EXPORT = 4006

Table 16 Possible Event Types

Action Type	Action
Workflow	WORKFLOW_CREATE = 5000 WORKFLOW_DELETE = 5001 WORKFLOW_MODIFY = 5002 WORKFLOW_VIEW = 5003 WORKFLOW_COPY = 5004 WORKFLOW_IMPORT = 5005 WORKFLOW_EXPORT = 5006
External Call	EXT_CALL_CREATE = 6000 EXT_CALL_DELETE = 6001 EXT_CALL_MODIFY = 6002 EXT_CALL_VIEW = 6003 EXT_CALL_COPY = 6004
Notification	NOTIFICATION_CREATE = 7000 NOTIFICATION_DELETE = 7001 NOTIFICATION_MODIFY = 7002 NOTIFICATION_VIEW = 7003 NOTIFICATION_COPY = 7004 NOTIFICATION_IMPORT = 7005 NOTIFICATION_EXPORT = 7006
Connectors	CONNECTOR_CREATE = 8000 CONNECTOR_DELETE = 8001 CONNECTOR_MODIFY = 8002
Rules	RULE_CREATE = 9000 RULE_DELETE = 9001 RULE_MODIFY = 9002
Admin Roles	ADMINROLE_CREATE = 10000 ADMINROLE_DELETE = 10001 ADMINROLE_MODIFY = 10002

E External Calls

Select Identity workflow processes and attributes support the ability to perform actions on external systems. This functionality, called **external calls**, enables integration of access approval processes with other business processes and systems. External system calls can also constrain or verify the value of identity attributes.

External calls can be used to perform the following types of tasks:

- Generate a user ID or password.
- Provide a list of possible attribute values.
- Validate the value of an attribute.
- Verify the value of an attribute.
- Query an external system for a list of approvers.
- Perform a workflow task.
- Retrieve a certificate from an external system.
- Filter and convert incoming extended SPML requests for reconciliation

The Select Identity external call and workflow APIs define a Java-based interface for creating external calls. Although the Select Identity-facing portion of the interface must be Java, it can be a “wrapper” for a program written in any language.

You must code the classes called by external calls using the external call API and workflow API. After you create the Java file(s) that comprise an external call, you can register it with Select Identity through the external calls capability.

This section provides a quick overview of external calls.

Refer to the *HP OpenView Select Identity External Calls Developer Guide* for complete information about external calls.

Default External Calls

Select Identity provides default external calls to enable you to interact with external systems. Each external call is within one of the following call types:

- Approver selection — searches an external system for a list of users who can approve provisioning requests during a workflow
- Attribute value generation — generates the name or ID of a user, the user's password, and any other attribute, such as the user's company, department and so on
- Attribute value constraint — provides a list of possible values for an attribute
- Attribute value validation — validates the value of an attribute
- Attribute value verification — verifies the value of an attribute
- CertificateManagementFunction - retrieves a certificate from an external system
- SPML Request Filter - used to process an SPML request
- Workflow action — performs a task as part of a workflow, enabling you to integrate approval processes with external processes and systems

Most external calls have predefined parameters you can modify. They are described in the *HP OpenView Select Identity External Calls Developer Guide*.

Creating an External Call For Workflow Templates

Select Identity also allows you to create your own external calls. To create an external call, you must write the code that issues a request to the external system. See the *HP OpenView Select Identity External Calls Developer Guide* for complete information regarding the creation of external calls.

When the external call returns information, it must return data that is valid in Select Identity. For example, for Approval Lookups, the external call must return a valid user ID existing in Select Identity. Therefore, when you create the external call, provide a way for it to map the returned user ID to the Select Identity user ID.



If the external system cannot send the Select Identity user ID, the workflow process terminates and an error is sent.

After you create the call, copy the Class files or jar files to a directory on the Select Identity server. Copy the files to a directory that is accessible by the user account running the Application Server. In the case of a cluster, copy the files to a shared folder so that each server in the cluster will access the files using the same path.

