Contents

Preface ............................................................................................................................................................ ix
Audience ............................................................................................................................................................ ix
Documentation Accessibility ........................................................................................................................... ix
Related Documents ........................................................................................................................................... ix
Documentation Updates .................................................................................................................................... ix
Conventions ....................................................................................................................................................... x

What’s New in Oracle Identity Manager Connector for Webservices? ......................................................... xi
Software Updates ............................................................................................................................................... xi
Documentation-Specific Updates .................................................................................................................... xi

1 About the Connector
1.1 Certified Components ............................................................................................................................... 1-2
1.2 Certified Languages .................................................................................................................................... 1-2
1.3 Connector Architecture ............................................................................................................................ 1-3
1.4 Features of the Connector ........................................................................................................................ 1-4
1.4.1 Support for Configuring the Connector for a New Target System ....................................................... 1-5
1.4.2 Support for Securing the Connector ........................................................................................................ 1-5
1.4.3 Support for Multiple Instances and Multiple Versions of Target Systems ........................................... 1-5
1.4.4 Support for Both Target Resource and Trusted Source Reconciliation ............................................. 1-5
1.4.5 Support for Both Full and Incremental Reconciliation ........................................................................ 1-5
1.4.6 Support for Limited Reconciliation ...................................................................................................... 1-5
1.4.7 Support for Batched Reconciliation ....................................................................................................... 1-5
1.4.8 Validation of Data .................................................................................................................................... 1-6
1.4.9 Transformation of Data ........................................................................................................................... 1-6
1.4.10 Support for Resource Exclusion Lists ................................................................................................. 1-6
1.5 Lookup Definitions Used During Connector Operations ......................................................................... 1-6
1.5.1 Lookup Definitions Synchronized with the Target System ................................................................. 1-6
1.5.2 Preconfigured Lookup Definitions ....................................................................................................... 1-6
1.5.2.1 Configuration Lookup Definitions .................................................................................................... 1-7
1.5.2.2 Lookup.ACME.UM.Configuration .................................................................................................... 1-7
1.5.2.3 Lookup.ACME.UM.Configuration.Trusted ....................................................................................... 1-8
1.5.2.4 Lookup.ACME.UM.ProvAttrMap ..................................................................................................... 1-8
1.5.2.5 Lookup.ACME.UM.ReconAttrMap ................................................................................................. 1-8
1.5.2.6 Lookup.ACME.UM.ReconAttrMap.Trusted ....................................................................................... 1-9
### 2 Preinstallation Steps

2.1 Prerequisites ........................................................................................................... 2-1
2.2 Building the Connector Bundle ............................................................................... 2-2
2.3 Creating a SOA Composite for the Target Webservice ............................................ 2-4
2.3.1 Configuring the Partner Link ............................................................................... 2-5
2.3.2 Configuring the Create Operation ....................................................................... 2-7
2.3.3 Configuring the Create Operation for SPML ....................................................... 2-10
2.3.4 Configuring the Delete Operation ....................................................................... 2-15
2.3.5 Configuring the Update Operation ....................................................................... 2-18
2.3.6 Configuring the Enable and Disable Operations for Provisioning ....................... 2-21
2.3.7 Configuring the Search Operation ....................................................................... 2-22
2.3.7.1 Mapping Simple Child Table Values in the SOA Composite ............................... 2-31
2.3.8 Configuring the Enable and Disable Operations for Reconciliation .................... 2-32
2.3.9 Configuring the Lookup Search Operation ............................................................ 2-33
2.3.10 Configuring the Reset Password Operation ........................................................ 2-38
2.4 Handling Faults....................................................................................................... 2-41
2.4.1 Handling Faults with Catch Blocks .................................................................... 2-49
2.5 Deploying and Testing the Webservice SOA Composite .......................................... 2-51

### 3 Deploying the Connector

3.1 Installation ............................................................................................................... 3-1
3.1.1 Running the Connector Installer ........................................................................... 3-1
3.1.2 Configuring the IT Resource .............................................................................. 3-3
3.2 Postinstallation ........................................................................................................ 3-6
3.2.1 Configuring Oracle Identity Manager 11.1.2 or Later ........................................ 3-6
3.2.1.1 Creating and Activating a Sandbox ................................................................. 3-6
3.2.1.2 Creating a New UI Form ................................................................................ 3-7
3.2.1.3 Creating an Application Instance ................................................................. 3-7
3.2.1.4 Publishing a Sandbox .................................................................................... 3-7
3.2.1.5 Harvesting Entitlements and Sync Catalog .................................................. 3-7
3.2.1.6 Updating an Existing Application Instance with a New Form ....................... 3-8
3.2.2 Enabling Logging ............................................................................................... 3-8
3.2.3 Setting up the Lookup Definition for Connection Pooling ................................ 3-10
3.2.4 Changing to the Required Input Locale .............................................................. 3-11
3.2.5 Clearing Content Related to Connector Resource Bundles from the Server Cache 3-11
3.2.6 Disabling Child Tables ...................................................................................... 3-12
3.2.7 Removing Bulk Attribute Update Task .............................................................. 3-12
3.2.8 Localizing Field Labels in UI Forms .................................................................... 3-12
## 4 Using the Connector

4.1 Configuring Reconciliation ................................................................. 4-1
4.1.1 Full Reconciliation ........................................................................ 4-2
4.1.2 Limited Reconciliation ................................................................. 4-2
4.1.3 Batched Reconciliation ................................................................. 4-2
4.1.4 Configuring the Target System As a Trusted Source ................. 4-3
4.2 Scheduled Tasks .............................................................................. 4-3
4.2.1 Scheduled Task for Lookup Field Synchronization .................. 4-3
4.2.2 Scheduled Tasks for Reconciliation ........................................... 4-4
4.2.3 Delete User Target Reconciliation ............................................. 4-5
4.2.3.1 Adding defaultBatchSize as a Configuration Property .... 4-5
4.2.4 Configuring Scheduled Jobs ....................................................... 4-6
4.3 Configuring Provisioning in Oracle Identity Manager Release 11.1.1 4-7
4.3.1 Configuring Direct Provisioning .................................................. 4-8
4.3.2 Configuring Request-Based Provisioning .................................... 4-8
4.3.2.1 End User’s Role in Request-Based Provisioning ................. 4-9
4.3.2.2 Approver’s Role in Request-Based Provisioning ................ 4-10
4.3.2.3 Enabling the Auto Save Form Feature ................................. 4-10
4.3.2.4 Running the PurgeCache Utility ........................................... 4-11
4.3.3 Switching Between Request-Based Provisioning and Direct Provisioning 4-11
4.4 Configuring Provisioning in Oracle Identity Manager Release 11.1.2 4-12
4.5 Uninstalling the Connector ............................................................. 4-13

## 5 Extending the Functionality of the Connector

5.1 Securing the Connector ................................................................. 5-1
5.1.1 Handling Passwords ................................................................. 5-2
5.1.2 Configuring Webservice Security Policy ..................................... 5-5
5.1.3 Passing Credentials Using CSF .................................................. 5-5
5.1.4 Passing Credentials Using Custom Headers ............................. 5-7
5.1.5 Importing SSL Certificate for HTTPS-based Target Webservice 5-11
5.2 Adding Custom Attributes for Provisioning .................................... 5-12
5.2.1 Adding Custom Attributes for Provisioning in Oracle Identity Manager 5-12
5.2.2 Adding Custom Attributes for Provisioning in SOA Composite 5-13
5.2.3 Adding Custom Attribute for Update Operation ....................... 5-14
5.3 Adding Custom Attributes for Reconciliation ................................ 5-15
5.3.1 Adding Custom Attributes for Reconciliation in Oracle Identity Manager 5-16
5.3.2 Adding Custom Attributes for Reconciliation in SOA Composite 5-17
5.3.3 Adding Custom Attributes for Reconciling _UID_ Field ........... 5-20
5.4 Adding Custom Child Forms .......................................................... 5-24
5.4.1 Adding Custom Child Forms in Oracle Identity Manager .......... 5-24
5.4.2 Adding Custom Child Forms in SOA Composite ...................... 5-26
5.5 Adding Child Form Data ............................................................... 5-29
5.6 Mapping Timestamp Attribute ..................................................... 5-30
5.7 Configuring the Connector for Multiple Instances and Multiple Versions of the Target System 5-32
5.8 Configuring Validation of Data During Reconciliation and Provisioning ....... 5-32
List of Tables

<table>
<thead>
<tr>
<th>Table No.</th>
<th>Description</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>1–1</td>
<td>Certified Components</td>
<td>1-2</td>
</tr>
<tr>
<td>1–2</td>
<td>Entries in the Configuration Lookup Definitions</td>
<td>1-7</td>
</tr>
<tr>
<td>1–3</td>
<td>Entries in the Lookup.ACME.UM.Configuration Lookup Definition</td>
<td>1-8</td>
</tr>
<tr>
<td>1–4</td>
<td>Entries in the Lookup.ACME.UM.Configuration.Trusted Lookup Definition</td>
<td>1-8</td>
</tr>
<tr>
<td>1–5</td>
<td>Entries in the Lookup.ACME.UM.ReconAttrMap Lookup Definition</td>
<td>1-10</td>
</tr>
<tr>
<td>1–6</td>
<td>User Provisioning Functions</td>
<td>1-11</td>
</tr>
<tr>
<td>1–7</td>
<td>Entries in the Lookup.ACME.UM.ProvAttrMap Lookup Definition</td>
<td>1-11</td>
</tr>
<tr>
<td>2–1</td>
<td>Files and Directories Before Building the Connector</td>
<td>2-2</td>
</tr>
<tr>
<td>2–2</td>
<td>Files and Directories Generated After Building the Connector</td>
<td>2-4</td>
</tr>
<tr>
<td>3–1</td>
<td>IT Resource Parameters</td>
<td>3-5</td>
</tr>
<tr>
<td>3–2</td>
<td>Log Levels and ODL Message Type:Level Combinations</td>
<td>3-8</td>
</tr>
<tr>
<td>3–3</td>
<td>Connection Pooling Properties</td>
<td>3-10</td>
</tr>
<tr>
<td>4–1</td>
<td>Attributes of the Scheduled Task for Lookup Field Synchronization</td>
<td>4-3</td>
</tr>
<tr>
<td>4–2</td>
<td>Attributes of the Scheduled Tasks for Reconciliation</td>
<td>4-4</td>
</tr>
<tr>
<td>4–3</td>
<td>Attributes of the Delete User Target Reconciliation Scheduled Job</td>
<td>4-5</td>
</tr>
</tbody>
</table>
This guide describes the connector that is used to integrate Oracle Identity Manager with Webservices.

**Audience**

This guide is intended for resource administrators and target system integration teams.

**Documentation Accessibility**

For information about Oracle's commitment to accessibility, visit the Oracle Accessibility Program website at http://www.oracle.com/pls/topic/lookup?ctx=acc&id=docacc.

**Access to Oracle Support**

Oracle customers that have purchased support have access to electronic support through My Oracle Support. For information, visit http://www.oracle.com/pls/topic/lookup?ctx=acc&id=info or visit http://www.oracle.com/pls/topic/lookup?ctx=acc&id=trs if you are hearing impaired.

**Related Documents**

For information about installing and using Oracle Identity Manager, visit the following Oracle Help Center page:

http://docs.oracle.com/cd/E52734_01/index.html

For information about Oracle Identity Manager Connectors documentation, visit the following Oracle Help Center page:

http://docs.oracle.com/cd/E22999_01/index.htm

**Documentation Updates**

Oracle is committed to delivering the best and most recent information available. For information about updates to the Oracle Identity Manager Connectors documentation library, visit Oracle Technology Network at

http://download.oracle.com/docs/cd/E22999_01/index.htm
Conventions

The following text conventions are used in this document:

<table>
<thead>
<tr>
<th>Convention</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>boldface</strong></td>
<td>Boldface type indicates graphical user interface elements associated with an action, or terms defined in text or the glossary.</td>
</tr>
<tr>
<td><em>italic</em></td>
<td>Italic type indicates book titles, emphasis, or placeholder variables for which you supply particular values.</td>
</tr>
<tr>
<td>monospace</td>
<td>Monospace type indicates commands within a paragraph, URLs, code in examples, text that appears on the screen, or text that you enter.</td>
</tr>
</tbody>
</table>
What's New in Oracle Identity Manager Connector for Webservices?

This chapter provides an overview of the updates made to the software and documentation for the Webservices connector in release 11.1.1.5.0.

The updates discussed in this chapter are divided into the following categories:

- **Software Updates**
  - This section describes updates made to the connector software.

- **Documentation-Specific Updates**
  - This section describes major changes made to this guide. These changes are not related to software updates.

### Software Updates

The following section discusses the software update:

**Software Update in Release 11.1.1.5.0**

This is the first release of the Oracle Identity Manager connector for Webservices based on ICF architecture. Therefore, there are no software updates in this release.

### Documentation-Specific Updates

The following section discusses the documentation-specific update:

**Documentation-Specific Update in Release 11.1.1.5.0**

The following documentation-specific updates have been made in revision “9” of release 11.1.1.5.0:

- The "Oracle Identity Manager" row of Table 1–1, "Certified Components" has been renamed as "Oracle Identity Governance or Oracle Identity Manager" and also updated for Oracle Identity Governance 12c (12.2.1.3.0) certification.

- The "Note" in Section 2.1, "Prerequisites" has been modified to include a link to download and install the SOAComposite Editor extension.

- The command to build the connector zip file for the ACME Webservice on Linux has been modified in Step 3 of Section 2.2, "Building the Connector Bundle."

- Section 2.3.1, "Configuring the Partner Link", the following changes have been made:
- The url to download and install the 11.1.1.9.0 version of JDeveloper in the "Note" in Step 1 has been modified
- A partner link url is added to Step 4.

**■ The following updates are made in Section 2.3.3, "Configuring the Create Operation for SPML."
- Two bullet point about testing SPML and non-SPML targets have been added to Step 1.
- A link for creating your own WSDL has been added to Step 3.

■ Step 1.b of Section 5.1.1, "Handling Passwords" has been modified.

■ The following appendices have been added:
  - Appendix B, "Sample Outbound Policy"
  - Appendix C, "Sample WSDL for Security Policy"
  - Appendix D, "Sample XSDs"

The following documentation-specific updates have been made in revision "8" of release 11.1.1.5.0:

■ The "Connector Server" row has been added to Table 1–1, "Certified Components".

■ The "JDK" row of Table 1–1, "Certified Components" has been renamed to "Connector Server JDK".

The following documentation-specific updates have been made in revision "7" of release 11.1.1.5.0:

■ A "Note" regarding the target.payload.namespace property has been added to Step 10 of Section 2.3.10, "Configuring the Reset Password Operation."

■ A "Note" regarding trusted source IT resource has been added at the beginning of Section 3.1.2, "Configuring the IT Resource."

■ Updated a "Note" present in Step 3 of Section 5.1.1, "Handling Passwords."

■ Added information specific to the target.payload.namespace property to Section 6.1, "Troubleshooting."

■ Schema Definition (XSD) links have been added to Step 1 of Section 2.3.3, "Configuring the Create Operation for SPML."

■ Step 4 of Section 2.3.3, "Configuring the Create Operation for SPML." has been modified.

■ In Step 1.b of Section 5.1.2, "Configuring Webservice Security Policy", the OutboundPolicy.zip is added.

The following documentation-specific update has been made in revision "6" of release 11.1.1.5.0:

The "Oracle Identity Manager" row of Table 1–1, "Certified Components" has been updated.

The following documentation-specific update has been made in revision "5" of release 11.1.1.5.0:

A "Note" has been added at the beginning of Chapter 5, "Extending the Functionality of the Connector."

The following documentation-specific updates have been made in the revision "4" of release 11.1.1.5.0:
- Updated the "Note" in step 18 of Section 2.3.7, "Configuring the Search Operation."
- Added Section 2.3.7.1, "Mapping Simple Child Table Values in the SOA Composite."
- Added Section 5.11, "Reconciliation of Complex Child Forms With Multiple Attributes."
- Added a "Note" at the end of Section 2.3.7, "Configuring the Search Operation."
- Added Section 4.2.3.1, "Adding defaultBatchSize as a Configuration Property."
- Added a "Note" to Section 5.4.1, "Adding Custom Child Forms in Oracle Identity Manager."

The following documentation-specific update has been made in the revision "3" of release 11.1.1.5.0:

- Information about limited reconciliation has been modified in Section 4.1.2, "Limited Reconciliation."

The following documentation-specific updates have been made in the revision "2" of release 11.1.1.5.0:

- The "Oracle Identity Manager" row of Table 1–1, "Certified Components" has been updated.
- Figure 1–1, "Architecture of the Connector" has been updated.
- Added a note to step 18 of Section 2.3.7, "Configuring the Search Operation."
- Added Section 5.3.3, "Adding Custom Attributes for Reconciling _UID_ Field."
- A screenshot has been added to Section 3.1.2, "Configuring the IT Resource."
- A note has been added in step 3 of Section 2.3.1, "Configuring the Partner Link."
- Step 5 has been added to Section 2.3.2, "Configuring the Create Operation."
- The name of the "Known Issues" chapter has been changed to "Known Issues and Workarounds." In addition, Chapter 7, "Known Issues and Workarounds" has been restructured.
About the Connector

Oracle Identity Manager automates access rights management, and the security of resources to various target systems. Oracle Identity Manager connectors are used to integrate Oracle Identity Manager with target applications. This guide discusses the Webservices connector that connects to a target system exposing a webservice endpoint. The target system can be used as managed (target) resource or authoritative (trusted) source of identity information for Oracle Identity Manager. The connector uses Oracle SOA Suite as the indirection layer and supports all versions of webservices supported by that version of SOA Suite.

**Note:** In this guide, a target system that exposes webservice endpoint has been referred to as the **target system**. ACME Webservice is used as a sample target system to discuss the configurations and the connector objects.

In the account management (target resource) mode of the connector, data about users created or modified directly on the target system can be reconciled into Oracle Identity Manager. This data is used to provision (allocate) new resources or update resources already assigned to OIM Users. In addition, you can use Oracle Identity Manager to provision or update target resources assigned to OIM Users. These provisioning operations performed on Oracle Identity Manager translate into the creation of or updates to target system accounts.

In the identity reconciliation (trusted source) configuration of the connector, persons are created or modified only on the target system and information about these persons is reconciled into Oracle Identity Manager.

This chapter contains the following sections:

- Section 1.1, "Certified Components"
- Section 1.2, "Certified Languages"
- Section 1.3, "Connector Architecture"
- Section 1.4, "Features of the Connector"
- Section 1.5, "Lookup Definitions Used During Connector Operations"
- Section 1.6, "Connector Objects Used During Reconciliation"
- Section 1.7, "Connector Objects Used During Provisioning"
- Section 1.8, "Roadmap for Deploying and Using the Connector"
1.1 Certified Components

Table 1–1 lists the components certified for use with the connector.

<table>
<thead>
<tr>
<th>Item</th>
<th>Requirement</th>
</tr>
</thead>
<tbody>
<tr>
<td>Oracle Identity Governance or Oracle Identity Manager</td>
<td>You can use one of the following releases of Oracle Identity Governance or Oracle Identity Manager:</td>
</tr>
<tr>
<td></td>
<td>■ Oracle Identity Governance 12c (12.2.1.3.0)</td>
</tr>
<tr>
<td></td>
<td>■ Oracle Identity Manager 11g Release 2 PS3 (11.1.2.3.0)</td>
</tr>
<tr>
<td></td>
<td>■ Oracle Identity Manager 11g Release 2 (11.1.2.0.7) BP07 and any later BP in this release track</td>
</tr>
<tr>
<td></td>
<td>■ Oracle Identity Manager 11g Release 1 (11.1.1.5.6) BP06 (with patch 15971939) and any later BP in this release track</td>
</tr>
<tr>
<td>Target system</td>
<td>Any target system that exposes webservice endpoints</td>
</tr>
<tr>
<td>Connector Server</td>
<td>11.1.2.1.0</td>
</tr>
<tr>
<td>Connector Server JDK</td>
<td>JDK 1.6 or later, or JRockit 1.6 or later</td>
</tr>
</tbody>
</table>

1.2 Certified Languages

The connector supports the following languages:

- Arabic
- Chinese (Simplified)
- Chinese (Traditional)
- Czech
- Danish
- Dutch
- English
- Finnish
- French
- German
- Greek
- Hebrew
- Hungarian
- Italian
- Japanese
- Korean
- Norwegian
- Polish
- Portuguese
- Portuguese (Brazilian)
1.3 Connector Architecture

Figure 1–1 shows the architecture of the connector.

Figure 1–1  Architecture of the Connector

The connector is implemented by using the Identity Connector Framework (ICF). ICF is a component that provides basic reconciliation and provisioning operations that are common to all Oracle Identity Manager connectors. In addition, ICF provides common features that developers would otherwise need to implement on their own, such as connection pooling, buffering, time outs, and filtering. The ICF is shipped along with Oracle Identity Manager. Therefore, you need not configure or modify the ICF.

The connector architecture can be described as follows:

- Webservices are web APIs exposed by web applications to enable inter operability with their applications. Operations exposed by webservice can be invoked and used via SOAP protocol.

- The connector uses SOA to bind and invoke operations on the target webservice. SOA acts as the indirection layer. SOA composite is wired on one end to the webservice client, represented as WSConnector Client PartnerLink in Figure 1–1. This client is the connector's webservice endpoint that exposes ICF-based operations such as create, delete, update, and search. On the other end, the SOA composite is wired to a target webservice that exposes similar operations but with its own input and output conventions.

- The WSConnector Client endpoint makes use of oracle/wss_username_token_client_policy webservice security policy for authentication.

Note: However, the connector does not support the entry of multibyte characters in some of the fields.
The connector is responsible for invoking ICF operations on the SOA composite that are generated by the connector’s webservice client with a specific input structure. This triggers the BPEL process for the specific type of operation, such as create, which in turn invokes the operation on the target webservice.

The output is passed to the SOA composite, which optionally can use XSLT to transform the payload into a structure that ICF understands. This transformation and wiring is handled in the SOA composite, which has to be manually configured by the user.

The webservice client that the connector interacts with exposes a contract corresponding to the ICF adapters and objects. WSDL contains definitions for the operations, the input and output schema specific to each operation, exception handling by declaring exceptions such as UnknownUidException and AlreadyExistsException, and the custom or child table attributes.

The connector package contains a base SOA composite template, the ICF webservice connector bundle, and the Oracle Identity Manager metadata. The Oracle Identity Manager metadata is pre-defined and can be updated as per your requirements. The metadata can be considered as a template and can be customized to suit the target accounts. The target system operations trigger ICF operations and are routed to the ICF webservice connector bundle. The wiring of the target weervices is done at the SOA composite layer. The ICF connector bundle makes a call to the respective operation at the SOA composite layer.

The connector expects operations exposed as SOAP services. The SOAP operations are offered based on a pre-defined WSDL contract. This WSDL contract has one operation each for create, update, delete, addAttributeValue, removeAttributeValue, lookupSearch, and accountSearch operations. Each operation in the BPEL process is processed within its respective branch.

The connector can invoke a different webservice operation or a different target system webservice for various operations. This architecture is primarily focused to support synchronous webservces where the result is returned within the same call.