Creating an External Call for Attributes

You can assign external functions to different attributes for the following purposes:

- Value, which defines the acceptable values for an attribute
- Constraint, which constrains the attribute value to a particular format or requirement
- Validation, which calls an external program to validate the value of the attribute
- Verification, which verifies that the value is what was previously saved. This is used to verify passwords
- Generation, which automatically generates a value for an attribute

These functions are created and made available in the Select Identity system through the **External Call** pages. For examples, see the *HP OpenView Select Identity External Calls Developer Guide*.

Deploying an External Call

After you create the files and copy them to the Select Identity server, you need to register the external call. The **External Calls** section in Service Studio is where you register and manage the external calls used by Select Identity.

To register an external call, you enter in the basic information (Name, Description, Class Name, Class Path and Call Type) on the first page, and then enter in any parameters used by the external call.

If the external is a used by an attribute, the parameter values you enter when you register it will be used as default values, which can be overridden.

Glossary

actions

Actions are associated with workflow activities. Actions invoke functions provided by the Select Identity workflow engine or external applications. For example, actions can log information to a file, set a variable to be used later in the workflow, call an external process to provision a user in Select Identity, or store data in a database.

activities

An activity represents a step in a process represented by a workflow template. Activities are the core components of workflow templates; the actions defined in activities do the work necessary to provision users. An activity can contain actions that, for example, set workflow variables, track approvals, start a sub-workflow, send email, and call external applications.

AD

Active Directory

administrative (admin) service

A service used by the Select Identity System Administrator to add approvers. A separate service for administrators eliminates the need to add administrators each time for every service for which they are responsible. Administrators can thus manage user approval requests from numerous services.

agent

A connector that allows for reverse data flow from the SI data store to the enterprise resource. This permits bidirectional replication.

agent-based connector

A 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 connector

A one-way connectors. Connectors reside in the Select Identity server and perform communication brokering with the resource.

application invocation

Use the Application Invocation workflow action to call a Select Identity application. Select Identity provides many applications that you can use. You can also develop your own customized applications within Select Identity.

approval process

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

approver

A Select Identity administrator who approves user management requests. Approvers must have an admin role that provides approver-level privileges.

asynchronous invocation

An activity or action property that allows the workflow to continue processing a request before an invoked application completes its operation. This property can help reduce request processing bottlenecks.

attribute

A data field, containing a value, that helps define an object in Select Identity, such as a service or an identity profile. For example, an attribute could be “department” with possible values of “IT,” “sales,” or “support.”

attribute mapping

The process of associating the name of a resource attribute with that of its corresponding Select Identity attribute. This facilitates and maintains the integrity of data exchange between the two.

audit

A record of events, transactions, configuration changes, and other data about the use, operation, and maintenance of a system.

audit report

A report that presents audit data so that it is organized and readable.

authentication

Verification of the credentials associated with an identity, such as a password and user ID combination, to prevent improper access.

authoritative resource

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

authorization

Real-time enforcement of an identity’s entitlements. Authentication is a prerequisite for authorization.

auto discovery

The process of adding user accounts to Select Identity for a specified Service by importing them from a data file.

block

A set of related activities within a workflow. Blocks have two purposes, to define information to be shared 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. Think of a block as a subprocess within a workflow.

block form

A form specifically associated with a block in a workflow. For example, a workflow could have different forms for each of its approval blocks, each form showing a different set of user attributes.

block view

A view associated with a specific block in a workflow. For example, a workflow could have several views for each of its approval blocks, each showing a different set of user attributes.

business process engine

A system component that serves up all Workflow, Reconciliation, Policy, Forms, Tiered Access, Audit and Report features.

business relationship

see the definition for service role.

business service

A product, facility, or essential business process offered or used by an organization to support day-to-day operations. Examples include online banking services, customer support process, and IT infrastructure offerings such as email, calendaring, and network access. See also: service

Business Service Identity Management (BSIM)

A new, dynamic, and scalable approach to identity management within and between enterprises. BSIM automates the process of provisioning and managing user accounts and access privileges across platforms, applications, and corporate boundaries.

challenge and response

A method of supplying alternate authentication credentials, typically used when a password is forgotten. Select Identity challenges the user with a question and, if the answer is correct, resets the password to a random value and sends email to the user.

challenge question

A question designed to elicit very specific information about the person seeking system access. These are most commonly used as a secure fail-safe when a password is lost. Examples are "What is your mother's maiden name" or "Which city were you born in?" Standard challenge questions are preset questions that all users must answer to reset their password. This is usually written by the Select Identity System Administrator as part of setting a challenge/response policy. A preset question written by the user and stored with its answer in that user's profile. It must be answered by the user to reset his or her password. Users write their personal challenge question or questions at initial log on after they receive a user name and

password. Typical question could include “What year did I graduate from college?” or “What is my pet’s name?”

class path

A setting that specifies the directory location of important system files.

cluster

A group of servers that function as a single entity, for example by operating as a jBoss AS, BEA WebLogic, or IBM WebSphere Web application server. This term is also used to refer to a group of user accounts, known specifically as a user cluster.

composite service

A mechanism for grouping services so that they can be accessed as a unit. For example, users who register for a composite service actually have access to multiple services as a result of registering.

Concero sys admin

A special-purpose administrative role that confers the highest level of entitlements and access.

Configuration Approver

A special-purpose administrative role that confers the privilege of approving configuration management changes in systems where this feature is enabled. Only users who have this role can approve changes to the system's configuration.

configuration management

The configuration management feature establishes an approval workflow for Select Identity configuration changes.

configuration report

Configuration reports provide current system information for user, administrator, and service management activities.

configurations

The configurations capability enables you to import and export Select Identity settings and configurations, such as workflows, resources, services and attributes. This is useful when moving from a test to a production environment.

connectors

Connectors are programs that enable various databases and applications to be accessed by Java application servers that run on the J2EE platform from Sun Microsystems. They enable Select Identity to access an enterprise application and communicate with the system resources that contain your identity profile information. SI comes with several predefined connectors to support data access with backend data stores. For example, a J2EE connector is included that communicates with the Select Identity system resources that contain your identity profile information.

context

A logical grouping of users who are able to access a service.

context attribute

A common attribute that groups users so that they can access a service through a specific service role. For example, an East context groups users with an East attribute so that they can assume the East service role in the XYZ Service.

context engine

A system component that retrieves data according to a service/users context definition.

credential

Information that is used for validation of a person's identity for security and access control purposes. Examples are a user name, password, challenge/response questions, digital certificates, and biometrics.

data file

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

data services template

A specification that provides protocols for the query and modification of data attributes related to a Principal, and exposed by a data service. The protocols are also provided for subscribing to notification related to those attributes and sending and receiving those notifications. Additionally, some guidelines, common XML attributes and data types are defined for data services.

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 to avoid having to build new workflow templates each time. You can use these templates as they are, or you can copy them, rename them, and modify them as you wish. See also workflow templates.

delegated administration

user identity management functions performed by an administrator on behalf of end users. See also self-service.

delegated registration

Account registration performed by an administrator on behalf of another person.

deployment

To install and start software, hardware, capabilities, or services so that they begin to function as intended in the business environment.

disable

To put something out of use without deleting it, usually on a temporary basis. User accounts and services can be disabled in Select Identity, for example.

EAR file

Enterprise Archive; a compressed file format for storing application packages such as Select Identity.

enable

To reinstate something that has been previously disabled, such as a user account, or to put a feature or setting into operation.

end user

A role assigned by default to every user in Select Identity. The end user role allows access to the Self Service pages, but conveys no administrative rights.

entitlement

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

event handler

Notification templates or workflow templates associated with a particular system event. Notification templates are notification event handlers; workflow templates are request event handlers.

event manager

A system that provides for an event handler interface to process all system events such as sending out e-mail notifications or executing workflows associated with a particular system event. Notification templates are notification event handlers. Workflow templates are request event handlers.

export

To format and store data for use by other applications or systems.

external call

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

fixed entitlement

An entitlement automatically granted to a certain users as determined by the service role and context associated with their identity.