1.4 Features of the Connector

- Section 1.4.1, "Support for Configuring the Connector for a New Target System"
- Section 1.4.2, "Support for Securing the Connector"
- Section 1.4.3, "Support for Multiple Instances and Multiple Versions of Target Systems"
- Section 1.4.4, "Support for Both Target Resource and Trusted Source Reconciliation"
- Section 1.4.5, "Support for Both Full and Incremental Reconciliation"
- Section 1.4.6, "Support for Limited Reconciliation"
- Section 1.4.7, "Support for Batched Reconciliation"
- Section 1.4.8, "Validation of Data"
- Section 1.4.9, "Transformation of Data"
- Section 1.4.10, "Support for Resource Exclusion Lists"
1.4.1 Support for Configuring the Connector for a New Target System

You can configure the connector to support an additional target system that exposes webservice endpoint.

The connector package contains a base SOA template composite, the ICF webservice connector bundle, and the Oracle Identity Manager metadata.

For more information, see the procedures described in Chapter 2, "Preinstallation Steps" and Chapter 3, "Deploying the Connector."

1.4.2 Support for Securing the Connector

You can configure the connector in Oracle Identity Manager and in the SOA composite to secure the connector.

See Section 5.1, "Securing the Connector" for security-related topics.

1.4.3 Support for Multiple Instances and Multiple Versions of Target Systems

The connector supports multiple instances and multiple versions of target systems.

You can deploy a single connector bundle on Oracle Identity Manager and create multiple IT resources for multiple instances and multiple versions of target systems. Then, you can use Oracle Identity Manager to manage accounts on these target systems. See Section 5.7, "Configuring the Connector for Multiple Instances and Multiple Versions of the Target System" for more information.

1.4.4 Support for Both Target Resource and Trusted Source Reconciliation

You can use the connector to configure the target system as either a target resource or trusted source of Oracle Identity Manager.

See Section 4.1, "Configuring Reconciliation" for more information.

1.4.5 Support for Both Full and Incremental Reconciliation

After you deploy the connector, you can perform full reconciliation to bring all existing user data from the target system to Oracle Identity Manager. After the first full reconciliation run, incremental reconciliation is automatically enabled from the next run of the user reconciliation.

You can perform a full reconciliation run at any time. See Section 4.1.1, "Full Reconciliation" for more information.

1.4.6 Support for Limited Reconciliation

You can set a reconciliation filter as the value of the Filter attribute of the scheduled tasks. This filter specifies the subset of newly added and modified target system records that must be reconciled.

See Section 4.1.2, "Limited Reconciliation" for more information.

1.4.7 Support for Batched Reconciliation

You can break down a reconciliation run into batches by specifying the number of records that must be included in each batch.

See Section 4.1.3, "Batched Reconciliation" for more information.
1.4.8 Validation of Data

You can configure single-valued data to be validated during provisioning and reconciliation operations.

See Section 5.8, "Configuring Validation of Data During Reconciliation and Provisioning" for more information.

1.4.9 Transformation of Data

You can configure transformation of data that is brought into Oracle Identity Manager during reconciliation.

See Section 5.9, "Configuring Transformation of Data During User Reconciliation" for more information.

1.4.10 Support for Resource Exclusion Lists

You can specify a list of accounts that must be excluded from reconciliation and provisioning operations. Accounts whose user IDs you specify in the exclusion list are not affected by reconciliation and provisioning operations.

Section 5.10, "Configuring Resource Exclusion Lists" describes the procedure to add entries in these lookup definitions.

1.5 Lookup Definitions Used During Connector Operations

Lookup definitions used during connector operations can be categorized as follows:

- Section 1.5.1, "Lookup Definitions Synchronized with the Target System"
- Section 1.5.2, "Preconfigured Lookup Definitions"

1.5.1 Lookup Definitions Synchronized with the Target System

During a provisioning operation, you use a lookup field on the process form to specify a single value from a set of values. For example, you use the Date Format lookup field to select a date format from the list of supported date formats. When you deploy the connector, lookup definitions corresponding to the lookup fields on the target system are automatically created in Oracle Identity Manager. Lookup field synchronization involves copying additions or changes made to the target system lookup fields into the lookup definitions in Oracle Identity Manager.

The lookup reconciliation scheduled job is used to synchronize value of lookup definitions with the target system. See Section 4.2.1, “Scheduled Task for Lookup Field Synchronization” for more information.

While performing a provisioning operation on the Administrative and User Console, you select the IT resource for the target system on which you want to perform the operation. When you perform this action, the lookup definitions on the page are automatically populated with values corresponding to the IT resource (target system installation) that you select.

1.5.2 Preconfigured Lookup Definitions

This section discusses the other lookup definitions that are created in Oracle Identity Manager when you deploy the connector. These lookup definitions are either prepopulated with values or values must be manually entered in them after the connector is deployed. The other lookup definitions are as follows:
### 1.5.2.1 Configuration Lookup Definitions

The Lookup.ACME.Configuration and Lookup.ACME.Configuration.Trusted lookup definitions hold connector configuration entries that are used during reconciliation and provisioning operations.

Table 1–2 lists the default entries in this lookup definition.

<table>
<thead>
<tr>
<th>Code Key</th>
<th>Decode</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Bundle Name</td>
<td>org.identityconnectors.webservices</td>
<td>This entry holds the name of the connector bundle package. Do not modify this entry.</td>
</tr>
<tr>
<td>Bundle Version</td>
<td>1.0.112</td>
<td>This entry holds the version of the connector bundle class. Do not modify this entry.</td>
</tr>
<tr>
<td>Connector Name</td>
<td>org.identityconnectors.genericws.GenericWSConnector</td>
<td>This entry holds the name of the connector class. Do not modify this entry.</td>
</tr>
<tr>
<td>internalPolicyReference</td>
<td>false</td>
<td>Internal OWSM policy reference for the connector during standalone operations.</td>
</tr>
<tr>
<td>User Configuration Lookup</td>
<td></td>
<td>For target resource mode: Lookup.ACME.UM.Configuration For trusted mode: Lookup.ACME.UM.Configuration.Trusted This entry holds the name of the lookup definition that contains user-specific configuration properties. Do not modify this entry.</td>
</tr>
</tbody>
</table>

### 1.5.2.2 Lookup.ACME.UM.Configuration

The Lookup.ACME.UM.Configuration lookup definition holds configuration entries that are specific to the user object type. This lookup definition is used during user management operations when your target system is configured as a target resource.

Table 1–3 lists the default entries in this lookup definition.
### 1.5.2.3 Lookup.ACME.UM.Configuration.Trusted

The Lookup.ACME.UM.Configuration.Trusted lookup definition holds configuration entries that are specific to the user object type. This lookup definition is used during user management operations when your target system is configured as a trusted source.

Table 1–4 lists the default entries in this lookup definition.

<table>
<thead>
<tr>
<th>Code Key</th>
<th>Decode</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Provisioning Attribute Map</td>
<td>Lookup.ACME.UM.ProvAttrMap</td>
<td>This entry holds the name of the lookup definition that maps process form fields and target system attributes. See Section 1.5.2.4, &quot;Lookup.ACME.UM.ProvAttrMap&quot; for more information about this lookup definition.</td>
</tr>
<tr>
<td>Recon Attribute Map</td>
<td>Lookup.ACME.UM.ReconAttrMap</td>
<td>This entry holds the name of the lookup definition that maps resource object fields and target system attributes. See Section 1.5.2.5, &quot;Lookup.ACME.UM.ReconAttrMap&quot; for more information about this lookup definition.</td>
</tr>
</tbody>
</table>

### 1.5.2.4 Lookup.ACME.UM.ProvAttrMap

The Lookup.ACME.UM.ProvAttrMap lookup definition holds mappings between process form fields and target system attributes. This lookup definition is used during provisioning. This lookup definition is preconfigured. Table 1–7 lists the default entries.

You can add entries in this lookup definitions if you want to map new target system attributes for provisioning. See Section 5.2, "Adding Custom Attributes for Provisioning" for more information.

### 1.5.2.5 Lookup.ACME.UM.ReconAttrMap

The Lookup.ACME.UM.ReconAttrMap lookup definition holds mappings between resource object fields and target system attributes. This lookup definition is used during reconciliation. This lookup definition is preconfigured. Table 1–5 lists the default entries.

You can add entries in this lookup definitions if you want to map new target system attributes for reconciliation. See Section 5.3, "Adding Custom Attributes for Reconciliation" for more information.
1.5.2.6 Lookup.ACME.UM.ReconAttrMap.Trusted

The Lookup.ACME.UM.ReconAttrMap.Trusted lookup definition holds mappings between resource object fields and target system attributes. This lookup definition is used during reconciliation in trusted mode.

This lookup definition contains the following entries:

<table>
<thead>
<tr>
<th>Code Key</th>
<th>Decode</th>
</tr>
</thead>
<tbody>
<tr>
<td>First Name</td>
<td>FirstName</td>
</tr>
<tr>
<td>Last Name</td>
<td>LastName</td>
</tr>
<tr>
<td>Status[TRUSTED]</td>
<td>Status</td>
</tr>
<tr>
<td>Unique Id</td>
<td><strong>UID</strong></td>
</tr>
</tbody>
</table>

1.5.2.7 Lookup.ACME.UM.ReconDefaults.Trusted

The Lookup.ACME.UM.ReconDefaults.Trusted lookup definition holds mappings between process form fields and target system attributes. This lookup definition is used during reconciliation in trusted mode.

This lookup definition contains the following entries:

<table>
<thead>
<tr>
<th>Code Key</th>
<th>Decode</th>
</tr>
</thead>
<tbody>
<tr>
<td>Empl Type</td>
<td>Full-Time</td>
</tr>
<tr>
<td>Organization Name</td>
<td>Xellerate Users</td>
</tr>
<tr>
<td>Status</td>
<td>Active</td>
</tr>
<tr>
<td>User Type</td>
<td>End User</td>
</tr>
</tbody>
</table>

1.6 Connector Objects Used During Reconciliation

The User Target Reconciliation and User Trusted Reconciliation scheduled jobs are used to initiate reconciliation runs. These scheduled jobs are discussed in Section 4.2, "Scheduled Tasks."

See Also: Managing Reconciliation in Oracle Fusion Middleware Administering Oracle Identity Manager for conceptual information about reconciliation

This section discusses the following topics:

- Section 1.6.1, "User Attributes for Reconciliation"

1.6.1 User Attributes for Reconciliation

The Lookup.ACME.UM.ReconAttrMap lookup definition maps resource object fields and target system attributes.

The Code Key column stores the names of resource object fields.

The Decode column stores the names of the target system attributes.

Table 1–5 lists entries in this lookup definition.
<table>
<thead>
<tr>
<th>Resource Object Field</th>
<th>Target System Attribute</th>
</tr>
</thead>
<tbody>
<tr>
<td>Address</td>
<td>Address</td>
</tr>
<tr>
<td>Common Name</td>
<td>CommonName</td>
</tr>
<tr>
<td>Country</td>
<td>Country</td>
</tr>
<tr>
<td>Department Number</td>
<td>DepartmentNumber</td>
</tr>
<tr>
<td>Deprovisioning Date</td>
<td>DeprovisioningDate</td>
</tr>
<tr>
<td>Display Name</td>
<td>DisplayName</td>
</tr>
<tr>
<td>Email</td>
<td>Email</td>
</tr>
<tr>
<td>Employee Number</td>
<td>Empno</td>
</tr>
<tr>
<td>End Date</td>
<td>EndDate</td>
</tr>
<tr>
<td>Fax</td>
<td>Fax</td>
</tr>
<tr>
<td>First Name</td>
<td>FirstName</td>
</tr>
<tr>
<td>Generation Qualifier</td>
<td>GenerationQualifier</td>
</tr>
<tr>
<td>Hire Date</td>
<td>HireDate</td>
</tr>
<tr>
<td>Home Address</td>
<td>HomeAddress</td>
</tr>
<tr>
<td>Home Phone</td>
<td>HomePhone</td>
</tr>
<tr>
<td>Initials</td>
<td>Initials</td>
</tr>
<tr>
<td>Last Name</td>
<td>LastName</td>
</tr>
<tr>
<td>Locality</td>
<td>Locality</td>
</tr>
<tr>
<td>Login</td>
<td><strong>NAME</strong></td>
</tr>
<tr>
<td>Manager</td>
<td>Manager</td>
</tr>
<tr>
<td>Middle Name</td>
<td>MiddleName</td>
</tr>
<tr>
<td>Mobile</td>
<td>Mobile</td>
</tr>
<tr>
<td>OIMObjectStatus</td>
<td><strong>ENABLE</strong></td>
</tr>
<tr>
<td>Organization</td>
<td>Organization</td>
</tr>
<tr>
<td>Pager</td>
<td>Pager</td>
</tr>
<tr>
<td>Password</td>
<td><strong>PASSWORD</strong></td>
</tr>
<tr>
<td>PO Box</td>
<td>POBox</td>
</tr>
<tr>
<td>Provisioning Date</td>
<td>ProvisioningDate</td>
</tr>
<tr>
<td>Roles~Role</td>
<td>Role</td>
</tr>
<tr>
<td>Start Date</td>
<td>StartDate</td>
</tr>
<tr>
<td>State</td>
<td>State</td>
</tr>
<tr>
<td>Status</td>
<td>Status</td>
</tr>
<tr>
<td>Street</td>
<td>Street</td>
</tr>
<tr>
<td>Telephone Number</td>
<td>TelephoneNumber</td>
</tr>
<tr>
<td>Title</td>
<td>Title</td>
</tr>
<tr>
<td>Unique Id</td>
<td><strong>UID</strong></td>
</tr>
<tr>
<td>User Type</td>
<td>UserType</td>
</tr>
</tbody>
</table>
1.7 Connector Objects Used During Provisioning

Provisioning involves creating or modifying user data on the target system through Oracle Identity Manager.

This section discusses the following topics:

- Section 1.7.1, "User Provisioning Functions"
- Section 1.7.2, "User Attributes for Provisioning"

1.7.1 User Provisioning Functions

Table 1–6 lists the supported user provisioning functions and the adapters that perform these functions. The functions listed in the table correspond to either a single or multiple process tasks.

See Also: Types of Adapters in Oracle Fusion Middleware Developing and Customizing Applications for Oracle Identity Manager for generic information about process tasks and adapters

<table>
<thead>
<tr>
<th>Function</th>
<th>Task Adapter</th>
</tr>
</thead>
<tbody>
<tr>
<td>Create a user account</td>
<td>adpACMEWEBSERVICECREATEOBJECT</td>
</tr>
<tr>
<td>Delete a user account</td>
<td>adpACMEWEBSERVICEDELETEOBJECT</td>
</tr>
<tr>
<td>Enable a user account</td>
<td>adpACMEWEBSERVICEENABLEUSER</td>
</tr>
<tr>
<td>Disable a user account</td>
<td>adpACMEWEBSERVICEDISABLEUSER</td>
</tr>
<tr>
<td>Update an attribute</td>
<td>adpACMEWEBSERVICEUPDATEATTRIBUTEVALUE</td>
</tr>
<tr>
<td>Bulk update of attributes</td>
<td>adpACMEWEBSERVICEBULKUPDATE</td>
</tr>
<tr>
<td>Add a child table value</td>
<td>adpACMEWEBSERVICEADDCHILDTABLEVALUE</td>
</tr>
<tr>
<td>Remove a child table value</td>
<td>adpACMEWEBSERVICEREMOVECHILDTABLEVALUE</td>
</tr>
<tr>
<td>Update a child table value</td>
<td>adpACMEWEBSERVICEUPDATECHILDTABLEVALUE</td>
</tr>
</tbody>
</table>

1.7.2 User Attributes for Provisioning

The Lookup.ACME.UM.ProvAttrMap lookup definition maps process form fields with single-valued target system attributes.

The Code Key column holds the names of process form fields.

The Decode column stores the names of the target system attributes.

Table 1–7 lists the entries in this lookup definition.

<table>
<thead>
<tr>
<th>Process Form Field</th>
<th>Target System Attribute</th>
</tr>
</thead>
<tbody>
<tr>
<td>Address</td>
<td>Address</td>
</tr>
<tr>
<td>Common Name</td>
<td>CommonName</td>
</tr>
<tr>
<td>Country</td>
<td>Country</td>
</tr>
<tr>
<td>Department Number</td>
<td>DepartmentNumber</td>
</tr>
<tr>
<td>Deprovisioning Date</td>
<td>DeprovisioningDate</td>
</tr>
</tbody>
</table>
Table 1–7  (Cont.) Entries in the Lookup.ACME.UM.ProvAttrMap Lookup Definition

<table>
<thead>
<tr>
<th>Process Form Field</th>
<th>Target System Attribute</th>
</tr>
</thead>
<tbody>
<tr>
<td>Display Name</td>
<td>DisplayName</td>
</tr>
<tr>
<td>Email</td>
<td>Email</td>
</tr>
<tr>
<td>Employee Number</td>
<td>Empno</td>
</tr>
<tr>
<td>End Date[DATE]</td>
<td>EndDate</td>
</tr>
<tr>
<td>Fax</td>
<td>Fax</td>
</tr>
<tr>
<td>First Name</td>
<td>FirstName</td>
</tr>
<tr>
<td>Generation Qualifier</td>
<td>GenerationQualifier</td>
</tr>
<tr>
<td>Hire Date[DATE]</td>
<td>HireDate</td>
</tr>
<tr>
<td>Home Address</td>
<td>HomeAddress</td>
</tr>
<tr>
<td>Home Phone</td>
<td>HomePhone</td>
</tr>
<tr>
<td>Initials</td>
<td>Initials</td>
</tr>
<tr>
<td>Last Name</td>
<td>LastName</td>
</tr>
<tr>
<td>Locality</td>
<td>Locality</td>
</tr>
<tr>
<td>Login</td>
<td><strong>NAME</strong></td>
</tr>
<tr>
<td>Manager</td>
<td>Manager</td>
</tr>
<tr>
<td>Middle Name</td>
<td>MiddleName</td>
</tr>
<tr>
<td>Mobile</td>
<td>Mobile</td>
</tr>
<tr>
<td>Organization</td>
<td>Organization</td>
</tr>
<tr>
<td>Pager</td>
<td>Pager</td>
</tr>
<tr>
<td>Password</td>
<td><strong>PASSWORD</strong></td>
</tr>
<tr>
<td>PO Box</td>
<td>POBox</td>
</tr>
<tr>
<td>Provisioning Date[DATE]</td>
<td>ProvisioningDate</td>
</tr>
<tr>
<td>Start Date[DATE]</td>
<td>StartDate</td>
</tr>
<tr>
<td>State</td>
<td>State</td>
</tr>
<tr>
<td>Status</td>
<td>Status</td>
</tr>
<tr>
<td>Street</td>
<td>Street</td>
</tr>
<tr>
<td>Telephone Number</td>
<td>TelephoneNumber</td>
</tr>
<tr>
<td>Title</td>
<td>Title</td>
</tr>
<tr>
<td>UD_ACME_CH~Role</td>
<td>Role</td>
</tr>
<tr>
<td>Unique Id</td>
<td><strong>UID</strong></td>
</tr>
<tr>
<td>User Type</td>
<td>UserType</td>
</tr>
</tbody>
</table>

1.8 Roadmap for Deploying and Using the Connector

The following shows how information is organized in the rest of the guide:

- Chapter 2, "Preinstallation Steps" describes procedures that you must perform before deploying the connector.
Chapter 3, "Deploying the Connector" describes procedures that you must perform on Oracle Identity Manager and the target system during each stage of connector deployment.

Chapter 4, "Using the Connector" provides information about the tasks that must be performed each time you want to run reconciliation.

Chapter 5, "Extending the Functionality of the Connector" describes procedures that you can perform to extend the functionality of the connector.

Chapter 6, "Testing and Troubleshooting" provides information about testing the connector.

Chapter 7, "Known Issues and Workarounds" lists the known issues associated with this release of the connector.

Appendix A, "Sample WSDL for ACME Webservice" provides a sample WSDL for ACME Webservice used as a sample target webservice in this guide.
Preinstallation Steps

Preinstallation of the connector involves the following steps:

---

**Note:** In this guide, a target system that exposes webservice endpoint has been referred to as the **target system**. ACME Webservice is used as a sample target system to discuss the configurations and the connector objects.

---

- Section 2.1, "Prerequisites"
- Section 2.2, "Building the Connector Bundle"
- Section 2.3, "Creating a SOA Composite for the Target Webservice"
- Section 2.4, "Handling Faults"
- Section 2.5, "Deploying and Testing the Webservice SOA Composite"

### 2.1 Prerequisites

The following are the prerequisites for configuring the SOA composite with the connector webservice client and the target webservice:

- Knowledge of webservises, WSDLs, SOA composite, and BPEL Process components
  
The WSDL should be well-defined for the target webservice, exposing the schema details and the operations.

- Oracle JDeveloper 11g (11.1.1.9.0) with SOA Composite Editor extension, for configuring and wiring SOA composite with the connector

---

**Note:** Ensure the version of JDeveloper you are using is compatible with the SOA server. You can download JDeveloper from:


For information about downloading and installing the SOA Composite Editor extension, visit:

http://www.oracle.com/ocom/groups/public/@otn/documents/webcontent/156082.xml#oracle.sca.modeler

---

- XSL Transformations, for payload conversions
The complexity of the wiring depends on the target webservice. For example, Amazon webservice expects every SOAP request to be signed and the signature would change for every request. This signature has to be computed as part of the composite.

### 2.2 Building the Connector Bundle

The connector package contains a set of templates and a build utility. The build utility is a script that generates Oracle Identity Manager artifacts specific to the target webservice from the set of templates. It also generates SOA composite project that you can use to wire the connector client webservice against the target webservice.

**Note:** You can build a connector specific to different target webservices using the build utility. Cloning of this connector is not supported.

To build the connector:

1. Create a directory for the connector, for example, Webservices-11.1.1.5.0, in the `OIM_HOME/server/ConnectorDefaultDirectory` directory.
2. Copy and unzip the contents of the connector installation media directory into directory created in Step 1.

   Table 2–1 lists the files and directories on the installation media, before building the connector.

**Table 2–1 Files and Directories Before Building the Connector**

<table>
<thead>
<tr>
<th>File in the Installation Media Directory</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>build-connector.bat</td>
<td>Batch files for generating the webservice connector package from templates.</td>
</tr>
<tr>
<td>build-connector.sh</td>
<td></td>
</tr>
<tr>
<td>bundle/org.identityconnectors.webservices-1.0.112.jar</td>
<td>This JAR file is the ICF bundle that the connector is using for the current release.</td>
</tr>
<tr>
<td>configuration</td>
<td>This folder is empty.</td>
</tr>
<tr>
<td>javadoc</td>
<td>This directory contains information about the Java APIs used by the connector.</td>
</tr>
<tr>
<td>lib/ConnectorBuildTools.jar</td>
<td>This JAR file contains class files for generating the webservice connector package from templates.</td>
</tr>
</tbody>
</table>

Files in the resources directory

Each of these resource bundles contains language-specific information that is used by the connector. During connector deployment, this file is copied to the Oracle Identity Manager database.

**Note:** A resource bundle is a file containing localized versions of the text strings that are displayed on the Administrative and User Console. These text strings include GUI element labels and messages.
3. Run one of the following commands.

On Microsoft Windows:

```
build-connector.bat "LONG_CODE" "SHORT_CODE"
```

On UNIX:

```
sh build-connector.sh "LONG_CODE" "SHORT_CODE"
```

where `LONG_CODE` is the descriptive name of the connector and `SHORT_CODE` is the concise 4-character name of the target system that will be used in lookup names, adapter names, and so on.

For example, to build the connector zip file for the ACME Webservice on Linux, run the following command:

```
sh build-connector.sh "ACME Web" "ACME"
```

Table 2–2 lists the files and directories generated after building the connector.
Creating a SOA Composite for the Target Webservice

### Table 2–2  Files and Directories Generated After Building the Connector

<table>
<thead>
<tr>
<th>File in the Installation Media Directory</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>configuration/ACME-CI.xml</td>
<td>This file contains configuration information that is used during connector installation.</td>
</tr>
<tr>
<td>Files and directories in the soa/project/ACMEWebserviceWSCconnector directory:</td>
<td>These files and directories form SOA composite project. You can open this project in JDeveloper to wire the templates against the target webservice.</td>
</tr>
<tr>
<td>• ACMEWebserviceWSCconnector.jpr</td>
<td></td>
</tr>
<tr>
<td>• classes</td>
<td></td>
</tr>
<tr>
<td>• composite.xml</td>
<td></td>
</tr>
<tr>
<td>• SCA-INF</td>
<td></td>
</tr>
<tr>
<td>• testsuites</td>
<td></td>
</tr>
<tr>
<td>• WebservicesConnectorServiceWrapper.wsdI</td>
<td></td>
</tr>
<tr>
<td>• SConnector.bpel</td>
<td></td>
</tr>
<tr>
<td>• WSConnector.componentType</td>
<td></td>
</tr>
<tr>
<td>• wsdI</td>
<td></td>
</tr>
<tr>
<td>• xsd</td>
<td></td>
</tr>
<tr>
<td>• xsl</td>
<td></td>
</tr>
<tr>
<td>xml/ACME-ConnectorConfig.xml</td>
<td>This file contains definitions for the connector components.</td>
</tr>
<tr>
<td>• Resource object</td>
<td></td>
</tr>
<tr>
<td>• Process definition</td>
<td></td>
</tr>
<tr>
<td>• IT resource type</td>
<td></td>
</tr>
<tr>
<td>• Reconciliation rules</td>
<td></td>
</tr>
<tr>
<td>• Scheduled jobs</td>
<td></td>
</tr>
<tr>
<td>• Lookup definitions</td>
<td></td>
</tr>
</tbody>
</table>

### 2.3 Creating a SOA Composite for the Target Webservice

The connector uses SOA to connect to the target webservice and perform operations on them. The variables in the SOA composite must be mapped to the variables on the target system.

After building the connector as per Section 2.2, "Building the Connector Bundle," you can open the generated SOA composite project in JDeveloper to wire the templates against the target webservice. After completing the wiring of the SOA composite, you can configure SOA WebSecurity policies in the composite.xml file for authentication by including the specific policy and binding properties.

**Note:** As a best practice, you can configure how errors and faults are handled for each operation. See Section 2.4, "Handling Faults" for more information.

This section describes the following procedures:

- Section 2.3.1, "Configuring the Partner Link"
- Section 2.3.2, "Configuring the Create Operation"
- Section 2.3.3, "Configuring the Create Operation for SPML"
- Section 2.3.4, "Configuring the Delete Operation"
Creating a SOA Composite for the Target Webservice

Section 2.3.5, "Configuring the Update Operation"
Section 2.3.6, "Configuring the Enable and Disable Operations for Provisioning"
Section 2.3.7, "Configuring the Search Operation"
Section 2.3.8, "Configuring the Enable and Disable Operations for Reconciliation"
Section 2.3.9, "Configuring the Lookup Search Operation"
Section 2.3.10, "Configuring the Reset Password Operation"

2.3.1 Configuring the Partner Link
To configure the partner link before configuring the operations:

1. In JDeveloper, open the SOA composite project file, ACMEWebseviceWSConnector.jpr, located in the following directory:
   OIM_HOME/server/ConnectorDefaultDirectory/Webservices-11.1.1.5.0/soa/proj
   ect/ACMEWebseviceWSConnector

   **Note:** Ensure the version of JDeveloper you are using is compatible with the SOA server. SOA Composite Editor extension must also be installed. For information about downloading and installing version 11.1.1.9.0 of JDeveloper, visit:
   http://www.oracle.com/technetwork/developer-tools/jdev/downl
   oads/jdeveloper111190-2538883.html

2. Open the composite.xml file.
   This configuration file shows the relations of the BPEL Process and partner clients, as shown in the following sample screenshot.

3. Double-click the BPEL Process component to view the ICF operations.
   By default, this BPEL Process is wired to the connector WSDL, wsdI\WebservicesConnectorService.wsdI. If you double-click the WSDL, the
connector operations with their input and output schema and the exceptions are displayed.

**Note:** The WebserviceConnectorService schema (xsd/WebservicesConnectorService.xsd) and wsdl file (wsdl/WebservicesConnectorService.wsdl) in the JDeveloper SOA project should not be modified as this will break the contract with the Connector and it might not work as expected.

In the BPEL Process, each branch will be invoked based on the webservice connector operation that is called.

4. Include a partner link (http://<machine-name>:<port-number>/spml-xsd/SPMLService?WSDL) for an operation, such as Create, by importing and defining the target WSDL.
In the composite.xml file, the ACMEUserService is listed with all the operations.

5. Save the project.

2.3.2 Configuring the Create Operation

Deploy your target on SOA server.

After performing the procedure described in Section 2.3.1, "Configuring the Partner Link," you can configure the create operation in the SOA composite as follows:

See Also: Section 5.2.2, "Adding Custom Attributes for Provisioning in SOA Composite" for information about adding custom attributes for the Create operation
1. Include a partner link for the create operation by importing and defining the target WSDL.

In the composite.xml file, the ACMEUserService is listed with all the operations.

2. Invoke an operation on the user service by dragging InvokeCreate onto the ACMEUserService partner link.
For the create operation, you can invoke the CreateAccount operation and specify the input and the output variables.

Now, InvokeCreate has a call to the target webservice.

3. Map the input and the output variables to the target webservice target variables by editing the respective Assign activity, for example CreateAssignment, in the BPEL Process.
4. Right-click the mapping and select `ignoreMissingFromData` for all the non-mandatory fields.

5. Assign the Unique Id of the account returned by the target webservice to `createResponse` or `Uid` field before `CreateOpReply` activity. The returned value will be considered as the unique identifier of the user or account which is used to refer to the created object. Subsequent updates of this user's attributes will send this value along with the updated attributes to the webservice connector composite.

6. Save the assignment and the project.

7. To test the create operation, you can comment all the `onMessage` lines except the `CreateOp` line in the BPEL source file in the project.

### 2.3.3 Configuring the Create Operation for SPML

Service Provisioning Markup Language (SPML) is an XML-based framework based on the concepts of Directory Service Markup Language (DSML) for exchanging user, resource, and service provisioning information between cooperating organizations.

Before configuring create operation for SPML:
1. Use the SPML and DSML XML Schema Definition (XSD) files depending on the SPML version of the target webservice, for example, ACME Webservice. See Appendix D, "Sample XSDs" for the SPML and DSML XSDs.

   - To test against SPML, the target is already placed in server/apps/oim.ear and is auto deployed on its first access from oim_server1 of OIM. WSDL path: http://<machine-name>:14000/spml-xsd/SPMLService?WSDL. The operations of SPML are asynchronous, therefore OIM does not wait for the response from target.

   - To test against non-SPML target, use the target as attached. Deploy this target on SOA server.

2. Modify the XSD files to match the syntax expected by the target webservice.

3. Use the SPML WSDL to test the operations through a separate webservice testing tool such as SOAP UI. Ensure you can perform the operations using the testing tool.

   If a WSDL is not available by default for the SPML service, you can create your own WSDL using the following link:

   http://<machine-name>:<port-number>/spml-xsd/SPMLService?WSDL

4. In JDeveloper, create a partner link in the SOA composite using the WSDL, as per the procedure described in Section 2.3.1, "Configuring the Partner Link."

5. Retain the SPML and DSML XSD files in the project and ensure that the imports and references to these files are set up properly.

Consider the following sample CreateUser SPML request:

```xml
<?xml version='1.0' encoding='UTF-8'?>
  <SOAP-ENV:Body>
    <spml:Request xmlns:urn:oasis:names:tc:SPML:2.0:returnData='everything'>
      <data>
        <spml:attr xmlns:urn:oasis:names:tc:DSML:2.0:core='name' name='objectClass'>
          <spml:value>spml2Person</spml:value>
        </spml:attr>
        <spml:attr xmlns:urn:oasis:names:tc:DSML:2.0:core='name' name='accountID'>
          <spml:value>testuser10</spml:value>
        </spml:attr>
        <spml:attr xmlns:urn:oasis:names:tc:DSML:2.0:core='name' name='credentials'>
          <spml:value>password</spml:value>
        </spml:attr>
        <spml:attr xmlns:urn:oasis:names:tc:DSML:2.0:core='name' name='firstnames'>
          <spml:value>testuser101</spml:value>
        </spml:attr>
      </data>
    </spml:Request>
  </SOAP-ENV:Body>
</SOAP-ENV:Envelope>
```

To configure create operation for the sample CreateUser SPML request:

1. Edit the Invoke activity to call the CreateUser operation of the target and add the variables.
2. Drop a Transform activity before the Invoke activity. Remove unwanted Assign activity, if any. Set the source variable to `CreateOp_inputVariable` and the target variable to `InvokeCreate`, which is the Invoke activity input variable created in the previous step.

3. Provide a mapper file name and click OK. The file is opened for editing.

4. Loop over the `userAccount` attributes using a for-each construct.
5. Switch to the Source tab to see the XSL transform code for the attribute. Use `choose-when` construct to check for the attribute name and assign values.

The following is a sample code. In this sample, firstName is the attribute name that is received from the connector webservice. The name is transformed to firstname, which is the attribute name used by the SPML target as shown before Step 1 in the sample SPML request.

Similarly, the mappings for other attributes such as lastName can be done by adding when nodes.

```xml
<xsl:for-each select="/types:create/userAccount/**">
  <xsl:choose>
    <xsl:when test='(name() = "firstName") and /types:create/userAccount/firstName'>
      <dsml:attr>
        <xsl:attribute name="name">
          <xsl:text disable-output-escaping="no">firstname</xsl:text>
        </xsl:attribute>
        <dsml:value>
          <xsl:value-of select="/types:create/userAccount/firstName"/>
        </dsml:value>
      </dsml:attr>
    </xsl:when>
    <xsl:when test='name() = "lastName" and /types:create/userAccount/lastName'>
      ...
    </xsl:when>
  </xsl:choose>
</xsl:for-each>
```

6. The connector expects the UID of the user created on the target as output Reply. In the example, the following create response is received:

7. Another transformation is required to read accountId from the response and send it as UID in reply. Drop another Transform activity between the Invoke activity and the CreateOpReply activity.
8. Set the source variable as the output of Invoke and the target variable as output variable of Reply.

9. In the Transformation page:
   - Apply for-each loop on the dsml:attr.
   - Drop equals and if lines on the UID and link them, as shown in the following sample screenshot.
   - Assign dsml:value to UID of createResponse.

10. Double-click the equals box to edit it and set the second parameter as accountId.
11. The source of the transformation file will be as follows:

   `<types:createResponse>
     <xsl:for-each select="/spml:addResponse/spml:pso/spml:data/dsml:attr">
       <xsl:if test='@name = "accountId"'>
         <uid>
           <xsl:value-of select="dsml:value"/>
         </uid>
       </xsl:if>
     </xsl:for-each>
   </types:createResponse>`

12. Save the assignment and the project.

   You can compile and deploy the project. Test the operation from the Enterprise Manager. See Section 2.5, "Deploying and Testing the Webservice SOA Composite" for more information.

### 2.3.4 Configuring the Delete Operation

After performing the procedure described in Section 2.3.1, "Configuring the Partner Link," you can configure the delete operation in the SOA composite using the following procedure. The UID of the user to be deleted will be the input from Oracle Identity Manager to the SOA composite. This input has to be mapped to the Unique Id of the user to be deleted in the target webservice.

1. Link the InvokeDelete operation to the appropriate partner link for delete operation.
2. Specify the operation and the input/output variables for this Invoke activity and click OK.

3. Drag an Assign activity from the component palette before the Invoke activity.
4. Edit the Assign activity to map the input variables for the delete operation.

5. In the Edit Assign window, map the fields in DeleteOp_InputVariable to the corresponding fields in Input variable of the target operation.

6. After the variables are mapped, compile and deploy the project. Test the operation from the Enterprise Manager. See Section 2.5, "Deploying and Testing the Webservice SOA Composite" for more information.
2.3.5 Configuring the Update Operation

Before configuring the update operation in the SOA composite:

1. Verify if the target webservice supports simultaneous updates of multiple attributes.

2. If the target webservice supports update of only one attribute at a time, then remove the `FORM_NAME` Updated process task from the process definition in the Design Console.

   See Section 3.2.7, "Removing Bulk Attribute Update Task" for more information.

3. In JDeveloper, create a partner link in the SOA composite using the WSDL, as per the procedure described in Section 2.3.1, "Configuring the Partner Link."

Assuming that the target webservice supports simultaneous updates of multiple attributes, you can configure the update operation as follows:

1. Link the InvokeUpdate operation to the appropriate partner link for update operation.

2. Specify the operation and the input/output variables for this Invoke activity and click OK.
3. Drop a Transform activity to Invoke. Set the source variable as `UpdateOp_InputVariable` and the target variable as the input variable of the Invoke activity set in the previous step.

4. Open the translation mapper file. Drag the `if` construct to the target variable.

5. Map the name under `updatedAttribute` to the `if` construct and the value to the target variable.
6. Switch to source. The following is a sample source:

```
<xsl:if test="/types:update/updatedAttribute/name">
  <firstName>
    <xsl:value-of select="/types:update/updatedAttribute/value"/>
  </firstName>
</xsl:if>
```

7. Verify the Decode value for the connector field in the Lookup.ACME.UM.ProvAttrMap lookup definition.

For the example attribute name first name, the Decode value is FirstName.

8. Modify the source as follows:

```
<xsl:if test="/types:update/updatedAttribute/name = "FirstName"">
  <firstName>
    <xsl:value-of select="/types:update/updatedAttribute/updatedAttribute[name = 'FirstName']/value"/>
  </firstName>
</xsl:if>
```

9. Follow Steps 5 to 8 to add other attributes.

For custom attributes, ensure that the attributes are already included in the Lookup.ACME.UM.ProvAttrMap lookup definition. See Section 5.2.3, "Adding Custom Attribute for Update Operation" for more information.

10. If the target returns the UID of the user updated, drop an Assign activity after Invoke and map the return value to the updateResponse uid. Otherwise, map the uid from updateOp input variable to the updateResponse uid.
11. Save the project.
You can compile and deploy the project. Test the operation from the Enterprise Manager.

### 2.3.6 Configuring the Enable and Disable Operations for Provisioning

As a prerequisite, configure the update operation and create the transformation XSL file as described in Section 2.3.5, "Configuring the Update Operation." Consider the target variable Status that can have a value of Active or Inactive.

To configure the enable or disable operation for provisioning in the SOA composite:

1. Drop an if, choose, and when constructs on the target variable as shown in the following sample screenshot.

2. Drag name under updatedAttribute to the if and when constructs.

3. Right-click the target variable, Status, and set two Text values to Active and Inactive.
4. Switch to the Source tab to see the XSL transform code for the attribute. Update the code as follows:

```xml
<xsl:if test='/types:update/updatedAttribute[name = "__ENABLE__"]'>
  <xsl:choose>
    <xsl:when test='/types:update/updatedAttribute[name = "__ENABLE__"]/value = "true"'>
      <status>
        <xsl:text disable-output-escaping="no">Active</xsl:text>
      </status>
    </xsl:when>
    <xsl:when test='/types:update/updatedAttribute[name = "__ENABLE__"]/value = "false"'>
      <status>
        <xsl:text disable-output-escaping="no">Inactive</xsl:text>
      </status>
    </xsl:when>
  </xsl:choose>
</xsl:if>
```

5. Save the project.

You can compile and deploy the project. Test the operation from the Enterprise Manager. See Section 2.5, "Deploying and Testing the Webservice SOA Composite" for more information.

### 2.3.7 Configuring the Search Operation

The search branch is invoked when a trusted source or target resource user reconciliation scheduled job is run from Oracle Identity Manager. This operation will fetch a list of users and their attributes from the target webservice. The list is converted to a list of userSearchRecords that are returned to Oracle Identity Manager.

**See Also:** Section 5.3.2, "Adding Custom Attributes for Reconciliation in SOA Composite" for information about adding custom attributes for the Search operation

Section 5.6, "Mapping Timestamp Attribute" for information about converting timestamp attribute to long type, which is the type used in the connector

After performing the procedure described in Section 2.3.1, "Configuring the Partner Link," you can configure the search operation as follows:
1. Link the InvokeSearch operation to the appropriate partner link for search operation.

2. Specify the operation and the input/output variables for this Invoke activity and click OK.

3. Drag an Assign activity from the component palette before the InvokeSearch activity.
4. Edit the Assign activity to map the input variables for the search operation.

5. In the Edit Assign window, map the fields in SearchOp_InputVariable to the corresponding fields in Input variable of the target operation.

The batchStart and batchEnd parameters can be used to specify the batching/paging parameters that are supported by the target webservice. If the batchStart and batchEnd parameters are not mapped, then batching is disabled. See Section 4.1.3, "Batched Reconciliation" for more information.

The timestamp is used for incremental reconciliation. The user records with value of the attribute that is mapped to the timestamp variable and greater than the timestamp value are fetched from the target webservice. For example, if timestamp is mapped to the ModifiedDate attribute on the target webservice, then the search will fetch all the users whose ModifiedDate attribute is greater than the value specified in the timestamp variable.
6. After the variables are mapped, the output received from the target webservice needs to be converted to a convention that the connector understands (list of userSearchRecords). To do so, drag a Transform activity from the component palette and drop it after the SearchInvoke activity.
7. Edit the Transform activity.

8. In the Edit Transform window, add the source variable that has to be converted.
9. Select the source variable. It will be the output variable of the SearchInvoke activity.

10. Verify the source variable and click OK.
11. Specify the target variable. The target variable will be `SearchReply_OutputVariable`.

12. Specify the mapper file name. Click Apply.
13. An XSL file with the specified Mapper File name will be created and opened. The input variable will be on the left and the output structure will be on the right. Click **OK**.

14. Expand the source and the target variables to verify the structures before mapping.

15. Map the Search Output response appropriately. After the return variable is wired to the userSearchRecords, JDeveloper automatically maps the variables using the **AutoMap** feature.
16. The AutoMap feature automatically maps the source elements to similar names in the target webservice and includes a **for-each** statement before userSearchRecords for fetching a list of userRecords. If this is not accurate, map and transform the mappings manually. You can also switch to the Source tab and update the XSL transform code directly.

17. Click **OK**. The following is a sample screenshot of the transformation mappings:

18. Expand the nodes and verify if the transformation mappings are appropriate.
19. After verifying the transformation mappings, save the project.

20. Build and deploy the SOA composite. Test the search operation from Enterprise Manager. See Section 2.5, "Deploying and Testing the Webservice SOA Composite" for more information.

**Note:** If the name and Uid attributes in the target system schema hold the same values, then ensure that the login field in the connector schema is mapped either to the name or Uid attribute. In the Transform after UserInvokeSearch, map the login field from the target wsdl to the login field in WebserviceConnectorService.wsdl.

If the name and Uid attributes hold different values, then map the login attribute in the WSConnector SOA Composite to the Login attribute in the webservice process form of a user. See Section 5.3.3, "Adding Custom Attributes for Reconciling _UID_ Field" for more information.

If name and Uid attributes are same or different, it is mandatory for the login field to be mapped as internally the connector uses this value to set the Uid attributes and name of the connector object.

**2.3.7.1 Mapping Simple Child Table Values in the SOA Composite**

You can map simple child table values in the SOA Composite. To do so, in the search transform, perform the following mapping for the child table values:

Add a "for each" loop for the child table value and map the child table value to the "values" attribute in the multivaluedAttribute.

For example, in this case, "Group" is the child table, and each group is mapped to the "values" element in the multivaluedAttribute.
This mapping will have the following values:

```xml
<multivaluedAttributes>
  <name>
    <xsl:text disable-output-escaping="no">Group</xsl:text>
  </name>
  <xsl:for-each select="tns:Groups/group">
    <values>
      <xsl:value-of select="."/>
    </values>
  </xsl:for-each>
</multivaluedAttributes>
```

### 2.3.8 Configuring the Enable and Disable Operations for Reconciliation

As a prerequisite, configure the search operation and create the transformation XSL file as described in Section 2.3.7, "Configuring the Search Operation." Consider the target variable Status that can have a value of Active or Inactive.

To configure the enable or disable operation for reconciliation in the SOA composite:

1. For reconciliation of the Status attribute, populate the otherAttributes named `__ENABLE__`.

For example, if the status of the target user is either Active or Inactive, the following XSL code can be used for mapping the Status attribute:

```xml
<otherAttributes>
  <name>
    <xsl:text disable-output-escaping="no">__ENABLE__</xsl:text>
  </name>
  <xsl:choose>
    <xsl:when test='userAccount/status = "Active"'>
      <value>
        <xsl:text disable-output-escaping="no">true</xsl:text>
      </value>
    </xsl:when>
    <xsl:when test='userAccount/status = "Inactive"'>
      <value>
        <xsl:text disable-output-escaping="no">false</xsl:text>
      </value>
    </xsl:when>
  </xsl:choose>
</otherAttributes>
```

The following is a sample screenshot of the Source tab:
2. The object status is reflected in Oracle Identity Manager as either Enabled or Disabled.

3. Save the project.

You can compile and deploy the project. Test the operation from the Enterprise Manager. See Section 2.5, "Deploying and Testing the Webservice SOA Composite" for more information.

2.3.9 Configuring the Lookup Search Operation

The lookup search branch is invoked when the webservice connector lookup scheduled job is run from Oracle Identity Manager. The lookup search operation accepts objectClass as input that is passed as scheduled task parameter and returns a list of lookupEntries, which is a list of name, value pair. The list of names and values in the output will be set as the Decode and Code Key values of the lookup definition respectively.

After performing the procedure described in Section 2.3.1, "Configuring the Partner Link," you can configure the lookup search operation as follows:

---

**Note:** Entries in the lookup definition, process form, reconciliation field mappings, and profile need not be added for the _ENABLE_ attribute. They are configured by default.
1. Link the InvokeLookupSearch operation to the appropriate partner link for the lookup search operation.

2. Specify the operation and the input/output variables for the InvokeLookupSearch activity and click OK.

3. The output received from the target webservice needs to be converted to a convention that the connector understands (list of name, value pairs). To do so, drag a Transform activity from the component palette and drop it after the InvokeLookupSearch activity.
4. Edit the Transform activity.

5. In the Edit Transform window, add the source and target variables. Then, specify the mapper file name and click OK.
6. An XSL file with the specified Mapper File name will be created and opened. The input variable will be on the left and the output structure will be on the right. Click **OK**.

7. Expand the source and the target variables to verify the structures before mapping.

8. Map the Lookup Search response appropriately. After the return variable is wired to the lookupEntries, JDeveloper automatically maps the variables using the AutoMap feature.

The AutoMap feature automatically maps the source elements to similar names in the target webservice and includes a **for-each** statement before lookupEntries for fetching a list of roles. If this is not accurate, map and transform the mappings manually. You can also switch to the Source tab and update the XSL transform code directly. Click **OK**.

9. After verifying the transformation mappings, save the project.

10. In Oracle Identity Manager Design Console, create an empty lookup definition that will be populated as a result of the lookup search scheduled task.