form

An electronic document used to capture information from end users. Forms are used by Select Identity for information capture and system operation in many business processes. Most Select Identity forms consist of a subset of service attribute fields in a presentation view that allows certain logical groups users to enter values. For example, you could define a form for administrators to add users, a form for administrators to grant user entitlements, another form for users to modify their own profile, and so on.

function

A grouping of Select Identity permissions.

identity

A set of data relating to a specific individual within a system, including their personal details, contact information, and access privileges to various resources and services. Some identities are special-purpose, such as a system administrator.

identity management (IdM)

Identity management is a set of processes, tools, and agreements among organizations and individuals. It enables people, systems and services to access resources to achieve business objectives.

identity provider (IdP)

An identity provider or IDP is a website that authenticates a user before the user is sent off to a federated website. It is possible for a website to function as an IDP as well as SP.

import

To read, reformat, and store data from another application or system.

JCA

Java Connection Architecture

JDBC

Java Database Connectivity

JDK

Java Developer Kit

JMS

Java Messaging Service

JNDI

Java Naming Directory Interface

key rollover

Scheduled, automatic, and periodic changes of encryption keys. This is a security measure of the same type as that requiring users to change passwords regularly.

keystore

A database file that contains both public and private keys. Public keys are stored as signer certificates while private keys are stored as personal certificates. The keys are used for a variety of purposes, including authentication and data integrity.

LDAP

Lightweight Directory Access Protocol

LDIF

Lightweight Directory Access Format

Liberty Identity Federation Framework (ID-FF)

Popular open standard federation protocol developed by the Liberty Alliance Project, an alliance of more than 150 companies, nonprofit and government organizations from around the globe. The consortium is committed to developing an open standard for federated network identity.

Lightweight Directory Access Protocol (LDAP)

LDAP is a software protocol for enabling anyone to locate organizations, individuals, and other resources such as files and devices in a network. An LDAP directory can be distributed among many servers. LDIF is used to synchronize each LDAP directory.

Lightweight Directory Interchange Format (LDIF)

An ASCII file format used to exchange data and enable the synchronization of that data between Lightweight Directory Access Protocol (LDAP) servers called Directory System Agents (DSAs).

logging.properties file

A text file that defines how Select Identity logs messages and exceptions.

name-value pair

A name-value pair is combination of an attribute identifier (field name) 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.

notification

A message sent when a system event occurs, typically via email. You configure and set up Notifications in Select Identity using a template that controls the information sent in the notification message.

notification template

A notification template defines the format and content of standard e-mail messages automatically sent by Select Identity when certain types of system event occur, such as a user's account request approval, an account deletion, or a password reset.

OASIS

Organization for the Advancement of Structured Information Standards. A not-for-profit, international consortium that drives the development, convergence, and adoption of e-business standards. The consortium produces Web services standards for security, e-business, and standardization efforts in the public sector and for application-specific markets.

optional entitlement

An entitlement available to certain users as determined by the service role and context associated with their identity. Users have the option of choosing the entitlement.

password reset

Setting a password to a system-generated value.

password validation function

Function that validates the password value against a predefined set of parameters; for example 6-12 alphanumeric characters, at least two numerals, and so forth.

password verification function

Function that verifies the password against a list of authorized passwords.

permission

A permission that allows a user to perform an administrative task within Select Identity.

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.

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.

pre-defined variables

Variables for administrator names, user names, and email addresses. These variables enable the system to supply the appropriate information based on the action and the user performing the action.

primary user

The user ID in a user cluster that serves as the Select Identity login name. All other user accounts in the cluster are associated with it as secondary accounts.

profile

A group of descriptive attributes associated with an identity, such as name, address, title, company, or cost center.

profile attributes

Descriptive attributes associated with an identity, such as name, address, title, company, or cost center.