11. Configure the scheduled task accordingly. Specify the lookup name and the ObjectType of the lookup.
The lookupSearch output will be transformed as follows:

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>string</td>
<td>lookupVal1</td>
</tr>
<tr>
<td>name</td>
<td>string</td>
<td>lookupVal1</td>
</tr>
<tr>
<td>value</td>
<td>string</td>
<td>lookupVal1</td>
</tr>
<tr>
<td></td>
<td>string</td>
<td>lookupVal2</td>
</tr>
<tr>
<td>name</td>
<td>string</td>
<td>lookupVal2</td>
</tr>
<tr>
<td>value</td>
<td>string</td>
<td>lookupVal2</td>
</tr>
<tr>
<td></td>
<td>string</td>
<td>lookupVal3</td>
</tr>
<tr>
<td>name</td>
<td>string</td>
<td>lookupVal3</td>
</tr>
<tr>
<td>value</td>
<td>string</td>
<td>lookupVal3</td>
</tr>
<tr>
<td></td>
<td>string</td>
<td>lookupVal4</td>
</tr>
<tr>
<td>name</td>
<td>string</td>
<td>lookupVal4</td>
</tr>
<tr>
<td>value</td>
<td>string</td>
<td>lookupVal4</td>
</tr>
</tbody>
</table>

After running the scheduled job, the lookup definition will be populated as follows:

12. Build and deploy the SOA composite. Test the operation from Enterprise Manager. See Section 2.5, “Deploying and Testing the Webservice SOA Composite” for more information.
2.3.10 Configuring the Reset Password Operation

This section describes how to configure password reset operations from the SOA composite. After the mappings are configured, a custom outbound policy will be attached to decrypt the password fields. Sensitive fields that are sent from Oracle Identity Manager are encrypted. The outbound policy also ensures that the password fields do not appear in clear text in the SOAP payloads in Enterprise Manager.

After performing the procedure described in Section 2.3.1, "Configuring the Partner Link," you can configure the reset password operation as follows:

1. Link the InvokeResetPassword operation to the appropriate partner link for reset password operation.

2. Specify the operation and the input/output variables for this Invoke activity and click OK.

3. Drag an Assign activity from the component palette before the Invoke activity.

4. Edit the ResetPasswordAssign activity to map the input variables for the reset password operation.
5. By default, the uid field of ResetPasswordOp_InputVariable is mapped to the uid field of ResetPasswordOp_OutputVariable.

6. In the Edit Assign window, map the fields in ResetPasswordOp_InputVariable to the target webservice payload for the new password that has to be updated.

7. Map the uid field from ResetPasswordOp_InputVariable to the corresponding Unique Id field in the target webservice.
8. After the variables are mapped, configure the custom outbound policy. See Section 5.1.1, "Handling Passwords" for information about this procedure.

9. Specify the `password.field.xpath.locations` property in the composite. This property can be obtained from the ResetPasswordAssign activity.

10. Specify the `target.payload.namespaces` property. This property should be the corresponding namespace of the password field that is available in BPEL source.

   **Note:** Ensure that the namespace in the `target.payload.namespaces` property does not include quotation marks.
11. Build and deploy the SOA composite. Test the operation from Enterprise Manager. See Section 2.5, "Deploying and Testing the Webservice SOA Composite" for more information.

### 2.4 Handling Faults

Fault handling is an important aspect of configuring the SOA composite. In the case of any faults and errors, a correct response must be provided to the connector and to Oracle Identity Manager from the target webservice. This should be configured at the SOA composite level as the remote fault thrown by the target webservice operation has to be mapped against the corresponding connector-specific faults.

The following table lists the faults defined in the connector webservice (WebserviceConnectorService) WSDL:

<table>
<thead>
<tr>
<th>Fault</th>
<th>Description</th>
<th>Operations that can throw this fault</th>
</tr>
</thead>
<tbody>
<tr>
<td>AlreadyExistsException</td>
<td>An account already exists in the target webservice.</td>
<td>Create</td>
</tr>
<tr>
<td>UnknownUidException</td>
<td>The passed unique ID is invalid or does not exist in the target webservice.</td>
<td>All operations except Create</td>
</tr>
<tr>
<td>ConnectionBrokenException</td>
<td>The target webservice endpoint is not reachable.</td>
<td>All operations</td>
</tr>
<tr>
<td>ConnectorException</td>
<td>Any other fault.</td>
<td>All operations</td>
</tr>
</tbody>
</table>

To configure fault handling in the SOA composite for the Create operation:

**See Also:** Section 2.4.1, "Handling Faults with Catch Blocks" for information about handling faults for the target webservice operations that throw faults instead of sending responses

1. Drag a Switch activity from the Component Palette and drop it after the InvokeCreate activity.
The following is a sample screenshot after dragging the Switch activity:

2. Specify the **Success** condition in the first branch. Click the Expression Builder icon to specify the condition.
3. Enter a valid expression for the Success condition that is based on the return output variable.
   You can browse for the variable under BPEL Variables pane and select the appropriate field.

4. Enter a valid expression in the Expression Builder pane.
   The following sample screenshot uses a **contains** function:

5. Verify the condition and click **OK**.
6. Drag the CreateOpReply node into the Success branch.

7. You can add multiple Switch branches to define error conditions and throw the corresponding faults. As an example, a fault branch for the AlreadyExistsException fault (account already exists) will be added.
8. Specify the condition for “account already exists” condition.

9. Drag a Throw activity from the Component Palette and drop it under the AccountExists branch.

10. Configure the Throw activity.
11. Select the `AlreadyExistsException` fault from the `WebserviceConnectorService WSDL`.

![Fault Chooser Image]

12. Create a Fault Variable of the fault type.

![Create Variable Image]

13. Click **Apply** and **OK**.
14. Drag an Assign activity above the Throw activity to assign the fault message.

The following is a sample screenshot of the AlreadyExistsException branch after configuration:
15. You can catch the default exceptions in Otherwise branch and throw the generic `ConnectorException` fault in the `WebserviceConnectorService` WSDL.

The following is a sample screenshot of the complete fault handling:
2.4.1 Handling Faults with Catch Blocks

If the target webservice operations throw faults instead of sending responses, fault handling can be configured by adding a Catch block in the SOA composite. The procedure for the Delete operation is as follows:

1. Drag the Scope activity into the BPEL process to scope the Invoke operation.

2. Add a Catch statement by clicking the alarm icon in the scope context.

You can configure the Catch block by specifying the type of fault thrown by the target webservice operation.
3. Add a **Throw** activity in the Catch block to throw the corresponding connector-specific fault type.

   You can refer to the previous procedure (Step 10 onward) for information about configuring a Throw activity.
2.5 Deploying and Testing the Webservice SOA Composite

After the SOA composite is ready, you can build and deploy it to SOA server using JDeveloper. To perform this procedure:

1. (Optional) If you need to pass sensitive data from Oracle Identity Manager to the composite and in turn to the target webservice, you can configure outbound policy as described in Section 5.1, "Securing the Connector."

2. Deploy the configured SOA composite on the SOA Server, as shown in the following sample screenshot.

3. Test the composite from Enterprise Manager by visiting the URL (for example, http://adminhost:adminport/em) or from the Enterprise Manager console.
Deploying the Connector involves the following steps:

- Section 3.1, "Installation"
- Section 3.2, "Postinstallation"

## 3.1 Installation

Note: In this guide, the term **Connector Installer** has been used to refer to the Connector Installer feature of the Oracle Identity Manager Administrative and User Console.

Installing the connector on a Connector Server is not supported.

Installation on Oracle Identity Manager consists of the following procedures:

- Section 3.1.1, "Running the Connector Installer"
- Section 3.1.2, "Configuring the IT Resource"

### 3.1.1 Running the Connector Installer

Note: In this guide, the term **Connector Installer** has been used to refer to the Connector Installer feature of the Administrative and User Console.

To run the Connector Installer:

1. Depending on the Oracle Identity Manager release you are using, perform one of the following steps:
   - For Oracle Identity Manager release 11.1.1:
     a. Log in to the Administrative and User Console.
b. On the Welcome to Identity Manager Advanced Administration page, in the System Management region, click Manage Connector.

- For Oracle Identity Manager release 11.1.2.x:
  a. Log in to Oracle Identity System Administration.
  b. In the left pane, under System Management, click Manage Connector.

2. In the Manage Connector page, click Install.

3. From the Connector List list, select ACME Webservice 11.1.1.5.0. This list displays the names and release numbers of connectors whose installation files you copy into the default connector installation directory in Step 1.

   The name of the connector is derived from the LONG_CODE provided when building the connector in the preinstallation steps.

   If you have copied the installation files into a different directory, then:
   a. In the Alternative Directory field, enter the full path and name of that directory.
   b. To repopulate the list of connectors in the Connector List list, click Refresh.
   c. From the Connector List list, select ACME Webservice 11.1.1.5.0.

4. Click Load.

5. To start the installation process, click Continue.

   The following tasks are performed, in sequence:
   a. Configuration of connector libraries
   b. Import of the connector XML files (by using the Deployment Manager)
   c. Compilation of adapter definitions

   On successful completion of a task, a check mark is displayed for the task. If a task fails, then an X mark and a message stating the reason for failure is displayed. Depending on the reason for the failure, make the required correction and then perform one of the following steps:
   - Retry the installation by clicking Retry.
   - Cancel the installation and begin again from Step 1.

6. If all three tasks of the connector installation process are successful, then a message indicating successful installation is displayed. In addition, a list of steps that you must perform after the installation is displayed. These steps are as follows:
   a. Configuring the IT resource for the connector
      See Section 3.1.2, "Configuring the IT Resource" for more information.
   b. Configuring the scheduled tasks
      See Section 4.2.4, "Configuring Scheduled Jobs" for more information.

When you run the Connector Installer, it copies the connector files and external code files to destination directories on the Oracle Identity Manager host computer.
3.1.2 Configuring the IT Resource

Note: If you have configured your target system as a trusted source, then create an IT resource of type WEBSERVICES. For example, Webservices Trusted. The parameters of this IT resource are the same as the parameters of the IT resources described in Table 3–1 of this section. See Creating IT Resources in Oracle Fusion Middleware Administering Oracle Identity Manager for more information about creating an IT resource.

The IT resource for the target system contains details of the SOA server where the webservice composite is deployed. Oracle Identity Manager uses this information during reconciliation.

When you run the Connector Installer, the ACME Webservice Server IT resource is automatically created in Oracle Identity Manager. As an example, ACME Webservice is the name of the target system that exposes webservice endpoint. You can specify values for the parameters of this IT resource as follows:

1. Depending on the Oracle Identity Manager release you are using, perform one of the following steps:
   - For Oracle Identity Manager release 11.1.1:
     Log in to the Administrative and User Console
   - For Oracle Identity Manager release 11.1.2.x:
     Log in to Oracle Identity System Administration

2. If you are using Oracle Identity Manager release 11.1.1, then:
   a. On the Welcome page, click Advanced in the upper-right corner of the page.
   b. On the Welcome to Oracle Identity Manager Advanced Administration page, in the Configuration region, click Manage IT Resource.

3. If you are using Oracle Identity Manager release 11.1.2.x, create a sandbox as follows:

   See Also: Managing Sandboxes in Oracle Fusion Middleware Developing and Customizing Applications for Oracle Identity Manager for more information about application instance and sandbox

   a. On the upper navigation bar, click Sandboxes. The Manage Sandboxes page is displayed.
   b. On the toolbar, click Create Sandbox. The Create Sandbox dialog box is displayed.
   c. In the Sandbox Name field, enter a name for the sandbox. This is a mandatory field.
   d. In the Sandbox Description field, enter a description of the sandbox. This is an optional field.
   e. Click Save and Close. A message is displayed with the sandbox name and creation label.
   f. Click OK. The sandbox is displayed in the Available Sandboxes section of the Manage Sandboxes page.
g. Select the sandbox that you created.

h. On the toolbar, click **Activate Sandbox**.
   The table refreshes and a marker in the Active column is displayed. In addition, the Sandboxes link on the upper navigation bar also displays the active sandbox name in parentheses.

i. In the left pane, under Configuration, click **Application Instances**. The Application Instances page is displayed.

j. From the Actions menu, select **Create**. Alternatively, click **Create** on the toolbar. The Create Application Instance page is displayed.

k. Enter the values of the attributes. For example:
   - **Name**: ACMEInstance
   - **Display Name**: ACMEInstance
   - **Resource Object**: ACME Webservice
   - **IT Resource Instance**: ACME Webservice Server

l. Click **Save**. The application instance is created, and the details of the application instance is displayed in a page.

m. To create a form to be associated with the application instance, open the Create Application Instance page or the Attributes tab of the Application Instance details page.

n. Adjacent to the Forms field, click **Create**. The Create Form page is displayed.

o. Enter values for the form attributes. For example:
   - **Resource Type**: ACME Webservice
   - **Form Name**: ACME Form

p. Click **Create**. A message is displayed stating that the form is created.

q. In the Create Application Instance page or the Attributes tab of the Application Instance details page, click **Refresh** adjacent to the Form field. The newly created form is available for selection in the Form list.

r. Select the new form from the drop-down list and click **Save**.
   The application instance is created.

s. Before publishing the sandbox, close all the open tabs and pages.

t. From the table showing the available sandboxes in the Manage Sandboxes page, select the sandbox that you created.

u. On the toolbar, click **Publish Sandbox**. A message is displayed asking for confirmation.

v. Click **Yes** to confirm. The sandbox is published and the customizations it contained are merged with the main line.

w. In the left pane, under Configuration, click **IT Resource**.

   4. In the IT Resource Name field on the Manage IT Resource page, enter **ACME Webservice Server** and then click **Search**. Alternatively, from the IT Resource Type menu, select **ACME Webservice Server**, and then click **Search**.

   5. Click the edit icon for the IT resource.

   6. From the list at the top of the page, select **Details and Parameters**.
7. Specify values for the parameters discussed in Table 3–1. The remaining parameters of IT resource are not applicable for this connector.

**Table 3–1 IT Resource Parameters**

<table>
<thead>
<tr>
<th>Code Key</th>
<th>Decode</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Configuration lookup</td>
<td>Lookup.ACME.Configuration</td>
<td>Name of the lookup definition that contains configuration information.</td>
</tr>
<tr>
<td></td>
<td></td>
<td><strong>Note:</strong> You must not change the value of this parameter. However, if you create a copy of all the connector objects, then you can specify the unique name of the copy of this lookup definition as the value of the Configuration Lookup Name parameter in the copy of the IT resource.</td>
</tr>
<tr>
<td>passcode</td>
<td></td>
<td>Encryption key that will be used for encrypting passwords and other sensitive information that is passed to the SOA composite.</td>
</tr>
<tr>
<td></td>
<td></td>
<td><strong>Note:</strong> This field is mandatory. See &quot;Guidelines for Passcode&quot; for more information. Section 5.1.1, &quot;Handling Passwords&quot; describes the procedure for configuring decryption.</td>
</tr>
<tr>
<td>securityPolicies</td>
<td>oracle/wss_username_toke n_client_policy</td>
<td>OWSM security policy used to authenticate the webservice client endpoint.</td>
</tr>
<tr>
<td>soaServiceWSDL</td>
<td><a href="http://soa-host:soa-port">http://soa-host:soa-port</a></td>
<td>WSDL URL of the connector webservice that invokes the target webservice operations. You can copy the complete URL from the WSDL text box field in the SOA composite testing page from Enterprise Manager.</td>
</tr>
<tr>
<td>soaUserName</td>
<td>weblogic</td>
<td>SOA server user name where the SOA composite is deployed.</td>
</tr>
<tr>
<td>soaUserPassword</td>
<td></td>
<td>Password of the SOA server user entered in previous field.</td>
</tr>
<tr>
<td>WSS_CSF_KEY</td>
<td></td>
<td>CSF key of SOA server credentials. This field is optional.</td>
</tr>
</tbody>
</table>

The following is a screenshot of the View IT Resource Details and Parameters page. The screenshot displays sample values for the parameters of the IT resource.
8. To save the values, click **Update**.

### 3.2 Postinstallation

Postinstallation information is divided across the following sections:

- **Section 3.2.1, "Configuring Oracle Identity Manager 11.1.2 or Later"**
- **Section 3.2.2, "Enabling Logging"**
- **Section 3.2.3, "Setting up the Lookup Definition for Connection Pooling"**
- **Section 3.2.4, "Changing to the Required Input Locale"**
- **Section 3.2.5, "Clearing Content Related to Connector Resource Bundles from the Server Cache"**
- **Section 3.2.6, "Disabling Child Tables"**
- **Section 3.2.7, "Removing Bulk Attribute Update Task"**
- **Section 3.2.8, "Localizing Field Labels in UI Forms"**

### 3.2.1 Configuring Oracle Identity Manager 11.1.2 or Later

If you are using Oracle Identity Manager release 11.1.2 or later, you must create additional metadata such as a UI form and an application instance. In addition, you must run entitlement and catalog synchronization jobs. These procedures are described in the following sections:

- **Section 3.2.1.1, "Creating and Activating a Sandbox"**
- **Section 3.2.1.2, "Creating a New UI Form"**
- **Section 3.2.1.3, "Creating an Application Instance"**
- **Section 3.2.1.4, "Publishing a Sandbox"**
- **Section 3.2.1.5, "Harvesting Entitlements and Sync Catalog"**
- **Section 3.2.1.6, "Updating an Existing Application Instance with a New Form"**

#### 3.2.1.1 Creating and Activating a Sandbox

Create and activate a sandbox as follows. For detailed instructions, see Managing Sandboxes in *Oracle Fusion Middleware Developing and Customizing Applications for Oracle Identity Manager*.

1. On the upper navigation bar, click **Sandboxes**. The Manage Sandboxes page is displayed.
2. On the toolbar, click **Create Sandbox**. The Create Sandbox dialog box is displayed.
3. In the Sandbox Name field, enter a name for the sandbox. This is a mandatory field.
4. In the Sandbox Description field, enter a description of the sandbox. This is an optional field.
5. Click **Save and Close**. A message is displayed with the sandbox name and creation label.
6. Click **OK**. The sandbox is displayed in the Available Sandboxes section of the Manage Sandboxes page.
7. Select the sandbox that you created.
8. From the table showing the available sandboxes in the Manage Sandboxes page, select the newly created sandbox that you want to activate.

9. On the toolbar, click **Activate Sandbox**.
   
The sandbox is activated.

### 3.2.1.2 Creating a New UI Form

Create a new UI form as follows. For detailed instructions, see Managing Forms in *Oracle Fusion Middleware Administering Oracle Identity Manager*.

1. In the left pane, under Configuration, click **Form Designer**.
2. Under Search Results, click **Create**.
3. Select the resource type for which you want to create the form.
4. Enter a form name and click **Create**.

### 3.2.1.3 Creating an Application Instance

Create an application instance as follows. For detailed instructions, see Managing Application Instances in *Oracle Fusion Middleware Administering Oracle Identity Manager*.

1. In the System Administration page, under Configuration in the left pane, click **Application Instances**.
2. Under Search Results, click **Create**.
3. Enter appropriate values for the fields displayed on the Attributes form and click **Save**.
4. In the Form drop-down list, select the newly created form and click **Apply**.
5. Publish the application instance for a particular organization.

### 3.2.1.4 Publishing a Sandbox

To publish the sandbox that you created in Section 3.2.1.1, "Creating and Activating a Sandbox":

1. Close all the open tabs and pages.
2. From the table showing the available sandboxes in the Manage Sandboxes page, select the sandbox that you created in Section 3.2.1.1, "Creating and Activating a Sandbox."
3. On the toolbar, click **Publish Sandbox**. A message is displayed asking for confirmation.
4. Click **Yes** to confirm. The sandbox is published and the customizations it contained are merged with the main line.

### 3.2.1.5 Harvesting Entitlements and Sync Catalog

To harvest entitlements and sync catalog:

1. Run the scheduled jobs for lookup field synchronization listed in Section 4.2.1, "Scheduled Task for Lookup Field Synchronization."
2. Run the Entitlement List scheduled job to populate Entitlement Assignment schema from child process form table.
3. Run the Catalog Synchronization Job scheduled job. See Predefined Scheduled Tasks in Oracle Fusion Middleware Administering Oracle Identity Manager for more information about this scheduled job.

### 3.2.1.6 Updating an Existing Application Instance with a New Form

For any changes you do in the Form Designer, you must create a new UI form and update the changes in an application instance. To update an existing application instance with a new form:

1. Create a sandbox and activate it as described in Section 3.2.1.1, "Creating and Activating a Sandbox."
2. Create a new UI form for the resource as described in Section 3.2.1.2, "Creating a New UI Form."
3. Open the existing application instance.
4. In the Form field, select the new UI form that you created.
5. Save the application instance.
6. Publish the sandbox as described in Section 3.2.1.4, "Publishing a Sandbox."

### 3.2.2 Enabling Logging

Oracle Identity Manager uses Oracle Java Diagnostic Logging (OJDL) for logging. OJDL is based on java.util.logger. To specify the type of event for which you want logging to take place, you can set the log level to one of the following:

- **SEVERE.intValue()+100**
  This level enables logging of information about fatal errors.
- **SEVERE**
  This level enables logging of information about errors that may allow Oracle Identity Manager to continue running.
- **WARNING**
  This level enables logging of information about potentially harmful situations.
- **INFO**
  This level enables logging of messages that highlight the progress of the application.
- **CONFIG**
  This level enables logging of information about fine-grained events that are useful for debugging.
- **FINE, FINER, FINEST**
  These levels enable logging of information about fine-grained events, where FINEST logs information about all events.

These message types are mapped to ODL message type and level combinations as shown in Table 3–2.

<table>
<thead>
<tr>
<th>Java Level</th>
<th>ODL Message Type:Level</th>
</tr>
</thead>
<tbody>
<tr>
<td>SEVERE.intValue()+100</td>
<td>INCIDENT_ERROR:1</td>
</tr>
</tbody>
</table>
The configuration file for OJDL is logging.xml, which is located at the following path:

`DOMAIN_HOME/config/fmwconfig/servers/OIM_SERVER/logging.xml`

Here, `DOMAIN_HOME` and `OIM_SERVER` are the domain name and server name specified during the installation of Oracle Identity Manager.

To enable logging in Oracle WebLogic Server:

1. Edit the logging.xml file as follows:
   a. Add the following blocks in the file:

   ```xml
   <log_handler name='webservice-handler' level='[LOG_LEVEL]'
               class='oracle.core.ojdl.logging.ODLHandlerFactory'>
      <property name='logreader:' value='off'/>
      <property name='path' value='[FILE_NAME]' />
      <property name='format' value='ODL-Text'/>
      <property name='useThreadName' value='true'/>
      <property name='locale' value='en'/>
      <property name='maxFileSize' value='5242880'/>
      <property name='maxLogSize' value='52428800'/>
      <property name='encoding' value='UTF-8'/>
   </log_handler>

   <logger name="ORG.IDENTITYCONNECTORS.WEBSERVICES" level='{[LOG_LEVEL]'}
      useParentHandlers="false">
      <handler name="webservice-handler"/>
      <handler name="console-handler"/>
   </logger>
   ```

   b. Replace all occurrences of `[LOG_LEVEL]` with the ODL message type and level combination that you require. Table 3–2 lists the supported message type and level combinations.

   Similarly, replace `[FILE_NAME]` with the full path and name of the log file in which you want log messages to be recorded.

   The following blocks show sample values for `[LOG_LEVEL]` and `[FILE_NAME]`:

   ```xml
   <log_handler name='webservice-handler' level='NOTIFICATION:1'
               class='oracle.core.ojdl.logging.ODLHandlerFactory'>
      <property name='logreader:' value='off'/>
      <property name='path' value='F:\MyMachine\middleware\user_projects\domains\base_domain1\servers\oim_server1\logs\oim_server1-diagnostic-1.log'/>
      <property name='format' value='ODL-Text'/>
   </log_handler>
   ```
With these sample values, when you use Oracle Identity Manager, all messages generated for this connector that are of a log level equal to or higher than the NOTIFICATION:1 level are recorded in the specified file.

**Note:** The logging level for console-handler must be as fine as the level set in the loggers.

For example, if the NOTIFICATION:1 level is specified in the ORG.IDENTITYCONNECTORS.WEBSERVICES logger, and the console-handler has ERROR:1 level, then only logs at ERROR:1 or coarser levels would be available.

2. Click **Save** and close the file.
3. Restart the application server.

### 3.2.3 Setting up the Lookup Definition for Connection Pooling

By default, this connector uses the ICF connection pooling. **Table 3–3** lists the connection pooling properties, their description, and default values set in ICF:

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Pool Max Idle</td>
<td>Maximum number of idle objects in a pool.</td>
</tr>
<tr>
<td></td>
<td>Default value: 10</td>
</tr>
<tr>
<td>Pool Max Size</td>
<td>Maximum number of connections that the pool can create.</td>
</tr>
<tr>
<td></td>
<td>Default value: 10</td>
</tr>
<tr>
<td>Pool Max Wait</td>
<td>Maximum time, in milliseconds, the pool must wait for a free object to make itself available to be consumed for an operation.</td>
</tr>
<tr>
<td></td>
<td>Default value: 150000</td>
</tr>
<tr>
<td>Pool Min Evict Idle Time</td>
<td>Minimum time, in milliseconds, the connector must wait before evicting an idle object.</td>
</tr>
<tr>
<td></td>
<td>Default value: 120000</td>
</tr>
<tr>
<td>Pool Min Idle</td>
<td>Minimum number of idle objects in a pool.</td>
</tr>
<tr>
<td></td>
<td>Default value: 1</td>
</tr>
</tbody>
</table>

If you want to modify the connection pooling properties to use values that suit requirements in your environment, then:

1. Log in to the Design Console.
2. Expand **Administration**, and then double-click **Lookup Definition**.

3. Search for and open the configuration lookup definition.
   For example, **Lookup.ACME.Configuration**.

4. On the Lookup Code Information tab, click **Add**.
   A new row is added.

5. In the **Code Key** column of the new row, enter **Pool Max Idle**.

6. In the **Decode** column of the new row, enter a value corresponding to the Pool Max Idle property.

7. Repeat Steps 4 through 6 for adding each of the connection pooling properties listed in **Table 3–3**.

8. Click **Save**.

### 3.2.4 Changing to the Required Input Locale

Changing to the required input locale (language and country setting) involves installing the required fonts and setting the required input locale.

You may require the assistance of the system administrator to change to the required input locale.

### 3.2.5 Clearing Content Related to Connector Resource Bundles from the Server Cache

When you deploy the connector, the resource bundles are copied from the resources directory on the installation media into the Oracle Identity Manager database. Whenever you add a new resource bundle to the connectorResources directory or make a change in an existing resource bundle, you must clear content related to connector resource bundles from the server cache.

To clear content related to connector resource bundles from the server cache:

1. In a command window, switch to the **OIM_HOME/server/bin** directory.

2. Enter one of the following commands:

   ```
   Note: You can use the PurgeCache utility to purge the cache for any content category. Run `PurgeCache.bat CATEGORY_NAME` on Microsoft Windows or `PurgeCache.sh CATEGORY_NAME` on UNIX. The CATEGORY_NAME argument represents the name of the content category that must be purged.
   
   For example, the following commands purge Metadata entries from the server cache:
   
   ```PurgeCache.bat MetaData```
   ```PurgeCache.sh MetaData```
   ```
   ```
   ```
   
   On Microsoft Windows: `PurgeCache.bat All`
   
   On UNIX: `PurgeCache.sh All`
   
   When prompted, enter the user name and password of an account belonging to the SYSTEM ADMINISTRATORS group. In addition, you are prompted to enter the service URL in the following format:
Postinstallation

\[
t3://OIM\_HOST\_NAME:OIM\_PORT\_NUMBER
\]

In this format:
- Replace \textit{OIM\_HOST\_NAME} with the host name or IP address of the Oracle Identity Manager host computer.
- Replace \textit{OIM\_PORT\_NUMBER} with the port on which Oracle Identity Manager is listening.

### 3.2.6 Disabling Child Tables

Some target systems do not support multivalued attributes. For such target systems, disable the corresponding child table of the process form in Oracle Identity Manager Design Console. The connector includes a default multivalued attribute called Role, which is stored in child tables in Oracle Identity Manager. You can disable the child table if your target system does not support multivalued attributes.

To disable a child table of a process form:

1. Log in to the Oracle Identity Manager Design Console.
2. Expand Development Tools.
3. Double-click Form Designer.
4. Search for and open the parent process form, such as UD\_ACME\_USR.
5. Click Create New Version.
   - On the Create a new version dialog box, enter a new version in the Label field, and then click Save.
6. Click the Child Tables tab and delete the child table.
7. Click the Save.
8. Click Make Version Active to activate the newly created form.

### 3.2.7 Removing Bulk Attribute Update Task

Some target systems do not support bulk update of attributes. For such target systems, remove the corresponding adapter task from the process definition in Oracle Identity Manager Design Console.

The process task will be in the name of FORM\_NAME Updated. In the case of the ACME webservice, remove the UD\_ACME\_USR Updated task from the process definition.

### 3.2.8 Localizing Field Labels in UI Forms

\begin{center}
\textbf{Note}: Perform the procedure described in this section only if you are using Oracle Identity Manager release 11.1.2.x or later and you want to localize UI form field labels.
\end{center}

To localize field label that is added to the UI forms:

1. Log in to Oracle Enterprise Manager.
2. In the left pane, expand Application Deployments and then select oracle.iam.console.identity.sysadmin.ear.
3. In the right pane, from the Application Deployment list, select MDS Configuration.

4. On the MDS Configuration page, click Export and save the archive to the local computer.

5. Extract the contents of the archive, and open one of the following files in a text editor:
   - For Oracle Identity Manager 11g Release 2 PS2 (11.1.2.2.0):
     
     SAVED_LOCATION\xliffBundles\oracle\iam\ui\runtime\BizEditorBundle_en.xlf
   - For releases prior to Oracle Identity Manager 11g Release 2 PS2 (11.1.2.2.0):
     
     SAVED_LOCATION\xliffBundles\oracle\iam\ui\runtime\BizEditorBundle.xlf

6. Edit the BizEditorBundle.xlf file in the following manner:
   a. Search for the following text:
      
      <file source-language="en"
      original="/xliffBundles/oracle/iam/ui/runtime/BizEditorBundle.xlf"
      datatype="x-oracle-adf">
   
   b. Replace with the following text:
      
      <file source-language="en" target-language="LANG_CODE"
      original="/xliffBundles/oracle/iam/ui/runtime/BizEditorBundle.xlf"
      datatype="x-oracle-adf">

      In this text, replace LANG_CODE with the code of the language that you want to localize the form field labels. The following is a sample value for localizing the form field labels in Japanese:
      
      <file source-language="en" target-language="ja"
      original="/xliffBundles/oracle/iam/ui/runtime/BizEditorBundle.xlf"
      datatype="x-oracle-adf">
   
   c. Search for the application instance code. This procedure shows a sample edit for ACME Webservice application instance. The original code is:
      
      <trans-unit
      id="${adfBundle['oracle.adf.businesseditor.model.util.BaseRuntimeResourceBundle']['persdef.sessiondef.oracle.iam.ui.runtime.form.model.user.entity.useReO.UD_ACME_USR_COUNTRY__c_description']}">
      
      In this text, replace LANG_CODE with the code of the language that you want to localize the form field labels. The following is a sample value for localizing the form field labels in Japanese:
      
      <trans-unit
      id="sessiondef.oracle.iam.ui.runtime.form.model.ACMEWS.entity.ACMEWS0.UD_ACME_USR_COUNTRY__c_LABEL">
      
      In this text, replace LANG_CODE with the code of the language that you want to localize the form field labels. The following is a sample value for localizing the form field labels in Japanese:
      
      c. Search for the application instance code. This procedure shows a sample edit for ACME Webservice application instance. The original code is:
      
      <trans-unit
      id="sessiondef.oracle.iam.ui.runtime.form.model.ACMEWS.entity.ACMEWS0.UD_ACME_USR_COUNTRY__c_LABEL">
      
      In this text, replace LANG_CODE with the code of the language that you want to localize the form field labels. The following is a sample value for localizing the form field labels in Japanese:
      
      d. Open the resource file from the connector package, for example ACMEWS_ja.properties, and get the value of the attribute from the file, for example, global.udf.UD_ACME_USR_COUNTRY="\u56FD".

5. Replace the original code shown in Step 6.b with the following:
f. Repeat Steps 6.a through 6.d for all attributes of the process form.

g. Save the file as BizEditorBundle_<LANG_CODE>.xlf. In this file name, replace <LANG_CODE> with the code of the language to which you are localizing.

Sample file name: BizEditorBundle_ja.xlf.

7. Repackage the ZIP file and import it into MDS.

See Also: Deploying and Undeploying Customizations in Oracle Fusion Middleware Developing and Customizing Applications for Oracle Identity Manager, for more information about exporting and importing metadata files.

8. Log out of and log in to Oracle Identity Manager.
After you deploy the connector, you must configure it to meet your requirements. This chapter discusses the following connector configuration procedures:

- Section 4.1, "Configuring Reconciliation"
- Section 4.2, "Scheduled Tasks"
- Section 4.3, "Configuring Provisioning in Oracle Identity Manager Release 11.1.1"
- Section 4.4, "Configuring Provisioning in Oracle Identity Manager Release 11.1.2"
- Section 4.5, "Uninstalling the Connector"

### 4.1 Configuring Reconciliation

Reconciliation involves duplicating in Oracle Identity Manager the creation of and modifications to user accounts on the target system. While configuring the connector, the target system can be designated as a trusted source or target resource.

If you designate the target system as a **trusted source**, then during a reconciliation run:

- For each newly created user on the target system, an OIM User is created.
- Updates made to each user on the target system are propagated to the corresponding OIM User.

If you designate the target system as a **target resource**, then during a reconciliation run:

- For each account created on the target system, a resource is assigned to the corresponding OIM User.
- Updates made to each account on the target system are propagated to the corresponding resource.

This section discusses the following topics related to configuring reconciliation:

- Section 4.1.1, "Full Reconciliation"
- Section 4.1.2, "Limited Reconciliation"
- Section 4.1.3, "Batched Reconciliation"
- Section 4.1.4, "Configuring the Target System As a Trusted Source"
4.1.1 Full Reconciliation

Full reconciliation involves reconciling all existing user records from the target system into Oracle Identity Manager. After you deploy the connector, you must first perform full reconciliation.

To perform a full reconciliation run, remove (delete) any value currently assigned to the Filter attribute of the Target User Reconciliation scheduled task. See Section 4.2.2, "Scheduled Tasks for Reconciliation" for information about this scheduled task.

During a full reconciliation run, if you provide both batching parameters and filters, the connector processes the data in batches. Then, filters are applied to the processed data.

4.1.2 Limited Reconciliation

By default, all target system records that are added or modified after the last reconciliation run are reconciled during the current reconciliation run. You can customize this process by specifying the subset of added or modified target system records that must be reconciled.

You can perform limited reconciliation by creating filters for the reconciliation module. This connector provides a Filter attribute (a scheduled task attribute) that allows you to use Webservices resource attributes to filter the target system records.

For detailed information about ICF Filters, see ICF Filter Syntax in Oracle Fusion Middleware Developing and Customizing Applications for Oracle Identity Manager.

While deploying the connector, follow the instructions in Section 4.2.4, "Configuring Scheduled Jobs" to specify attribute values.

4.1.3 Batched Reconciliation

During a reconciliation run, all changes in the target system records are reconciled into Oracle Identity Manager. Depending on the number of records to be reconciled, this process may require a large amount of time. In addition, if the connection breaks during reconciliation, then the process would take longer to complete.

You can configure batched reconciliation to avoid these problems.

To configure batched reconciliation, you must specify values for the Batch Size scheduled task attribute. Use this attribute to specify the number of records that must be included in each batch.

By default, the value of Batch Size attribute is blank, indicating that all records will be included (no batched reconciliation). You specify a value for this attribute by following the instructions described in Section 4.2.4, "Configuring Scheduled Jobs."

The mechanism of returning multiple records during a search operation may vary with the target webservice. For batching, the connector includes two parameters, batch start index and batch end index. Some target webservices expect different parameters such as Start index and Page size. In such cases, the page size in the SOA composite should be defined in terms of connector parameters, for example, batchEnd - batchStart + 1 as shown below:

```
acmeows:getVariableData('SearchOp_InputVariable','parameters','/ns2:search/batchEnd') -
acmeows:getVariableData('SearchOp_InputVariable','parameters','/ns2:search/batchStart') + 1
```
4.1.4 Configuring the Target System As a Trusted Source

To configure trusted source reconciliation:

1. Log in to Oracle Identity Manager Administrative and User Console.
2. Update the Configuration Lookup parameter of the IT Resource to Lookup.ACME.Configuration.Trusted, where ACME indicates the target system.

You can change the entries in this configuration lookup if needed.

4.2 Scheduled Tasks

When you run the Connector Installer or import the connector XML file, the following reconciliation scheduled tasks are automatically created in Oracle Identity Manager:

This section discusses the following topics related to scheduled tasks:

- Section 4.2.1, "Scheduled Task for Lookup Field Synchronization"
- Section 4.2.2, "Scheduled Tasks for Reconciliation"
- Section 4.2.3, "Delete User Target Reconciliation"
- Section 4.2.4, "Configuring Scheduled Jobs"

4.2.1 Scheduled Task for Lookup Field Synchronization

The ACME Webservice Lookup Reconciliation scheduled task is used for lookup field synchronization. The ACME Webservice code indicates the target system.

You can specify values for the attributes of this scheduled job listed in the following table:

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Description</th>
</tr>
</thead>
</table>
| Code Key Attribute | Enter the name of the connector or target system attribute that is used to populate the Code Key column of the lookup definition (specified as the value of the Lookup Name attribute). Default value: __UID__  
  **Note:** Do not modify the value of this attribute. |
| Decode Attribute   | Enter the name of the connector or target system attribute that is used to populate the Decode column of the lookup definition (specified as the value of the Lookup Name attribute). Sample value: __NAME__  
  **Note:** Do not modify the value of this attribute. |
| IT Resource Name   | Enter the name of the IT resource for the target system installation from which you want to reconcile user records. Default value: ACME Webservice Server where ACME Webservice indicates the target system. |
ACME Webservice User Target Reconciliation scheduled job is used to reconcile user data in the target resource (account management) mode of the connector.

ACME Webservice User Trusted Reconciliation scheduled job is used to reconcile user data in the trusted source (identity management) mode of the connector.

The ACME Webservice code in the job names indicate the target system configured with the connector.

Table 4–2 describes the attributes of the scheduled tasks.

### Table 4–2 Attributes of the Scheduled Tasks for Reconciliation

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Lookup Name</td>
<td>This attribute holds the name of the lookup definition that maps each lookup definition with the data source from which values must be fetched. By default, this field is blank. For example, in the case of Roles, create a new lookup definition such as Lookup.ACME.Roles in the Design Console and provide the lookup name as the value of this attribute.</td>
</tr>
<tr>
<td>Object Type</td>
<td>Enter the type of object whose values must be synchronized. Sample value for Role lookup reconciliation: Roles</td>
</tr>
<tr>
<td>Resource Object Name</td>
<td>Enter the name of the resource object that is used for reconciliation. Default value: ACME Webservice User where ACME Webservice indicates the target system.</td>
</tr>
<tr>
<td>Batch Size</td>
<td>Specify the number of records that must be included in each batch By default, this field is blank. See Section 4.1.3, &quot;Batched Reconciliation&quot; for more information.</td>
</tr>
<tr>
<td>Filter</td>
<td>Expression for filtering records that must be reconciled by the scheduled task By default, the value of this attribute is empty. Sample value: equalTo('LastName','USER1') See Section 4.1.2, &quot;Limited Reconciliation&quot; for the syntax of this expression.</td>
</tr>
<tr>
<td>Incremental Recon Attribute</td>
<td>Name of the target system attribute that holds last update-related number, non-decreasing value. For example, numeric or strings. The value in this attribute is used during incremental reconciliation to determine the newest or youngest record reconciled from the target system. Sample value: timestamp Note: Ensure that the timestamp value is correctly mapped in the SOA composite in the search operation output transformation. Provide the timestamp value only if you want to run incremental reconciliation. Leave this field blank for full reconciliation.</td>
</tr>
<tr>
<td>IT Resource Name</td>
<td>Name of the IT resource for the target system installation from which you want to reconcile user records Default value: ACME Webservice Server where ACME Webservice indicates the target system.</td>
</tr>
</tbody>
</table>
4.2.3 Delete User Target Reconciliation

The ACME Webservice Delete User Target Reconciliation scheduled job is used to reconcile data about deleted users and user records. The ACME Webservice indicates the target system name. Table 4–3 lists the attributes of this scheduled job.

### Table 4–3 Attributes of the Delete User Target Reconciliation Scheduled Job

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>IT Resource Name</td>
<td>Name of the IT resource instance that the connector must use to reconcile data. Default value: ACME Webservice Server</td>
</tr>
<tr>
<td>Object Type</td>
<td>Type of object you want to reconcile. Default value: User</td>
</tr>
<tr>
<td>Resource Object Name</td>
<td>Name of the resource object against which reconciliation runs must be performed. Default value: ACME Webservice User</td>
</tr>
</tbody>
</table>

Note: For the resource object shipped with this connector, you must not change the value of this attribute. However, if you create a copy of the resource object, then you can enter the unique name for that resource object as the value of this attribute.

4.2.3.1 Adding defaultBatchSize as a Configuration Property

You can add defaultBatchSize as a configuration property to specify the default batch size for the reconciliation of records during delete reconciliation. The "defaultBatchSize" attribute can also be used for reconciliation in general.

To add defaultBatchSize as a configuration property, perform the following procedure:

1. Log in to the Design Console.
2. Expand Administration and double-click Lookup Definition.
3. Search for and open the existing Webservice Connector configuration lookup definition.

4. In the lookup configuration, provide a value for the defaultBatchSize parameter to make it as the default batch size.

5. Click Save.

4.2.4 Configuring Scheduled Jobs

To configure a scheduled task:

1. Depending on the Oracle Identity Manager release you are using, perform one of the following steps:
   - For Oracle Identity Manager release 11.1.1:
     a. Log in to the Administrative and User Console.
     b. On the Welcome to Oracle Identity Manager Self Service page, click Advanced in the upper-right corner of the page.
   - For Oracle Identity Manager release 11.1.2.x:
     a. Log in to Oracle Identity System Administration.
     b. In the left pane, under System Management, click Scheduler.

2. Search for and open the scheduled job as follows:
   - If you are using Oracle Identity Manager release 11.1.1, then on the Welcome to Oracle Identity Manager Advanced Administration page, in the System Management region, click Search Scheduled Jobs.
   - In the Search field, enter the name of the scheduled job as the search criterion. Alternatively, you can click Advanced Search and specify the search criterion.
   - In the search results table on the left pane, click the scheduled job in the Job Name column.

3. On the Job Details tab, you can modify the following parameters:
   - **Retries**: Enter an integer value in this field. This number represents the number of times the scheduler tries to start the job before assigning the Stopped status to the job.
Schedule Type: Depending on the frequency at which you want the job to run, select the appropriate schedule type.

**Note:** See Creating Jobs in Oracle Fusion Middleware Administering Oracle Identity Manager for detailed information about schedule types.

In addition to modifying the job details, you can enable or disable a job.

4. On the Job Details tab, in the Parameters region, specify values for the attributes of the scheduled task.

**Note:**
- Attribute values are predefined in the connector XML file that you import. Specify values only for those attributes that you want to change.
- Attributes of the scheduled jobs are described in the earlier sections in this chapter.

5. After specifying the attributes, click **Apply** to save the changes.

**Note:** The Stop Execution option is available in the Administrative and User Console. You can use the Scheduler Status page to either start, stop, or reinitialize the scheduler.

### 4.3 Configuring Provisioning in Oracle Identity Manager Release 11.1.1

Provisioning a resource for an OIM User involves using Oracle Identity Manager to create a target system account for the user.

If you have configured the connector for request-based provisioning, then the process form is suppressed and the object form is displayed. In other words, direct provisioning is disabled when you configure the connector for request-based provisioning. If you want to revert to direct provisioning, then perform the steps described in Section 4.3.3, "Switching Between Request-Based Provisioning and Direct Provisioning."

The following are types of provisioning operations:
- Direct provisioning
- Request-based provisioning
- Provisioning triggered by policy changes

**See Also:** Manually Completing a Task in Oracle Fusion Middleware Performing Self Service Tasks with Oracle Identity Manager for information about the types of provisioning

This section discusses the following topics:
- Section 4.3.1, "Configuring Direct Provisioning"
- Section 4.3.2, "Configuring Request-Based Provisioning"
4.3.1 Configuring Direct Provisioning

When you install the connector on Oracle Identity Manager, the direct provisioning feature is automatically enabled. This means that the process form is enabled when you install the connector.

In direct provisioning, the Oracle Identity Manager administrator uses the Administrative and User Console to create a target system account for a user.

To provision a resource by using the direct provisioning approach:

1. Log in to the Administrative and User Console.
2. On the Welcome to Identity Administration page, in the Users region, click Create User.
3. On the Create User page, enter values for the OIM User fields, and then click Save.
4. If you want to provision a target system account to an existing OIM User, then:
   - On the Welcome to Identity Administration page, search for the OIM User by selecting Users from the list on the left pane.
   - From the list of users displayed in the search results, select the OIM User. The user details page is displayed on the right pane.
5. On the user details page, click the Resources tab.
6. From the Action menu, select Add Resource. Alternatively, you can click the add resource icon with the plus (+) sign. The Provision Resource to User page is displayed in a new window.
7. On the Step 1: Select a Resource page, select ACME Webservice User from the list and then click Continue.
9. On the Step 5: Provide Process Data for User Details page, enter the details of the account that you want to create on the target system and then click Continue.
10. On the Step 6: Verify Process Data page, verify the data that you have provided and then click Continue.
11. Close the window displaying the “Provisioning has been initiated” message.
12. On the Resources tab, click Refresh to view the newly provisioned resource.

4.3.2 Configuring Request-Based Provisioning

In request-based provisioning, an end user creates a request for a resource by using the Administrative and User Console. Administrators or other users can also create requests for a particular user. Requests for a particular resource on the resource can be viewed and approved by approvers designated in Oracle Identity Manager.

The following are features of request-based provisioning:

- A user can be provisioned only one resource (account) on the target system.

---

**Note:** Direct provisioning allows the provisioning of multiple target system accounts on the target system.
Direct provisioning cannot be used if you enable request-based provisioning.

---

**Note:** The request dataset provided with the connector does not contain the User Login field, which is usually fed directly from Oracle Identity Manager user profile to the process form using a prepopulate adapter.

To include the User Login field in request dataset, perform the following procedure:

1. Export the current dataset using the MDS export utility.
2. Update the dataset to include the User Login field.
3. Import the updated dataset using the MDS import utility.
4. Purge the cache, as described in Section 3.2.5, "Clearing Content Related to Connector Resource Bundles from the Server Cache."

For information about exporting and importing request datasets, see http://docs.oracle.com/cd/E14571_01/doc.1111/e14309/utils.htm#BEIHDGCD.

For information about uploading request datasets into MDS, see http://docs.oracle.com/cd/E14571_01/doc.1111/e14309/request.htm#CIHIBFFA.