properties

A name-value pair where the value is a string. Properties define constant data when the template is created. Property values do not change at runtime. A global property is shared by all the activities within a workflow instance. The first time you set a property, you initialize its type. Specified properties can be read by external applications using the Workflow API provided by Select Identity. They can also be referenced by a report template to show relevant information in a status report. Some property names are defined by the workflow engine. Use these properties when defining activities and blocks. When you create a workflow template, you must assign values to properties. These property values instruct the workflow how to

operate. For example, if you assign a value of three to the joinCount property, the workflow waits for three approvers to join the workflow before it exits the approval block.

reconciliation

The process by which Select Identity accounts are synchronized with a system resource. Account data is added to the Select Identity system from 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.

request

An event within Select Identity initiating the addition, modification, or removal of a user account.

request event

Whenever a user account is added, modified or removed, it is registered within Select Identity as a request event.

resource

Any single application or information repository that is part of your Select Identity BSIM solution. 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 is usually organized by job function.

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.

secondary user

An account other than the primary user in a user cluster.

self-service

The ability to securely allow end users to manage identity and service access on their own behalf.

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.

service attribute properties

Settings that determine how fields are displayed in forms.

service attribute values

Restrict the values that a user or approver can select from in a form.

service form

A restricted form of a service that is valid for a group of users. Forms enable you to define a subset of service registration fields, change field names, reorder fields, and mask field values for specific users.

service role

A Select Identity abstraction that defines how a logical grouping of users will access a subset of a Select Identity service's entitlements. For example XYZ Service could have three business relationships: East, Central, and West. These business relationships can be subdivided as well, for example the business relationship West could contain two more granular business relationships: Northwest and Southwest. A service can contain an unlimited number of levels of business relationships.

SHA

Secure Hash Algorithm

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.

SOAP

Simple Object Access Protocol

SPML

Service Provisioning Markup Language

SPML data file

A server-parsed XML file used to add and provision accounts in Select Identity. For reconciliation, SPML data files can be generated from both non-authoritative and authoritative resources. Typically, one SPML file is generated for each resource. The SPML file contains changes for a specific period to user account information, such as additions, deletions, or changes. The files are uploaded into Select Identity, and the user account information in Select Identity is updated during reconciliation.

SSO

single sign-on

terminate

Removing a user's account from Select Identity, so that it no longer exists.

transition

A transition provides a link from one activity to another. There are two kinds of transitions—conditional and unconditional. Using an unconditional transition to link two activities means that the second activity is always executed after the first. With a conditional transition, a

certain condition must first be met before the next activity is executed. For example, you can define a transition that only allows the workflow to progress if at least two administrators approve a request.

TruAccess.properties file

A text file that contains numerous Select Identity configuration settings that you can customize.

URI

Uniform Resource Identifier

user cluster

A group of user IDs consisting of one primary user account and multiple associated secondary accounts. This is the mechanism used to enable a single person whose identity is managed in Select Identity to have several user accounts on multiple resources.

user import

The process of adding user accounts for a specified Select Identity service by copying them into the database from a data file.

Wait activity

The Wait Activity check box is selected if the activity you are creating requires an action to occur before moving forward to the next activity (e.g. approver approves/rejects an account request).

wait instance

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

WAR file

Web Archive file; a compressed format for packaging multiple files. Has the extension .war and is used by servlets.

Web application server

A computer or group of computers configured to provide infrastructure for Internet-based data and communication transactions.

Web Services

An XML-based request framework using the Service Provisioning Markup Language (SPML) to provide customizable user management functions in Select Identity.

Web Services Definition Language (WSDL)

An XML format for describing network services as a set of endpoints operating on messages containing either document-oriented or procedure-oriented information. The operations and messages are described abstractly, and then bound to a concrete network protocol and message format to define an endpoint. Related concrete endpoints are combined into abstract endpoints (services). WSDL is extensible to allow description of endpoints and their messages regardless of what message formats or network protocols are used to communicate.

WfMC

Workflow Management Commission

workflow

The process by which requests are completed in Select Identity, including the various levels of approval needed for different types of request. Workflows are depicted in the form of a process flow, defined in a workflow template created in Workflow Studio. Requests can be tracked using the workflow to ascertain their current state of completion in detail.