---

The following sections discuss the steps to be performed to enable request-based provisioning:

---

**Note:** The procedures described in these sections are built on an example in which the end user raises or creates a request for provisioning a target system account. This request is then approved by the approver.

---

- Section 4.3.2.1, "End User's Role in Request-Based Provisioning"
- Section 4.3.2.2, "Approver's Role in Request-Based Provisioning"
- Section 4.3.2.3, "Enabling the Auto Save Form Feature"
- Section 4.3.2.4, "Running the PurgeCache Utility"

### 4.3.2.1 End User's Role in Request-Based Provisioning

The following steps are performed by the end user in a request-based provisioning operation:

1. Log in to the Administrative and User Console.
2. On the Welcome page, click **Advanced** in the upper-right corner of the page.
3. On the Welcome to Identity Administration page, click the **Administration** tab, and then click the **Requests** tab.
4. From the Actions menu on the left pane, select **Create Request**.
   The Select Request Template page is displayed.
5. From the Request Template list, select **Provision Resource** and click **Next**.
6. On the Select Users page, specify a search criterion in the fields to search for the user that you want to provision the resource, and then click Search. A list of users that match the search criterion you specify is displayed in the Available Users list.

7. From the Available Users list, select the user to whom you want to provision the account.

   If you want to create a provisioning request for more than one user, then from the Available Users list, select users to whom you want to provision the account.

8. Click Move or Move All to include your selection in the Selected Users list, and then click Next.

9. On the Select Resources page, click the arrow button next to the Resource Name field to display the list of all available resources.

10. From the Available Resources list, select ACME Webservice User, move it to the Selected Resources list, and then click Next.

11. On the Resource Details page, enter details of the account that must be created on the target system, and then click Next.

12. On the Justification page, you can specify values for the following fields, and then click Finish.

   ■ Effective Date
   ■ Justification

   On the resulting page, a message confirming that your request has been sent successfully is displayed along with the Request ID.

13. If you click the request ID, then the Request Details page is displayed.

14. To view details of the approval, on the Request Details page, click the Request History tab.

#### 4.3.2.2 Approver’s Role in Request-Based Provisioning

The following are steps performed by the approver in a request-based provisioning operation:

The following are steps that the approver can perform:

1. Log in to the Administrative and User Console.

2. On the Welcome page, click Self-Service in the upper-right corner of the page.

3. On the Welcome to Identity Manager Self Service page, click the Tasks tab.

4. On the Approvals tab, in the first section, you can specify a search criterion for request task that is assigned to you.

5. From the search results table, select the row containing the request you want to approve, and then click Approve Task.

   A message confirming that the task was approved is displayed.

#### 4.3.2.3 Enabling the Auto Save Form Feature

To enable the Auto Save Form feature:

1. Log in to the Design Console.

2. Expand Process Management, and then double-click Process Definition.

3. Search for and open the ACME Webservice User process definition.
4. Select the **Auto Save Form** check box.

5. Click **Save**.

### 4.3.2.4 Running the PurgeCache Utility

Run the PurgeCache utility to clear content belonging to the Metadata category from the server cache. See Section 3.2.5, "Clearing Content Related to Connector Resource Bundles from the Server Cache" for instructions.

The procedure to configure request-based provisioning ends with this step.

### 4.3.3 Switching Between Request-Based Provisioning and Direct Provisioning

**Note:** It is assumed that you have performed the procedure described in Section 4.3.2, "Configuring Request-Based Provisioning."

1. Log in to the Design Console.
2. Disable the Auto Save Form feature as follows:
   a. Expand **Process Management**, and then double-click **Process Definition**.
   b. Search for and open the **ACME Webservice User** process definition.
   c. Deselect the **Auto Save Form** check box.
   d. Click **Save**.
3. If the Self Request Allowed feature is enabled, then:
   a. Expand **Resource Management**, and then double-click **Resource Objects**.
   b. Search for and open the **ACME Webservice User** resource object.
   c. Deselect the **Self Request Allowed** check box.
   d. Click **Save**.

To switch from direct provisioning back to request-based provisioning:

1. Log in to the Design Console.
2. Enable the Auto Save Form feature as follows:
   a. Expand **Process Management**, and then double-click **Process Definition**.
   b. Search for and open the **ACME Webservice User** process definition.
   c. Select the **Auto Save Form** check box.
   d. Click **Save**.
3. If you want to enable end users to raise requests for themselves, then:
   a. Expand **Resource Management**, and then double-click **Resource Objects**.
   b. Search for and open the **ACME Webservice User** resource object.
   c. Select the **Self Request Allowed** check box.
   d. Click **Save**.
4.4 Configuring Provisioning in Oracle Identity Manager Release 11.1.2

To configure provisioning operations in Oracle Identity Manager release 11.1.2.x:

---

**Note:** The time required to complete a provisioning operation that you perform the first time by using this connector takes longer than usual.

---

1. Log in to Oracle Identity System Administration.

2. Create and activate a sandbox. For detailed instructions on creating and activating a sandbox, see Managing Sandboxes in *Oracle Fusion Middleware Developing and Customizing Applications for Oracle Identity Manager*.

3. Create an application instance. To do so:
   a. In the left pane, under Configuration, click **Application Instances**. The Application Instances page is displayed.
   b. From the Actions menu, select **Create**. Alternatively, click **Create** on the toolbar. The Create Application Instance page is displayed.
   c. Specify values for the following fields:
      - **Name**: The name of the application instance.
      - **Display Name**: The display name of the application instance.
      - **Description**: A description of the application instance.
      - **Resource Object**: The resource object name. Click the search icon next to this field to search for and select **ACME Webservice**.
      - **IT Resource Instance**: The IT resource instance name. Click the search icon next to this field to search for and select **ACME Webservice Server**.
      - **Form**: Select the form name, for example, **ACME**. To do so, click **Create** against the Form list, specify the form name, and then create it. On the Create Application Instance page, click the Refresh icon next to the Form field. From this list, select the form name that you created.

4. Publish the sandbox.

5. Run lookup field synchronization. See Section 4.2.1, "Scheduled Task for Lookup Field Synchronization" for more information.

6. Search for and run the Entitlement List scheduled job to populate the ENT_LIST table. See Section 4.2.4, "Configuring Scheduled Jobs" for more information about configuring and running scheduled jobs.

7. Publish the application instance (created in Step 3) to an organization. To do so:
   a. On the Organizations tab of the Application Instance page, click **Assign**.
   b. In the Select Organizations dialog box, select the organization to which you want to publish the application instance.
   c. Select the **Apply to entitlements** checkbox.
   d. Click **OK**.

8. Search for and run the Catalog Synchronization Job scheduled job. See Section 4.2.4, "Configuring Scheduled Jobs" for more information about configuring and running scheduled jobs.

10. Create a user. See Creating a User in Oracle Fusion Middleware Performing Self Service Tasks with Oracle Identity Manager for more information about creating a user.

11. On the Account tab, click Request Accounts.

12. In the Catalog page, search for and add to cart the application instance created in Step 3, and then click Checkout.

13. Specify value for fields in the application form and then click Ready to Submit.

14. Click Submit.

15. If you want to provision entitlements, then:
   a. On the Entitlements tab, click Request Entitlements.
   b. In the Catalog page, search for and add to cart the entitlement, and then click Checkout.
   c. Click Submit.

4.5 Uninstalling the Connector

If you want to uninstall the connector for any reason, see Uninstalling Connectors in Oracle Fusion Middleware Administering Oracle Identity Manager.
This chapter discusses the following optional procedures:

- Section 5.1, "Securing the Connector"
- Section 5.2, "Adding Custom Attributes for Provisioning"
- Section 5.3, "Adding Custom Attributes for Reconciliation"
- Section 5.4, "Adding Custom Child Forms"
- Section 5.5, "Adding Child Form Data"
- Section 5.6, "Mapping Timestamp Attribute"
- Section 5.7, "Configuring the Connector for Multiple Instances and Multiple Versions of the Target System"
- Section 5.8, "Configuring Validation of Data During Reconciliation and Provisioning"
- Section 5.9, "Configuring Transformation of Data During User Reconciliation"
- Section 5.10, "Configuring Resource Exclusion Lists"
- Section 5.11, "Reconciliation of Complex Child Forms With Multiple Attributes"

**Note:** From Oracle Identity Manager Release 11.1.2 onward, lookup queries are not supported. See Managing Lookups in *Oracle Fusion Middleware Administering Oracle Identity Manager* guide for information about managing lookups by using the Form Designer in the Oracle Identity Manager System Administration console.

**Note:** In this guide, a target system that exposes webservice endpoint has been referred to as the **target system**. ACME Webservice is used as a sample target system to discuss the configurations and the connector objects.

### 5.1 Securing the Connector

This section describes the following procedures that enable you to secure the connector:

- Section 5.1.1, "Handling Passwords"
- Section 5.1.2, "Configuring Webservice Security Policy"
5.1.1 Handling Passwords

In this connector, the target webservice operations are invoked using SOAP messages, which by default are not encrypted and can be viewed by anyone from the Enterprise Manager or a testing utility. This poses a threat when you have to pass sensitive information like passwords. To secure sensitive information, the following custom webservice policy can be used:

- Sensitive fields are encrypted by Oracle Identity Manager and this encrypted value is sent to the SOA composite.
- The passcode attribute in the IT Resource of the connector is used as a key for encrypting the value.

Guidelines for Passcode

- Passcode should contain alphabets, numbers, and special characters.
- Passcode should contain both upper case and lower case characters.
- Passcode should be minimum 8 characters long.
- Passcode should be changed periodically.
- Passcode should not resemble any known and obvious keywords.
- Passcode provided in the SOA composite should match with the value of the passcode parameter in the connector IT Resource.

- In the SOA composite, the custom outbound policy (oimcp_WS_CONNECTOR_OUTBOUND) that handles password decryption is attached to the target webservice partner link.
- Passcode fields, password fields, and target namespaces are specified in the composite, which are used by the policy to decrypt the password fields.
- In the runtime, the policy decrypts a password field using the passcode and replaces it in the target SOAP payload before invoking the target webservice operation.
- Only the masked passwords are displayed in the Enterprise Manager and payloads.

To configure this custom webservice policy:

1. Adding the connector outbound policy to the Policy store. To do so:
   a. In the Enterprise Manager console, select the WebLogic domain in the left pane. In the main page, navigate to WebLogic Domain, Web Services, Policies.
b. In the Web Service Policies page, click **Import From File** and browse for the outbound policy. The oimcp_WS_CONNECTOR_OUTBOUND policy is available in the ConnectorDefaultDirectory/WebServices-11.1.5.0/soa/policy directory. Attach the sample outbound policy in Appendix B, "Sample Outbound Policy".

c. Click **OK** to import the policy.

You can verify if the policy is imported by visiting http://soaserverhost:soaport/wsm-pm/validator and confirm if the imported policy is listed on this page.
2. Deploy the custom policy JAR file, Webservices-oim-integration.jar. To do so:
   a. Stop the WebLogic (admin) server.
   b. Copy the ConnectorDefaultDirectory/Webservices-11.1.1.5.0/soa/policy/Webservices-oim-integration.jar file to the $DOMAIN_HOME/lib directory.
   c. Restart WebLogic server.
   d. Stop the SOA server.
   f. Set the ANT_HOME environment variable and run the ant command.
   g. Restart the SOA server.

3. Configure the SOA composite in the composite.xml file. To do so, add the following entries within the <binding.ws> tags of the webservice that requires password decryption.

```xml
<wsp:PolicyReference URI="oimcp/WS_CONNECTOR_OUTBOUND"
orawsp:category="security" orawsp:status="enabled"/>
<wsp:property name="passcode" type="xs:string">abcd1234</wsp:property>
<wsp:property name="target.payload.namespaces" type="xs:string">ns6=urn:/acme/xml/password</wsp:property>
```

In these entries:
- **passcode** is the passcode field of the connector IT Resource
  See "Guidelines for Passcode" listed earlier.
- **password.field.xpath.locations** is the comma separated list of XPath locations that contain the encrypted password fields
- **target.payload.namespaces** is the comma separated list of target namespaces corresponding to the values of password.field.xpath.locations
4. Deploy the SOA composite in JDeveloper and test the password reset operation from Oracle Identity Manager. See Section 2.5, "Deploying and Testing the Webservice SOA Composite" for more information.

5.1.2 Configuring Webservice Security Policy

You can configure webservice security policy in the SOA composite in JDeveloper. To do so:

1. Right-click the target webservice and click **Configure WS Policies**.
2. In the Configure SOA WS Policies window, click the plus sign next to the **Security** field.
3. In the Select Client Security Policies window, select the webservice security policy you want to add for authentication and click **OK**.
4. Click **OK**.
5. In the Component Palette, click the plus sign below the Binding Properties to specify the binding properties as required.

6. After the SOA composite is ready, you can build and deploy it in JDeveloper. See Section 2.5, "Deploying and Testing the Webservice SOA Composite" for more information.

5.1.3 Passing Credentials Using CSF

You can pass target system credentials from the SOA composite using the Credential Store Factory (CSF) mechanism. The procedure is as follows:
1. Create a key for the target system credentials in CSF. To do so:
   a. In the Enterprise Manager console, select the WebLogic domain in the left pane. In the main page, navigate to WebLogic Domain, Security, Credentials.
   
   ![Enterprise Manager console screenshot showing WebLogic Domain, Security, Credentials]

   b. Click Create Key and add the credential details.

   ![Create Key in Enterprise Manager console]

   c. Click OK and save the key.

2. Configure the SOA composite in the composite.xml file. To do so:
   a. Right-click the target webservice and click Configure WS Policies.
   
   ![Configure WS Policies in Enterprise Manager console]

   b. In the Configure SOA WS Policies window, select the security policy under Security and click the pencil sign to edit.
c. In the **Override Value** column, enter the name of the CSF key that was created in the previous step and click **OK**.

Alternatively, you can add the binding properties in the composite.xml file directly in text mode:

```xml
<binding.ws port="urn:acme/ws/user/#wsdl.endpoint(User/Default)"
location="userWrapper.wsdl" soapVersion="1.1">
  <wsp:PolicyReference URI="oracle/wss_username_token_client_policy"
orawsp:category="security" orawsp:status="enabled"/>
  <property name="csf-key" type="xs:string" many="false">acme-csf-key</property>
</binding.ws>
```

### 5.1.4 Passing Credentials Using Custom Headers

Apart from the webservice security policy authentication mechanisms, the webservices may be authenticated using custom SOAP headers. This mechanism
would be useful for the target webservices whose format do not comply with webservice security policy.

To pass the credentials using custom headers:

1. Define the schema of the elements used in the custom headers in an XSD file. Copy the file to the xsd folder in the SOA composite project in JDeveloper.

2. Create variables for the headers. In the BPEL process, click the variable icon (x) and add a variable.

3. Select the variable type from the custom header's XSD in the Type Chooser window.
The variables will be added to the variables list:

4. Assign values to the headers in the Assign activity.

5. Edit the Invoke operation.
6. Click the Header tab and select the variables that you created. This will add the selected variables in the SOAP header when the operation is invoked.

7. Click Apply and OK.
8. After the SOA composite is ready, you can build and deploy it. See Section 2.5, "Deploying and Testing the Webservice SOA Composite" for more information. The configured variables will be a part of the SOAP headers when the target webservice operation is invoked.

### 5.1.5 Importing SSL Certificate for HTTPS-based Target Webservice

You can perform this procedure only if the target system exposes the webservice over SSL. To import the SSL certificate:

1. Download the SSL certificate from the target system’s website that is exposing the webservice.
2. Log in to the Oracle WebLogic Server administration console.
3. Under Domain Structure, expand Environment (by clicking the + next to it). Then click Servers.
4. Click the SOA managed Server name. For example, soa_server1.
5. Switch to the Keystores tab.

   This page lists the various keystores that are associated with the server.

6. Import the SSL certificate into all the keystores that are listed on this page using keytool, the key and certificate management utility.

   For example, to import the certificate into the Demo Trust Keystore, run the following command from the computer hosting the server:
   ```
   keytool -keystore `<MW_HOME>/server/server/lib/DemoTrust.jks
   -storepass DemoTrustKeyStorePassPhrase -import -file acme-cert.cer
   ```

   Similarly, import the certificate to the other keystores that are listed.

7. Restart Oracle WebLogic Server and SOA server.
5.2 Adding Custom Attributes for Provisioning

Note: In this section, the term "attribute" refers to the identity data fields that store user data.

To add a custom attribute, you must ensure that the corresponding attribute exists on the target system. If it does not exist, then you must first add the custom attribute on the target system. Contact an administrator for information about adding a custom attribute on the target system.

You can add custom attributes for provisioning by configuring in Oracle Identity Manager and in the SOA Composite. These procedures are described in the following sections:

- Section 5.2.1, "Adding Custom Attributes for Provisioning in Oracle Identity Manager"
- Section 5.2.2, "Adding Custom Attributes for Provisioning in SOA Composite"
- Section 5.2.3, "Adding Custom Attribute for Update Operation"

5.2.1 Adding Custom Attributes for Provisioning in Oracle Identity Manager

By default, the attributes listed in Section 1.7, "Connector Objects Used During Provisioning" are mapped for provisioning between Oracle Identity Manager and the target system. If required, you can also configure the connector for provisioning after adding custom attributes or other user attributes that are not available out of the box (OOTB) with the connector.

For example, if CountryName is a custom attribute added to the user profile on the target system, then you can configure the connector to provision this attribute by performing the following steps:

1. For the custom attribute, CountryName, determine the corresponding attribute name in ACME WSDL.
2. Log in to the Oracle Identity Manager Design Console.
   
   If you are using Oracle Identity Manager release 11.1.2.x, then log in to Oracle Identity System Administration and perform the steps described in http://docs.oracle.com/cd/E27559_01/admin.1112/e27149/form.htm#CACGHJIF.

3. Create a new version of the process form as follows:
   
   a. Expand Development Tools.
   b. Double-click Form Designer.
   c. Search for and open the UD_ACME_USR process form.
   d. Click Create New Version.
      
      On the Create a new version dialog box, enter a new version in the Label field, and then click Save.

4. Add the new field on the process form as follows:
   
   a. Click Add.
      
      A field is added to the list. Enter the details of the field.
For example, if you are adding the CountryName field, enter `UD_ACME_USR_COUNTRYNAME` in the Name field, `CountryName` in the Label Name field, and the remaining details of this field.

If the field is a LookupField type, create a new lookup definition, for example, `Lookup.ACME.CountryName`. Then, add appropriate entries to the lookup definition.

Open the `UD_ACME_USR` process form and click Properties. Select the newly added property and click Add Property. Select Property Name as Lookup Code, and then enter the newly created lookup, `Lookup.ACME.CountryName` as the property value.

b. Click Save.

c. To activate the newly created form, click Make Version Active.

5. Create an entry for the field in the lookup definition for provisioning as follows:

   a. Expand Administration.

   b. Double-click Lookup Definition.

   c. Search for and open the `Lookup.ACME.UM.ProvAttrMap` lookup definition.

   d. Click Add and enter the Code Key and Decode values for the field.

      The Code Key value must be the form field label name. The Decode value can be the same as the Code Key value, as the mapping is done in the SOA composite.

      For example, enter `CountryName` in the Code Key and the Decode fields. After this attribute is added in the Provisioning Attribute Map and in the process form, the attribute will appear in the Oracle Identity Manager App Instance form and the input values can be provided from Oracle Identity Manager.

   e. Click Save.

6. If you are using Oracle Identity Manager release 11.1.2.x or later, create a new UI form and attach it to the application instance to make this new attribute visible. See Section 3.2.1.2, "Creating a New UI Form" and Section 3.2.1.6, "Updating an Existing Application Instance with a New Form" for the procedures.

### 5.2.2 Adding Custom Attributes for Provisioning in SOA Composite

You can add custom attributes in the SOA composite for an operation such as Create. The custom attribute will be passed in the `<otherAttributes>` tag in the payload. The custom attribute can be the Decode value `CountryName` from the `Lookup.ACME.UM.ProvAttrMap` lookup definition. To do so:

1. In the Assign activity, expand `otherAttributes` under `userAccount` and map the value to the target attribute.
2. Edit the **From XPath** field and add \[name = 'CountryName'\] before /value.

3. Save the assignment.
   The blue assignment line shifts to userAccount in the From region.

4. Save the project.
   You can compile and deploy the project. Test the operation from the Enterprise Manager. See Section 2.5, "Deploying and Testing the Webservice SOA Composite" for more information.

### 5.2.3 Adding Custom Attribute for Update Operation

To add a custom attribute for update operation. To do so:

**Note:** Ensure that the field has been added to the Oracle Identity Manager process form and the entry has been added in Lookup.ACME.UM.ProvAttrMap so that create operation is working.

1. In the Design Console, add the **FIELD_NAME Updated** process task.
   For example, if the custom field is Location, add the Location Updated task.
2. Add the adpACMEUPDATEATTRIBUTEVALUE adapter and complete the integration similar to an existing task such as FirstName Updated.
3. In the SOA composite, map the name under udpatedAttribute to the if construct and the value to the target variable.
4. Switch to source. The following is a sample source:

```xml
<xsl:if test="/types:update/updatedAttribute/name">
  <firstName>
    <xsl:value-of select="/types:update/updatedAttribute/value"/>
  </firstName>
</xsl:if>
```

5. Verify the Decode value for the connector field in the Lookup.ACME.UM.ProvAttrMap lookup definition.

For the example attribute name first name, the Decode value is FirstName.

6. Modify the source as follows:

```xml
<xsl:if test='/types:update/updatedAttribute/name = "FirstName"'>
  <firstName>
    <xsl:value-of select='/types:update/updatedAttribute/updatedAttribute[name = 'FirstName']/value'/>
  </firstName>
</xsl:if>
```

7. Save the project.

You can compile and deploy the project. Test the operation from the Enterprise Manager. See Section 2.5, "Deploying and Testing the Webservice SOA Composite" for more information.

### 5.3 Adding Custom Attributes for Reconciliation

You can add custom attributes for reconciliation by configuring in Oracle Identity Manager and in the SOA Composite. These procedures are described in the following sections:

- [Section 5.3.1, "Adding Custom Attributes for Reconciliation in Oracle Identity Manager"](#)
- [Section 5.3.2, "Adding Custom Attributes for Reconciliation in SOA Composite"](#)
- [Section 5.3.3, "Adding Custom Attributes for Reconciling _UID_ Field"](#)
5.3.1 Adding Custom Attributes for Reconciliation in Oracle Identity Manager

To add a custom attribute such as Country Name in Oracle Identity Manager:

Note: If you have already added an attribute for provisioning, then you need not repeat steps performed as part of that procedure.

1. In the Design Console, search for and open the Lookup.ACME.UM.ReconAttrMap lookup definition.

2. Add a new entry for the custom attribute.

3. Add the custom field to the list of reconciliation fields in the resource object. Then, click Create Reconciliation Profile.

4. Add the custom field on the process form.
5. Create a reconciliation field mapping for the custom field in the provisioning process.

5.3.2 Adding Custom Attributes for Reconciliation in SOA Composite

You can add custom attributes in the SOA composite for an operation such as Search. The custom attribute will be passed in the `<otherAttributes>` tag in the payload, as shown in SearchOutputTransform.xsl.
To perform this procedure:

1. Map the custom attribute to the value field under `otherAttributes`.

2. The `name` field under `otherAttributes` has to be set with the Decode value of the attribute in `Lookup.ACME.UM.ReconAttrMap` lookup.

   Right-click `name` and click **Set Text**, then **Enter Text**.
3. Enter the decode value of the custom attribute as defined in the lookup definition.

4. Switch to the Source tab to view the XSL transform code for the `otherAttributes`.

   You can add as many `otherAttribute` tags as needed. By adding more tags, the Design view may throw errors that can be ignored. As long as the XSLT syntax is correct, the project can be deployed and tested.
5.3.3 Adding Custom Attributes for Reconciling _UID_ Field

**Note:** Perform the procedure described in this section only when the name and Uid fields in your target system do not store the same values.

Customizing the Uid field includes configuring the otherAttributes attribute in the composite and then assigning it to the Unique Id field in the Lookup.ACME.UM.ReconAttrMap lookup definition.

To do so, perform the following steps in the Transform after UserInvokeSearch:

1. Open the search branch transform and map the Uid field to the otherAttributes attribute as follows:
   a. Search for and expand the otherAttributes attribute and right-click the Name attribute.
   b. In the context menu that is displayed, select Set Text, and then select Enter Text.
c. In the Set Text dialog box, select **Text**, enter `uid` in the text field, and click **OK**.

d. In the UserSearchTransform tab, map the target `Uid` field to the attribute value in `otherAttributes`.
2. Save, compile, and deploy the composite to the SOA server.

3. Test the Search operation from Enterprise Manager and observe the response payload.
   Login is populated in the login field and Uid is set as uid in the otherAttributes attribute.

4. Update the Lookup.ACME.UM.ReconAttrMap lookup definition as follows:
   a. Log in to the Design Console.
   b. Expand Administration and then double-click Lookup Definition.
   c. Search for and open the Lookup.ACME.UM.ReconAttrMap lookup definition.
   d. Search for and update the decode value of Unique Id code key to uid.
Click Save.

5. Run the Target User Reconciliation scheduled job. See Section 4.2.2, "Scheduled Tasks for Reconciliation" for more information about this scheduled job.

After the reconciliation run is successful, the unique ID value is mapped to the custom attribute. You can view this mapping in the Event Management region by opening the latest event ID.
5.4 Adding Custom Child Forms

You can add custom child forms by configuring in Oracle Identity Manager and in the SOA Composite. These procedures are described in the following sections:

- Section 5.4.1, “Adding Custom Child Forms in Oracle Identity Manager”
- Section 5.4.2, “Adding Custom Child Forms in SOA Composite”

5.4.1 Adding Custom Child Forms in Oracle Identity Manager

To add a custom child form for a field such as Mailing List in Oracle Identity Manager:

1. In the Form Designer, create the child form for the Mailing List field.

2. Create a new version of the parent form and add the new child form to it. Then, make the new version of the parent form active.

3. Add process tasks for Mailing List Insert, Mailing List Update, and Mailing List Delete. These process tasks will be similar to the child form Insert, Update, and Delete tasks.
4. Add an entry in the Lookup.ACME.UM.ProvAttrMap lookup definition.

**Code Key:** CHILD_TABLE~FIELD_LABEL

**Decode:** RELEVANT_STRING_VALUE

For example, Code Key value is UD_ACME.CH2~Mailing List and Decode value is MailingList.

**Note:** For complex child tables, the decode value is AttributeName~ObjectClass~TargetFieldName

You must provide values for the AttributeName and ObjectClass attributes as mentioned under ComplexMultiAttributes in the SOA composite.
5.4.2 Adding Custom Child Forms in SOA Composite

In the ACME webservice, there are different operations for different multivalued attributes (child form attributes), such as AddRole and AddMailingList operations.

To add a custom child form for a field such as Mailing List in the SOA composite:

1. Drop a Switch activity to UpdateAddAttributeValues OnMessage. Uncomment it in the source view if it was previously commented.

2. Double-click the condition box to open the Edit Switch Case window. Provide the name as Role and click the XPath Expression Builder icon above the Expression region.

3. Select contains from String functions and click Insert Into Expression.

4. Set the first argument as attributeName under UpdateAddAttributeValuesOp_InputVariable.
5. Set second argument as Role and set the label as Role.

The second argument value is the same as the Decode value of the first child table entry in the Lookup.ACME.UM.ProvAttrMap lookup.

6. Drop an Invoke activity for calling the AddRole operation in the Role branch. Then, drop another Invoke activity for calling the AddMailingList operation in the otherwise branch.

Specify the input and output variables.
7. Add Assignment activities before the Invoke activities and map the attributes to the target operation attributes. Then, map the UID to the output variable of the Reply activity.

The following is a sample screenshot of the configured SOA composite:
5.5 Adding Child Form Data

The `UpdateAddAttributeValues` and `UpdateRemoveAttributeValues` operations are used for adding and removing child form data such as Roles (multivalued data or entitlements) respectively.

The mappings for `UpdateAddAttributeValues` are as follows. The mappings for `UpdateRemoveAttributeValues` are similar.

1. Depending on the target webservice, you may have to create a new partner link if the target exposes Role operations in a separate webservice.

2. Drop an Invoke activity to the target operation such as AddRole. Then, specify the input and output variables.

3. Drop an Assign activity before the InvokeRoleAd activity.

4. In the Assign activity, map the variables in the `UpdateAddAttributeValueOp_InputVariable` to the input variable of the Invoke activity for Role.
In the case of ACME web service, the first argument is the UID and the second argument is the attribute name, Role (for multiple child tables, this value would change). The third argument is the attribute value, which is the actual role name such as Administrator.

5. Drop an Assign activity between the Invoke and the AddAttrValReply activities. Then, map the output of the Invoke activity (which is the UID) to the Response variable.

If the target web service does not return the UID on role addition, then this assignment can be done between the UID passed in the input call to the UpdateAddAttributeValues operation and the response variable.

You can follow similar steps to configure the RemoveRole operation using the UpdateRemoveAttributeValues operation.

**5.6 Mapping Timestamp Attribute**

The timestamp attribute in the connector is of long type. The attribute on some target webservices may be of a different type or format. For example, in Oracle CRM On Demand, if you want to use the modified date as the incremental reconciliation attribute, then you need to convert the attribute type to long.

**See Also:** Table 4–2 for the usage of the timestamps in the scheduled task attributes
Initially, timestamp is a string in MM/dd/yyyy format. You can write a Java code to convert the string into java.sql.Timestamp format and then, use the getTime() method to derive the value of long type.

This Java code can be used to define a custom XPath function that can be used directly in the SOA composite, which takes the modified date as input and produces the long value as output. Then, this output can be mapped to the timestamp attribute.

Perform the procedure described in Creating User-Defined XPath Extension Functions in Oracle Fusion Middleware Developer’s Guide for Oracle SOA Suite to create a custom (user-defined) XPath extension function.

The following is a sample Java code to define a custom XPath function to convert DateString to Long type:

```java
package org.webservices.conversion;

import java.text.SimpleDateFormat;
import java.util.Date;
import java.util.List;
import oracle.fabric.common.xml.xpath.IXPathContext;
import oracle.fabric.common.xml.xpath.IXPathFunction;

public class ConvertDateStringToLong implements IXPathFunction {
    public ConvertDateStringToLong() {
        super();
    }

    public static Long convertDateStringToLong(String dateString) {
        Date date = null;
        SimpleDateFormat dateFormat = new SimpleDateFormat("MM/dd/yyyy HH:mm:ss");
        try {
            date = dateFormat.parse(dateString);
        } catch (Exception e) {
            e.printStackTrace();
        }
        long timeInLong = date.getTime();
        return timeInLong;
    }

    public Object call(IXPathContext ixPathContext, List<?> list) {
        return convertDateStringToLong((String)list.get(0));
    }
}
```
5.7 Configuring the Connector for Multiple Instances and Multiple Versions of the Target System

Note: Perform this procedure only if you want to configure the connector for multiple installations of the target system.

You may want to configure the connector for multiple installations of the target system. The following example illustrates this requirement:

The Tokyo, London, and New York offices of Example Multinational Inc. have their own installations of the target system. The company has recently installed Oracle Identity Manager, and they want to configure Oracle Identity Manager to link all the installations of the target system.

To meet the requirement posed by such a scenario, you must configure the connector for multiple installations of the target system.

To configure the connector for multiple installations of the target system:

See Also: Cloning Connectors in Oracle Fusion Middleware Administering Oracle Identity Manager for Oracle Identity Manager for detailed instructions on performing each step of this procedure

1. Create and configure one IT resource for each target system installation.

   The IT Resources form is in the Resource Management folder. An IT resource is created when you import the connector XML file. You can use this IT resource as the template for creating the remaining IT resources, of the same resource type.

2. Configure reconciliation for each target system installation. See Section 4.2, “Scheduled Tasks” for instructions. Note that you only need to modify the attributes that are used to specify the IT resource and to specify whether or not the target system installation is to be set up as a trusted source.

3. If required, modify the fields to be reconciled for the ACME Webservice User resource object.

   When you use the Administrative and User Console to perform provisioning, you can specify the IT resource corresponding to the target system installation to which you want to provision the user.

5.8 Configuring Validation of Data During Reconciliation and Provisioning

The Lookup.ACME.UM.ProvValidations and Lookup.ACME.UM.ReconValidations lookup definitions hold single-valued data to be validated during provisioning and reconciliation operations, respectively.

For example, you can validate data fetched from the First Name attribute to ensure that it does not contain the number sign (#). In addition, you can validate data entered in the First Name field on the process form so that the number sign (#) is not sent to the target system during provisioning operations.
To configure validation of data:

1. Write code that implements the required validation logic in a Java class with a fully qualified domain name (FQDN), such as `org.identityconnectors.acme.extension.ACMEValidator`. This validation class must implement the validate method. The following sample validation class checks if the value in the First Name attribute contains the number sign (#):

```java
package com.validationexample;
import java.util.HashMap;
public class MyValidator {
    public boolean validate(HashMap hmUserDetails, HashMap hmEntitlementDetails, String sField) throws ConnectorException {
        boolean valid = true;
        String sFirstName = (String) hmUserDetails.get(sField);
        for (int i = 0; i < sFirstName.length(); i++) {
            if (sFirstName.charAt(i) == '#') {
                valid = false;
                break;
            }
        }
        return valid;
    }
}
```

2. Log in to the Design Console.

3. Create one of the following new lookup definitions:

   - To configure validation of data for reconciliation:
     `Lookup.ACME.UM.ReconValidations`

Note: The Lookup.ACME.UM.ProvValidations and Lookup.ACME.UM.ReconValidations lookup definitions are optional and do not exist by default.

You must add these lookups as decode values to the Lookup.ACME.UM.Configuration lookup definition to enable exclusions during provisioning and reconciliation operations.
To configure validation of data for provisioning:
Lookup.ACME.UM.ProvValidations

4. In the **Code Key** column, enter the resource object field name that you want to validate. For example, Alias.

5. In the **Decode** column, enter the class name. For example, org.identityconnectors.acme.extension.ACMEValidator.

6. Save the changes to the lookup definition.

7. Search for and open the **Lookup.ACME.UM.Configuration** lookup definition.

8. In the **Code Key** column, enter one of the following entries:
   - To configure validation of data for reconciliation:
     Recon Validation Lookup
   - To configure validation of data for provisioning:
     Provisioning Validation Lookup

9. In the **Decode** column, enter one of the following entries:
   - To configure validation of data for reconciliation:
     Lookup.ACME.UM.ReconValidations
   - To configure validation of data for provisioning:
     Lookup.ACME.UM.ProvValidations

10. Save the changes to the lookup definition.

11. Create a JAR with the class and upload it to the Oracle Identity Manager database as follows:

Run the Oracle Identity Manager Upload JARs utility to post the JAR file to the Oracle Identity Manager database. This utility is copied into the following location when you install Oracle Identity Manager:

For Microsoft Windows:

```
OIM_HOME/server/bin/UploadJars.bat
```

For UNIX:

```
OIM_HOME/server/bin/UploadJars.sh
```

When you run the utility, you are prompted to enter the login credentials of the Oracle Identity Manager administrator, URL of the Oracle Identity Manager host computer, context factory value, type of JAR file being uploaded, and the location from which the JAR file is to be uploaded. Select 1 as the value of the JAR type.

12. Run the PurgeCache utility to clear content related to request datasets from the server cache.

13. Perform reconciliation or provisioning to verify validation for the field, for example, Alias.

---

**Note:** Before you use this utility, verify that the `WL_HOME` environment variable is set to the directory in which Oracle WebLogic Server is installed.
5.9 Configuring Transformation of Data During User Reconciliation

The Lookup.ACME.UM.ReconTransformations lookup definition holds single-valued user data to be transformed during reconciliation operations. For example, you can use First Name and Last Name values to create a value for the Full Name field in Oracle Identity Manager.

**Note:** The Lookup.ACME.UM.ReconTransformations lookup definition is optional and does not exist by default.
You must add this lookup as decode value to the Lookup.ACME.UM.Configuration lookup definition to enable exclusions during provisioning and reconciliation operations.

To configure transformation of single-valued user data fetched during reconciliation:

1. Write code that implements the required transformation logic in a Java class with a fully qualified domain name (FQDN), such as org.identityconnectors.acme.extension.ACMETransformation.
   
   This transformation class must implement the transform method. The following sample transformation class creates a value for the Full Name attribute by using values fetched from the First Name and Last Name attributes of the target system:
   
   ```java
   package com.transformationexample;

   import java.util.HashMap;

   public class MyTransformer {
       public Object transform(HashMap hmUserDetails, HashMap hmEntitlementDetails, String sField) throws ConnectorException {
           /*
           * You must write code to transform the attributes.
           * Parent data attribute values can be fetched by
           * using hmUserDetails.get("Field Name").
           * To fetch child data values, loop through the
           * ArrayList/Vector fetched by hmEntitlementDetails.get("Child Table")
           *
           * Return the transformed attribute.
           */
           String sFirstName = (String) hmUserDetails.get("First Name");
           String sLastName = (String) hmUserDetails.get("Last Name");
           return sFirstName + "." + sLastName;
       }
   }
   ```

2. Log in to the Design Console.

3. Create a new lookup definition, **Lookup.ACME.UM.ReconTransformations**.

4. In the **Code Key** column, enter the resource object field name you want to transform. For example, **Alias**.

5. In the **Decode** column, enter the class name. For example, org.identityconnectors.acme.extension.ACMETransformation.

6. Save the changes to the lookup definition.

7. Search for and open the **Lookup.ACME.UM.Configuration** lookup definition.
8. In the **Code Key** column, enter `Recon Transformation Lookup`.

9. In the **Decode** column, enter `Lookup.ACME.UM.ReconTransformations`.

10. Save the changes to the lookup definition.

11. Create a JAR with the class and upload it to the Oracle Identity Manager database as follows:

    Run the Oracle Identity Manager Upload JARs utility to post the JAR file to the Oracle Identity Manager database. This utility is copied into the following location when you install Oracle Identity Manager:

    ```
    Note: Before you use this utility, verify that the `WIL_HOME` environment variable is set to the directory in which Oracle WebLogic Server is installed.
    ```

    For Microsoft Windows:

    ```
    OIM_HOME/server/bin/UploadJars.bat
    ```

    For UNIX:

    ```
    OIM_HOME/server/bin/UploadJars.sh
    ```

    When you run the utility, you are prompted to enter the login credentials of the Oracle Identity Manager administrator, URL of the Oracle Identity Manager host computer, context factory value, type of JAR file being uploaded, and the location from which the JAR file is to be uploaded. Select 1 as the value of the JAR type.

12. Run the PurgeCache utility to clear content related to request datasets from the server cache.

13. Perform reconciliation to verify transformation of the field, for example, Alias.

### 5.10 Configuring Resource Exclusion Lists

The Lookup.ACME.UM.ProvExclusionList and Lookup.ACME.UM.ReconExclusionList lookup definitions hold user IDs of target system accounts for which you do not want to perform provisioning and reconciliation operations, respectively.

```
Note: The Lookup.ACME.UM.ProvExclusionList and Lookup.ACME.UM.ReconExclusionList lookup definitions are optional and do not exist by default.

You must add these lookups as decode values to the Lookup.ACME.UM.Configuration lookup definition to enable exclusions during provisioning and reconciliation operations.
```
To add entries in the lookup for exclusions during provisioning operations:

1. On the Design Console, expand Administration and then double-click Lookup Definition.

2. Create a new lookup definition, Lookup.ACME.UM.ProvExclusionList.

   - Click Add.
   - In the Code Key and Decode columns, enter the first user ID to exclude.

   **Note:** To specify user IDs to be excluded during reconciliation operations, create a new lookup definition called Lookup.ACME.UM.ReconExclusionList and add entries to that lookup.

3. Click Add.

4. In the Code Key and Decode columns, enter the first user ID to exclude.

   **Note:** The Code Key represents the resource object field name on which the exclusion list is applied during provisioning operations.

5. Repeat Steps 3 and 4 for the remaining user IDs to exclude.

   For example, if you do not want to provision users with user IDs User001, User002, and User088 then you must populate the lookup definition with the following values:

<table>
<thead>
<tr>
<th>Code Key</th>
<th>Decode</th>
</tr>
</thead>
<tbody>
<tr>
<td>User Login Id</td>
<td>User001</td>
</tr>
<tr>
<td>User Login Id</td>
<td>User002</td>
</tr>
<tr>
<td>User Login Id</td>
<td>User088</td>
</tr>
</tbody>
</table>

   You can also perform pattern matching to exclude user accounts. You can specify regular expressions supported by the representation in the java.util.regex.Pattern class.

   **See Also:** For information about the supported patterns, visit http://download.oracle.com/javase/6/docs/api/java/util/regex/PATTERN.html

To exclude users matching any of the user ID’s User001, User002, User088, then:

- Decode: User001 | User002 | User088

To exclude users whose user ID’s start with 00012, then:

- Decode: 00012*
For example, if you do not want to provision users matching any of the user IDs User001, User002, and User088, then you must populate the lookup definition with the following values:

<table>
<thead>
<tr>
<th>Code Key</th>
<th>Decode</th>
</tr>
</thead>
<tbody>
<tr>
<td>User Login Id[PATTERN]</td>
<td>User001</td>
</tr>
</tbody>
</table>

If you do not want to provision users whose user IDs start with 00012, then you must populate the lookup definition with the following values:

<table>
<thead>
<tr>
<th>Code Key</th>
<th>Decode</th>
</tr>
</thead>
<tbody>
<tr>
<td>User Login Id[PATTERN]</td>
<td>00012*</td>
</tr>
</tbody>
</table>

6. Click Save.

5.11 Reconciliation of Complex Child Forms With Multiple Attributes

After performing the configuration procedure described in Section 2.3.7, "Configuring the Search Operation," you can map a child table with more than one attribute as follows:

1. Open and expand the transformation mappings created for the Search operation.

2. Create required mappings for the User fields, after which you must map the complex child attribute. To do so, perform the following procedure:
   
a. Add a "for each" loop to repeat the procedure through the child table values.

   For example, in this case, you have to repeat the procedure continuously for each "Role" attribute of the User. Each attributeValue will represent a Role and
the procedure is repeated continuously through each "Role" using the "for each" loop.

b. Add a "for each" loop to repeat the procedure through the values of each child table entry, and map it to the name and values in the complexAttributeValues.

For example, in this case, you have to repeat the procedure continuously through roleName, roleStartDate, roleEndDate, and primary for each "Role" attribute. Each of the attributes of Role will be represented as a name-value pair.

As an example, the name variable will have "RoleName" and the value variable will have "Developer". Similarly all the other attributes of Role such as RoleStartDate, RoleEndDate, and primary will also be represented by name-value pairs.

The repetition of each attribute in the child table is done using the following code, which is present inside the "for-loop" to repeat the procedure through each "Role" attribute:

```xml
<xsl:for-each select="node()">
  <xsl:if test='(name() = "roleName") or ((name() = "roleStartDate") or ((name() = "roleEndDate") or (name() = "primary")))'>
    <complexAttributeValues>
      <name>
        <xsl:value-of select="name()"/>
      </name>
      <value>
        <xsl:value-of select="node()"/>
      </value>
    </complexAttributeValues>
  </xsl:if>
</xsl:for-each>
```
After completing the above procedure, you are required to perform the following steps in Oracle Identity Manager:

1. Log in to the Design Console.

2. Create a new child table with more than one attribute as follows:
   a. Create a new child table by following the procedure specified in Section 5.4.1, "Adding Custom Child Forms in Oracle Identity Manager."
   b. Add all the attributes of the child table to the additional columns.
      For example, if you have created a child table for Role, you have to add the following attributes to the newly created child table:
      - RoleName
      - RoleStartDate
      - RoleEndDate
      - Primary
   c. In the Properties tab, click Add Property and add the required properties of the attributes as shown below:
d. Click **Save**, and then click **Make Version Active**.

e. Create a new version of the parent form and add the new child form to it. Once this is complete, make the new version of the parent form active by clicking **Make Version Active**.

3. Add the new attribute to the list of reconciliation fields in the resource object as follows:


   b. Search for and open the **SampleWS User** resource object as per the example used in this procedure.

   c. On the **Object Reconciliation** tab, click **Add Field**.

   d. Enter the details of the field.

      For example, enter **Role** in the **Field Name** field and select **Multi-Valued Attribute** from the Field Type list.
e. Click **Save** and close the dialog box.

f. Right-click the newly created field and select **Define Property Fields**.

g. In the Add Reconciliation Fields dialog box, enter the details of the newly created field.

   For example, enter **RoleName** in the **Field Name** field and select **String** from the Field Type list.

h. Repeat step g for all the attributes of the child table with the following difference:

   - Enter **RoleStartDate** in the **Field Name** field and select **String** from the Field Type list.
   - Enter **RoleEndDate** in the **Field Name** field and select **String** from the Field Type list.
   - Enter **Primary** in the **Field Name** field and select **String** from the Field Type list.

   Click **Save** and close the dialog box repeatedly after you enter the details for each new attribute.
4. Create a reconciliation field mapping for the new attribute in the process definition as follows:
   a. Expand **Process Management**, and double-click **Process Definition**.
   b. Search for and open SampleWS User process definition as per the example used in this procedure.
   c. On the Reconciliation Field Mappings tab of the process definition form, click **Add Field Map**.
   d. In the Add Reconciliation Table Mapping dialog box, select the field name and table name from the list displayed.
   e. Click **Save** and close the dialog box.

The following screenshot displays the addition of field name Role to SampleWS User reconciliation mappings:
Reconciliation of Complex Child Forms With Multiple Attributes

5-44 Oracle Identity Manager Connector Guide for Webservices

f. Right-click the newly created field, and select Define Property Field Map.

g. In the Field Name field, select the value for the field that you want to add.

The following screenshot displays the addition of RoleName field of the child table Role:

h. Repeat steps f and g for fields RoleStartDate, RoleEndDate, and Primary.

5. Create an entry for the field in the lookup definition for reconciliation as follows:

a. Expand Administration.

b. Double-click Lookup Definition.

c. Click Add and enter the Code Key and Decode values for the field. The Code Key and Decode values must be in the following format:
Code Key:  
MULTIVALUED_FIELD_NAME~CHILD_RESOURCE_OBJECT_FIELD_NAME

Decode: AttributeName~ObjectClass~TargetFieldName

**Note:** Provide the values for AttributeName and ObjectClass as specified in AttributeName and ObjectClass under ComplexMultiAttributes in the SOA composite.

d. Click **Save**.

e. Repeat steps c and d for all the attributes of the child table.

The below screenshot displays the Reconciliation Attribute Map of SampleWS User on completion of all the mappings:
After you deploy the connector, you must test it to ensure that it functions as expected. This chapter discusses the topics related to connector testing.