Workflow Approver Role

A default Administrative role that grants permission to approve user provisioning workflow operations.

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 studio

The functionality that enables you to create and manage workflow templates.

Workflow Studio Editor

The Select Identity capability with a special graphical user interface that enables you to create and manage workflow templates.

WSDL

Web Services Definition Language

XML Processing Description Language (XPDL)

When you save a workflow template, it is saved in the Select Identity repository as an Extensible Markup Language (XML) file. Its format is the XML Processing Description Language (XPDL) as defined by the Workflow Management Coalition (WfMC).

XPDL

XML Processing Description Language

Index

Symbols

`$_instId`, 45
`$ConfigObjectName`, 46
`$ConfigObjectType`, 46
`$RequestActionName`, 46
`$RequestId`, 45

A

account
 primary, 69
 secondary, 69
account consolidation
 consolidation
 of a user's multiple accounts, 106
action, 52
 understanding, 52
addRequest, 65, 75
administrative action, concepts, 52
administrative function, concepts, 52
administrative role
 actions, 52
 capabilities and actions, 52
 creating, 53
 creating a, 53
 default roles, 53
 delegate, 54, 58
 delegation of, 54
 delete, 54
 deleting, 54
 designing, 24
 granting permissions, 53
 import, 89
 managing, 53
application server, 23
application server properties
 and reconciliation, 116
approver role, 53
architecture and design, 22
assumptions, about readers, 13

attribute, 28
 adding and mapping, 37
 creating external call for, 199
 encrypted, and reconciliation, 95
 import, 89
 mapping file, 39
 mapping utility, 163
 operational, 76
 profile, 58
 specifying in SPML, and reconciliation, 115
audience, for this book, 13
audit, 25
 event data, 185
 report, 54
 XSD, element definitions, 187
 XSD, event types, 193
audit client, 185
 jndi.properties file, 186
 using, 185
authoritative resource
 and reconciliation, 96
 See Also resource

B

backup, 25
batchRequest, 65, 75
bulk
 add, 139
 create SMPL containing users and attributes, 75
 dependencies, 74
 managing, 79
 procedure overview, 75
 scheduling, 79
 SPML, 74
bulk add
 concepts, 62
 multiple-user-id accounts, update, 81
 See Also bulk
bulk move, 79
 See Also bulk
bulk operations
 See bulk

C

- capability, 52
- challenge/response, 59
- change password, 59
- change password question, 59
- concepts
 - administrative action, 52
 - administrative function, 52
 - bulk add, 62
 - multiple-user-id accounts, 68
 - my identity, 57
 - reconciliation, 95
 - secure object migration, 91
 - test and deploy, 83
 - user import, 62
- concerto sys admin, 53
- configuration
 - edit export XML, 88
 - export, 88
 - import, 88, 89
 - import, recommendation, 90
 - import dependencies, 89
 - report, 54
 - secure replication, 91
 - troubleshooting replication, 92
- configuration approval
 - import setup, 90
- configuration approver, 53
- configuration replication, 87
- configuration report
 - report
 - configuration, 55
- connector, 23
 - import, 89
 - installing a, 37
 - one-way, 35
 - two-way, 36
- consolidation
 - See Also multiple-user-ID

D

- data
 - user, populating, 74
 - user, validating, 63
- database, 23
 - populating, 63, 74
- data processing, 62
- decryption, 91
- default administrative roles, 53

- delegate
 - administrative role, 54
 - administrative roles, 58
 - recover requests, 138
- deployment, 87
 - checklist, 93
 - final system check, 93
 - preparation, 85
 - users, set up, 92
- detailed discovery, 20
- disaster recovery, 25
 - testing, 85
- documentation map, 15
- document type definition
 - See DTD
- DSML, 64
- DTD, 143

E

- encryption, 91
- end-user role, 53
- entitlement, 27
 - add, in SPML, 115
 - caching policies, 41
 - transfer on move, 80
 - when user moves, 80
- event data, auditing, 185
- exclusion rule, 147
- export
 - secure, 91
 - troubleshooting, 92
- export configuration, 88
- extensible schema definition
 - XSD, 185
- external call, 28, 197
 - default, 198
 - deploying, 199
 - import, 89
 - uses, 197

F

- failover, 87
- filter, 104
- final system check, 93
- firewall, 87
- form, 28

function
understanding, 52

G

generator
See SPML

I

identity
multiple, for one user, 68

import
configuration, 88, 89
configuration, recommendation, 90
multiple-user-id account, 71
secure, 91
troubleshooting, 92
user, 26
user, scheduling, 67
users, 92
See Also user import

integration testing, 84

J

JMS
and audit client, 185

job results
non-authoritative results, 120
viewing, 79

K

keyFields, 65, 66
keystore, 92

L

LDIF, 63, 64
load balancing, 87
load testing, 84

M

mapping file, 39
mapping utility, 163
migration
secure, 91
multiple identity accounts
See multiple-user-ID

multiple-user-id
account, example, 70
concepts, 68
creating an account, 70
import, account, 71
SPML, 71
update, account, 71
update with bulk add, 81
uses, 69

my identity
concepts, 57

N

network architecture, 24
non-authoritative resource
reconciliation results, 120
See Also resource
notification, 28
user, 46
variables, 46
notification template
import, 89

O

operationalAttributes, 65, 75

P

password
change, 59
password question
change, 59
performance
reconciliation, tips, 134
performance testing, 84
permissions
administrative role, 53
populating Select Identity, 62
primary account, 69
primaryAcctKey, 71, 81
primaryAcctValue, 71, 81
processes
defining, 27
profile attribute, 58
properties configuration files, 157

R

- reconciliation, 62, 63, 104, 117
 - and TruAccess.properties settings, 116
 - application server properties, 116
 - attribute-level policy, 103
 - concepts, 95
 - creating SPML file, 109
 - non-authoritative results, 120
 - performance and configuration tips, 134
 - procedure overview, 96
 - recovering requests, 139
 - resource-level policy, 103
 - resource names, 110
 - rule, creating, 142
 - rule, tips, 142
 - rule, troubleshooting, 146
 - rules, 126, 141
 - separator string, 110
 - service, 82
 - service membership requirements, 96
 - SPML, 110
 - SPML examples, 112
 - tips, SPML, 111
 - troubleshooting, common, 122
 - troubleshooting, reports, 132
 - troubleshooting, request processing, 124
 - troubleshooting job submission and execution, 123
 - viewing task status, 108
 - See Also service reconciliation
- reconciliation tips, 111
- recovering requests, 140
- recover requests, 138
- release planning, 25
- replication
 - configuration, secure, 91
 - of a configuration, 87
- report
 - audit, 54
 - audit reports, 54
 - configuration, 54
 - parameters, 57
 - scheduling, 56
 - troubleshooting, reconciliation, 132
- reporting, 25
- request
 - locating, 138
 - recovering, 138
- request filter
 - SPML, 64
- requestID, 75

- request instance report
 - import, 89
- request status, 58
- resource, 28
 - accounts, viewing, 59
 - authoritative, and reconciliation, 96
 - import, 89
 - removing, 41
- role
 - approver, 53
 - concero sys admin, 53
 - configuration approver, 53
 - creating, 53
 - end-user, 53
 - permission, 58
- roles
 - administrative role, 54
 - approver, 53
 - default, 53
 - permissions, 53
 - system administrator, 53
- rule
 - creating, reconciliation, 142
 - DTD for rule definition, 143
 - exclusion, 147
 - import, 89
 - managing rules, 146
 - sample, 148, 149, 153
 - standards, 149
 - troubleshooting, 146
- rules
 - creating rules, 141
 - reconciliation, 98

S

- SampleXML directory, 64
- scheduled report, 56
- schedule services assignment, 68
- schema
 - XML, 185
- secondary account, 69
- secure migration
 - See secure object migration
- secure object migration, 92
 - concepts, 91
- security framework, 92
- Select Audit
 - and audit data stream, 185

- Select Identity
 - architecture and design, 22
 - audit client, 185
 - default external calls, 198
 - deployment, 87
 - deployment, populating, 92
 - designing administrative roles and services, 24
 - documentation map, 15
 - final system check, 93
 - methodology, test and deploy, 83
 - populate, 26, 74
 - release planning, 25
 - requirements, 20
 - testing, 84
 - three-phase methodology, 17, 31, 83
 - training, 29
 - user data, populating with, 62
 - web service, 36
 - See Also three-phase methodology
 - self-registration, 59
 - configuring, 59
 - context value, 60
 - form, configuring, 59
 - pre-defined context, 60
 - setting up, 59
 - URL, setting, 60
 - self-service, 57
 - recovering requests, 138
 - self-subscribe, 59
 - server
 - abnormal shutdown, 137, 138
 - server management, 137
 - server properties
 - and reconciliation, 116
 - server worklist, 137
 - service
 - assignment, scheduling, 68
 - assignments, 67
 - creating a, 49
 - defining a, 27, 28
 - designing, 24
 - import, 89
 - membership, viewing, 59
 - membership requirements for assignment, 68
 - multiple services, 29
 - reconciliation, 82
 - reusable components, 28
 - service assignments
 - See service
 - ServiceNameMap, 149
 - service provisioning markup language
 - See SPML
 - service reconciliation, 140
 - service role, 28
 - setting up self-registration, 59
 - signature, 91
 - simple object access protocol
 - See SOAP
 - SOAP, 36, 113
 - solution
 - architecture and design, 22
 - deployment, 87
 - documents, 85
 - final system check, 93
 - release planning, 25
 - requirements, 20
 - reusable components, 28
 - testing, 84
 - training, 86
 - SPML, 26, 36, 64, 111, 155
 - adding users to services with common attributes, 76
 - add users to services with specified entitlements, 78
 - bulk, 74
 - containing entitlements, creating, 66
 - create, containing users and attributes for bulk, 75
 - creating for reconciliation, 109
 - creating with entitlements, 115
 - example file, without taUserName, 66
 - examples, reconciliation, 112
 - for bulk operations, 74
 - for users and attributes, creating, 64
 - generator, 64, 155
 - multiple-user-id, 71
 - request filter, 64
 - specifying attributes, 115
 - upload, 79
 - writing, for reconciliation, 110
 - system administrator role, 53
- ## T
- taResourceKey, 65, 66
 - taUserName, 65
 - test and deploy
 - concepts, 83
 - testing, 84
 - disaster recovery, 85
 - integration, 84
 - load, 84
 - performance, 84
 - user acceptance, 84

- three-phase methodology, 17, 31, 83
 - design and planning, 19
 - project definition, 18
 - project initiation, 18
- training
 - Select Identity, 29
 - solution, 86
- transfer on move
 - See entitlement
- troubleshooting, 92
 - migration, 92
 - reconciliation rules, 146
 - See Also specific component
- TruAccess.properties
 - reconciliation-related settings, 116
- TruAccess properties, 117
- truststore, 92

U

- updatet
 - multiple-user-id account, 71
- upload data files, 79
- user
 - add to services with common attributes, SPML, 76
 - add to services with specified entitlements, SPML, 78
 - set up, 92
 - status dependencies, in rules, 141
- user acceptance testing, 84
- user account reconciliation
 - See reconciliation
- user data, 62
 - convert to SPML, 26
 - populating, 74
- user import, 26, 92
 - concepts, 62
 - multiple-user-id account, 71
 - performance, 67
 - scheduling, 67
 - tips, 26
 - See Also import
- user polling, 104

V

- variable, 28
 - notification, 46

- view
 - job results, bulk, 79
 - reconciliation task status, 108
- view resource accounts, 59

W

- web service, 36
- workflow, 28
 - import, 90
 - template, 43
 - template, creating external call for, 198
- workflow application definition
 - import, 89
- Workflow Studio, 42
- workflow template
 - import, 90
- worklist, 137

X

- XML
 - audit stream, 185
 - edit, export configuration, 88
- XSD, 185
- XSLT, 64