See Also: Section 2.5, "Deploying and Testing the Webservice SOA Composite" and Section 2.4, "Handling Faults"

6.1 Troubleshooting

The following table lists solutions to some commonly encountered issues associated with this connector:

<table>
<thead>
<tr>
<th>Problem Description</th>
<th>Solution</th>
</tr>
</thead>
<tbody>
<tr>
<td>When you try to run an operation, you get an error similar to the following on the SOA server:</td>
<td>Ensure that the mappings for the operation specified in the error are complete.</td>
</tr>
<tr>
<td>Correlation definition not registered</td>
<td>If the mappings are complete, then undeploy and redeploy the SOA composite on the SOA server.</td>
</tr>
<tr>
<td></td>
<td>If the issue persists, you can try restarting the SOA server after undeploying.</td>
</tr>
</tbody>
</table>
## Problem Description

When you try to run an operation, you get a RemoteFault similar to the following:

```
oracle.fabric.common.FabricInvocationException: Unable to invoke endpoint URI successfully due to:
oracle.fabric.common.PolicyEnforcementException: FailedCheck :failure in security check
```

You will also get an WSM-07501 error similar to the following on the SOA server:

```
Failure in Oracle WSM Agent processRequest, category=security, function=agent.function.client, application=default, composite=TargetWSConnector, modelObj=TargetUserService, policy=oimcp/WS_CONNECTOR_OUTBOUND, policyVersion=1, assertionName=oracle.wsm.common.sdk.WSMException: FailedCheck :failure in security check at oracle.iam.connectors.genericws.soa.GenericWSOutboundPolicy.execute(GenericWSOutboundPolicy.java:119)
```

## Solution

Ensure that the target.payload.namespace property within the `<binding.ws>` tags of the Webservice connector that requires password decryption does not include quotation marks.
The following section describes known issues and workarounds associated with this release of the connector:

7.1 Connector Issues

The following are issues and workarounds associated with the connector:

- Section 7.1.1, "Request Datasets are Not Generated"
- Section 7.1.2, "Translations Missing for Some Connector Fields"

7.1.1 Request Datasets are Not Generated

The connector does not generate request datasets out of the box.

As a workaround, to create request datasets for this connector on Oracle Identity Manager 11g Release 1 PS1, see Configuring Requests in Oracle Fusion Middleware Developer’s Guide for Oracle Identity Manager at http://docs.oracle.com/cd/E21764_01/doc.1111/e14309:request.htm#OMDEV2856

7.1.2 Translations Missing for Some Connector Fields

Translations for some OOTB connector fields are not provided in local languages. Some affected fields are Address, Common Name, Department Number, Deprovisioning Date, Employee Number, End Date, Fax, and Generation Qualifier.

As a workaround, perform the instructions described in Section 3.2.8, "Localizing Field Labels in UI Forms."
Sample WSDL for ACME Webservice

In this guide, a target system that exposes webservice endpoint has been referred to as the target system. ACME Webservice is used as a sample target system to discuss the configurations and the connector objects.

In this appendix, a sample WSDL called ACME.wsdl is provided for use while performing the procedures described in this guide.

See Also: Oracle Identity Manager 11g Sample Assets page on Oracle Technology Network (OTN) for more information related to the sample:

http://www.oracle.com/technetwork/middleware/id-mgmt/overvew/oim-11g-assets-504842.html

<?xml version="1.0" encoding="UTF-8"?>
<definitions xmlns:soap="http://schemas.xmlsoap.org/wsdl/soap/">
  xmlns:tns="http://sample.acme.com"
  xmlns:xsd="http://www.w3.org/2001/XMLSchema"
  xmlns="http://schemas.xmlsoap.org/wsdl/"
  targetNamespace="http://sample.acme.com" name="My_Service">
  <types>
    <xsd:schema>
      <xsd:import namespace="http://sample.acme.com"
    </xsd:schema>
  </types>
  <message name="DeleteAccount">
    <part name="parameters" element="tns:DeleteAccount"/>
  </message>
  <message name="DeleteAccountResponse">
    <part name="parameters" element="tns:DeleteAccountResponse"/>
  </message>
  <message name="AccountSearch">
    <part name="parameters" element="tns:AccountSearch"/>
  </message>
  <message name="AccountSearchResponse">
    <part name="parameters" element="tns:AccountSearchResponse"/>
  </message>
  <message name="CreateAccount">
    <part name="parameters" element="tns:CreateAccount"/>
  </message>
  <message name="CreateAccountResponse">
    <part name="parameters" element="tns:CreateAccountResponse"/>
  </message>
</definitions>
<message name="UpdateAccount">
  <part name="parameters" element="tns:UpdateAccount"/>
</message>

<message name="UpdateAccountResponse">
  <part name="parameters" element="tns:UpdateAccountResponse"/>
</message>

<message name="Remove Role">
  <part name="parameters" element="tns:Remove Role"/>
</message>

<message name="Remove RoleResponse">
  <part name="parameters" element="tns:Remove RoleResponse"/>
</message>

<message name="LookupSearch">
  <part name="parameters" element="tns:LookupSearch"/>
</message>

<message name="LookupSearchResponse">
  <part name="parameters" element="tns:LookupSearchResponse"/>
</message>

<message name="UserRecordSearch">
  <part name="parameters" element="tns:UserRecordSearch"/>
</message>

<message name="UserRecordSearchResponse">
  <part name="parameters" element="tns:UserRecordSearchResponse"/>
</message>

<message name="MyLookup">
  <part name="parameters" element="tns:MyLookup"/>
</message>

<message name="MyLookupResponse">
  <part name="parameters" element="tns:MyLookupResponse"/>
</message>

<message name="Add Role">
  <part name="parameters" element="tns:Add Role"/>
</message>

<message name="Add RoleResponse">
  <part name="parameters" element="tns:Add RoleResponse"/>
</message>

<portType name="MySample">
  <operation name="DeleteAccount">
    <input message="tns:DeleteAccount"/>
    <output message="tns:DeleteAccountResponse"/>
  </operation>
  
  <operation name="AccountSearch">
    <input message="tns:AccountSearch"/>
    <output message="tns:AccountSearchResponse"/>
  </operation>
  
  <operation name="CreateAccount">
    <input message="tns:CreateAccount"/>
    <output message="tns:CreateAccountResponse"/>
  </operation>
  
  <operation name="UpdateAccount">
    <input message="tns:UpdateAccount"/>
    <output message="tns:UpdateAccountResponse"/>
  </operation>
  
  <operation name="Remove Role">
    <input message="tns:Remove Role"/>
    <output message="tns:Remove RoleResponse"/>
  </operation>
  
  <operation name="LookupSearch">
    <input message="tns:LookupSearch"/>
  </operation>
</portType>
<operation name="UserRecordSearch">
  <input message="tns:UserRecordSearch"/>
  <output message="tns:UserRecordSearchResponse"/>
</operation>

<operation name="MyLookup">
  <input message="tns:MyLookup"/>
  <output message="tns:MyLookupResponse"/>
</operation>

<operation name="Add Role">
  <input message="tns:Add Role"/>
  <output message="tns:Add RoleResponse"/>
</operation>

<portType>
  <binding name="MySamplePortBinding" type="tns:MySample">
    <soap:binding transport="http://schemas.xmlsoap.org/soap/http" style="document"/>
    <operation name="DeleteAccount">
      <soap:operation soapAction=""/>
      <input>
        <soap:body use="literal"/>
      </input>
      <output>
        <soap:body use="literal"/>
      </output>
    </operation>
    <operation name="AccountSearch">
      <soap:operation soapAction=""/>
      <input>
        <soap:body use="literal"/>
      </input>
      <output>
        <soap:body use="literal"/>
      </output>
    </operation>
    <operation name="CreateAccount">
      <soap:operation soapAction=""/>
      <input>
        <soap:body use="literal"/>
      </input>
      <output>
        <soap:body use="literal"/>
      </output>
    </operation>
    <operation name="UpdateAccount">
      <soap:operation soapAction=""/>
      <input>
        <soap:body use="literal"/>
      </input>
      <output>
        <soap:body use="literal"/>
      </output>
    </operation>
    <operation name="Remove Role">
      <soap:operation soapAction=""/>
      <input>
        <soap:body use="literal"/>
      </input>
      <output>
        <soap:body use="literal"/>
      </output>
    </operation>
  </binding>
</portType>
<soap:body use="literal"/>
</operation>

<operation name="LookupSearch">
<soap:operation soapAction=""/>
<input>
<soap:body use="literal"/>
</input>
<output>
<soap:body use="literal"/>
</output>
</operation>

<operation name="UserRecordSearch">
<soap:operation soapAction=""/>
<input>
<soap:body use="literal"/>
</input>
<output>
<soap:body use="literal"/>
</output>
</operation>

<operation name="MyLookup">
<soap:operation soapAction=""/>
<input>
<soap:body use="literal"/>
</input>
<output>
<soap:body use="literal"/>
</output>
</operation>

<operation name="Add Role">
<soap:operation soapAction=""/>
<input>
<soap:body use="literal"/>
</input>
<output>
<soap:body use="literal"/>
</output>
</operation>
</binding>

<service name="My_Service">
<port name="MySamplePort" binding="tns:MySamplePortBinding">
<soap:address location="http://10.232.9.11:7001/SampleWebservice/My_Service"/>
</port>
</service>
</definitions>
In this appendix, a sample Outbound Policy is provided for use while performing the procedures described in this guide.

Note: Ensure to put the content in a META-INF folder and then zip the META-INF folder and rename it as OutboundPolicy.zip.

```xml
<?xml version="1.0" encoding="utf-8"?>
xmlns:orawsp="http://schemas.oracle.com/ws/2006/01/policy"
xmlns:wsu="http://docs.oasis-open.org/wss/2004/01/oasis-200401-wss-wssecurity-utility-1.0.xsd"
Name="oimcp/WS_CONNECTOR_OUTBOUND" orawsp:attachTo="binding.client"
orawsp:category="security"
orawsp:description="OIM Webservices connector outbound policy. This policy does the outbound processing for target webservice invocation calls"
orawsp:local-optimization="off" orawsp:oraSmartDigest="1332670244"
orawsp:smartDigest="1332670244"
orawsp:smartDigests="1332670244V1_2,1332670244V1_2,359294128V1_5359294128V1_5359294128V1_5,"
orawsp:status="enabled" orawsp:versionCreator="weblogic"
orawsp:versionNumber="1"
orawsp:versionTime="1349693084088" wsu:Id="WS_CONNECTOR_OUTBOUND">
<custom:custom-executor xmlns:custom="http://schemas.oracle.com/ws/soa/custom"
orawsp:Enforced="true" orawsp:Silent="false"
orawsp:category="security/custom"
orawsp:name="WSConnectorOutboundAssertion">
<orawsp:bindings>
<orawsp:Implementation>oracle.iam.connectors.genericws.soa.GenericWSOutboundPolicy</orawsp:Implementation>
```
/orawsp:bindings>
</custom:custom-executor>
</wsp:Policy>
In this appendix, a sample WSDL called SecurityPolicy.wsdl is provided for use while performing the procedures described in this guide.

```xml
<interface.wsdl
    interface="http://xmlns.oracle.com/idm/identity/webservice/SPMLService#wsdl.1.interface{SPMLRequestPortType}"/>

<binding.ws
    port="http://xmlns.oracle.com/idm/identity/webservice/SPMLService#wsdl.endpoint{SPMLService/SPMLServiceProviderSoap}"
    location="SPMLServiceWrapper.wsdl" soapVersion="1.1">
    <wsp:PolicyReference URI="oracle/wss_username_token_policy"
        orawsp:category="security" orawsp:status="enabled"/>
    <property name="oracle.webservices.auth.username" type="xs:string" many="false" override="may">weblogic</property>
    <property name="oracle.webservices.auth.password" type="xs:string" many="false" override="may">weblogic1</property>
</binding.ws>
</reference>
```
In this appendix, the following sample XSDs of SPML and DSML are provided for use while performing the procedures described in this guide:

- Appendix D.1, "Sample SPML XSD"
- Appendix D.2, "Sample DSML XSD"

### D.1 Sample SPML XSD

The following is a sample SPML XSD:

```xml
<?xml version="1.0" encoding="UTF-8"?>
<!--*****************************************************-->
<!-- draft_pstc_SPMLv2_core_27.xsd -->
<!-- Draft schema for SPML v2.0 core capabilities. -->
<!-- Editors: -->
<!-- Jeff Bohren (Jeff_Bohren@bmc.com) -->
<!-- Copyright (C) The Organization for the Advancement of Structured Information Standards [OASIS] 2005. All Rights Reserved. -->
<!-- Reserved. -->
<!--*****************************************************-->
<schema targetNamespace="urn:oasis:names:tc:SPML:2:0"
xmlns="http://www.w3.org/2001/XMLSchema"
xmlns:xsd="http://www.w3.org/2001/XMLSchema"
xmlns:spml="urn:oasis:names:tc:SPML:2:0" elementFormDefault="qualified">

<complexType name="ExtensibleType">

<sequence>

<any namespace="##other" minOccurs="0" maxOccurs="unbounded" processContents="lax"/>

</sequence>

</complexType>

</schema>
```
<complexType name="ExecutionModeType">
<restriction base="string">
<enumeration value="synchronous"/>
<enumeration value="asynchronous"/>
</restriction>
</complexType>

<complexType name="CapabilityDataType">
<complexContent>
<extension base="spml:ExtensibleType">
<attribute name="mustUnderstand" type="boolean" use="optional"/>
<attribute name="capabilityURI" type="anyURI"/>
</extension>
</complexContent>
</complexType>

<complexType name="RequestType">
<complexContent>
<extension base="spml:ExtensibleType">
<attribute name="requestID" type="xsd:ID" use="optional"/>
<attribute name="executionMode" type="spml:ExecutionModeType" use="optional"/>
</extension>
</complexContent>
</complexType>

<complexType name="StatusCodeType">
<restriction base="string">
<enumeration value="success"/>
<enumeration value="failure"/>
<enumeration value="pending"/>
</restriction>
</complexType>
<simpleType name="ErrorCode">
<restriction base="string">
<enumeration value="malformedRequest"/>
<enumeration value="unsupportedOperation"/>
<enumeration value="unsupportedIdentifierType"/>
<enumeration value="noSuchIdentifier"/>
<enumeration value="customError"/>
<enumeration value="unsupportedExecutionMode"/>
<enumeration value="invalidContainment"/>
<enumeration value="noSuchRequest"/>
<enumeration value="unsupportedSelectionType"/>
<enumeration value="resultSetTooLarge"/>
<enumeration value="unsupportedProfile"/>
<enumeration value="invalidIdentifier"/>
<enumeration value="alreadyExists"/>
<enumeration value="containerNotEmpty"/>
</restriction>
</simpleType>

<simpleType name="ReturnDataType">
<restriction base="string">
<enumeration value="identifier"/>
<enumeration value="data"/>
<enumeration value="everything"/>
</restriction>
</simpleType>

<complexType name="ResponseType">
<complexContent>
<extension base="spml:ExtensibleType">
<sequence>
<element name="errorMessage" type="xsd:string" minOccurs="0" maxOccurs="unbounded"/>
</sequence>
<attribute name="status" type="spml:StatusCodeType" use="required"/>
<attribute name="requestID" type="xsd:ID" use="optional"/>
<attribute name="error" type="spml:ErrorCode" use="optional"/>
</extension>
</complexContent>
</complexType>
<complexContent>
  <extension base="spml:ExtensibleType">
    <attribute name="ID" type="string" use="optional"/>
  </extension>
</complexContent>

<complexType name="PSOIdentifierType">
  <complexContent>
    <extension base="spml:IdentifierType">
      <sequence>
        <element name="containerID" type="spml:PSOIdentifierType" minOccurs="0"/>
      </sequence>
      <attribute name="targetID" type="string" use="optional"/>
    </extension>
  </complexContent>
</complexType>

<complexType name="PSOType">
  <complexContent>
    <extension base="spml:ExtensibleType">
      <sequence>
        <element name="psoID" type="spml:PSOIdentifierType"/>
        <element name="data" type="spml:ExtensibleType" minOccurs="0"/>
        <element name="capabilityData" type="spml:CapabilityDataType" minOccurs="0" maxOccurs="unbounded"/>
      </sequence>
    </extension>
  </complexContent>
</complexType>

<complexType name="AddRequestType">
<complexType name="AddResponseType">
<complexContent>
<extension base="spml:ResponseType">
<sequence>
<element name="psos" type="spml:PSOType" minOccurs="0" />
</sequence>
</extension>
</complexContent>
</complexType>

<simpleType name="ModificationModeType">
<restriction base="string">
<enumeration value="add"/>
<enumeration value="replace"/>
<enumeration value="delete"/>
</restriction>
</simpleType>

<complexType name="NamespacePrefixMappingType">
<complexContent>
<extension base="spml:ExtensibleType">
<attribute name="prefix" type="string" use="required"/>
</extension>
</complexContent>
</complexType>
<attribute name="namespace" type="string" use="required"/>
</extension>
</complexContent>
</complexType>

<complexType name="QueryClauseType">
<complexContent>
<extension base="spml:ExtensibleType">
</extension>
</complexContent>
</complexType>

<complexType name="SelectionType">
<complexContent>
<extension base="spml:QueryClauseType">
<sequence>
<element name="namespacePrefixMap" type="spml:NamespacePrefixMappingType" minOccurs="0" maxOccurs="unbounded"/>
</sequence>
<attribute name="path" type="string" use="required"/>
<attribute name="namespaceURI" type="string" use="required"/>
</extension>
</complexContent>
</complexType>

<complexType name="ModificationType">
<complexContent>
<extension base="spml:ExtensibleType">
<sequence>
<element name="component" type="spml:SelectionType" minOccurs="0"/>
<element name="data" type="spml:ExtensibleType" minOccurs="0"/>
<element name="capabilityData" type="spml:CapabilityDataType" minOccurs="0" maxOccurs="unbounded"/>
</sequence>
<attribute name="modificationMode" type="spml:ModificationModeType" use="optional"/>
</extension>
</complexContent>
</complexType>
<complexType name="ModifyRequestType">
<complexContent>
<extension base="spml:RequestType">
<sequence>
<element name="psoID" type="spml:PSOIdentifierType"/>
<element name="modification" type="spml:ModificationType" maxOccurs="unbounded"/>
</sequence>
<attribute name="returnData" type="spml:ReturnDataType" use="optional" default="everything"/>
</extension>
</complexContent>
</complexType>

<complexType name="ModifyResponseType">
<complexContent>
<extension base="spml:ResponseType">
<sequence>
<element name="pso" type="spml:PSOType" minOccurs="0" minOccurs="0"/>
</sequence>
</extension>
</complexContent>
</complexType>

<complexType name="DeleteRequestType">
<complexContent>
<extension base="spml:RequestType">
<sequence>
<element name="psoID" type="spml:PSOIdentifierType"/>
</sequence>
<attribute name="recursive" type="xsd:boolean" use="optional" default="false"/>
</extension>
</complexContent>
</complexType>

<complexType name="LookupRequestType">
<complexContent>
<extension base="spml:RequestType">
<sequence>
<element name="psoID" type="spml:PSOIdentifierType"/>
</sequence>
.attribute name="returnData" type="spml:ReturnDataType" use="optional"
default="everything"/>
<extension/>
</complexContent>
</complexType>
<complexType name="LookupResponseType">
<complexContent>
<extension base="spml:ResponseType">
<sequence>
<element name="pso" type="spml:PSOType" minOccurs="0" />
</sequence>
</extension>
</complexContent>
</complexType>
<complexType name="SchemaType">
<complexContent>
<extension base="spml:ExtensibleType">
<sequence>
<annotation>
<documentation>Profile specific schema elements should be included here</documentation>
</annotation>
<element name="supportedSchemaEntity" type="spml:SchemaEntityRefType"
minOccurs="0" maxOccurs="unbounded"/>
</sequence>
<attribute name="ref" type="anyURI" use="optional"/>
</extension>
</complexContent>
</complexType>
<complexType name="SchemaEntityRefType">
<complexContent>
<extension base="spml:ExtensibleType">
  <attribute name="targetID" type="string" use="optional"/>
  <attribute name="entityName" type="string" use="optional"/>
  <attribute name="isContainer" type="xsd:boolean" use="optional"/>
</extension>
</complexContent>
</complexType>
<complexType name="CapabilityType">
  <complexContent>
    <extension base="spml:ExtensibleType">
      <sequence>
        <element name="appliesTo" type="spml:SchemaEntityRefType" minOccurs="0" maxOccurs="unbounded"/>
      </sequence>
      <attribute name="namespaceURI" type="anyURI"/>
      <attribute name="location" type="anyURI" use="optional"/>
    </extension>
  </complexContent>
</complexType>
<complexType name="CapabilitiesListType">
  <complexContent>
    <extension base="spml:ExtensibleType">
      <sequence>
        <element name="capability" type="spml:CapabilityType" minOccurs="0" maxOccurs="unbounded"/>
      </sequence>
    </extension>
  </complexContent>
</complexType>
<complexType name="TargetType">
  <complexContent>
    <extension base="spml:ExtensibleType">
      <sequence>
        <element name="schema" type="spml:SchemaType" maxOccurs="unbounded"/>
        <element name="capabilities" type="spml:CapabilitiesListType" minOccurs="0"/>
      </sequence>
    </extension>
  </complexContent>
</complexType>
<attribute name="targetID" type="string" use="optional"/>
<attribute name="profile" type="anyURI" use="optional"/>
</extension>
</complexContent>
</complexType>

<complexType name="ListTargetsRequestType">
<complexContent>
<extension base="spml:RequestType">
</extension>
<attribute name="profile" type="anyURI" use="optional"/>
</complexContent>
</complexType>

<complexType name="ListTargetsResponseType">
<complexContent>
<extension base="spml:ResponseType">
<sequence>
<element name="target" type="spml:TargetType" minOccurs="0" maxOccurs="unbounded"/>
</sequence>
</extension>
</complexContent>
</complexType>

<element name="select" type="spml:SelectionType"/>
<element name="addRequest" type="spml:AddRequestType"/>
<element name="addResponse" type="spml:AddResponseType"/>
<element name="modifyRequest" type="spml:ModifyRequestType"/>
<element name="modifyResponse" type="spml:ModifyResponseType"/>
<element name="deleteRequest" type="spml:DeleteRequestType"/>
<element name="deleteResponse" type="spml:ResponseType"/>
<element name="lookupRequest" type="spml:LookupRequestType"/>
<element name="lookupResponse" type="spml:LookupResponseType"/>
<element name="list TargetsRequest" type="spml:ListTargetsRequestType"/>
<element name="list TargetsResponse" type="spml:ListTargetsResponseType"/>
</schema>
D.2 Sample DSML XSD

The following is a sample DSML XSD:

```xml
<xsd:schema targetNamespace="urn:oasis:names:tc:DSML:2:0:core"
    xmlns="urn:oasis:names:tc:DSML:2:0:core"
    xmlns:xsd="http://www.w3.org/2001/XMLSchema"
    elementFormDefault="qualified">
  <!-- Copyright (C) The Organization for the Advancement of Structured Information Standards [OASIS] 2001. All Rights Reserved. -->
  <!-- DSML Requests -->
  <xsd:group name="DSMLRequests">
    <xsd:choice>
      <xsd:element name="authRequest" type="AuthRequest"/>
      <xsd:group ref="BatchRequests"/>
    </xsd:choice>
  </xsd:group>
  <!-- DSML Responses -->
  <xsd:group name="DSMLResponses">
    <xsd:choice>
      <xsd:element name="authResponse" type="LDAPResult"/>
      <xsd:element name="searchResultEntry" type="SearchResultEntry"/>
      <xsd:element name="searchResultReference" type="SearchResultReference"/>
      <xsd:element name="searchResultDone" type="LDAPResult"/>
      <xsd:element name="modifyResponse" type="LDAPResult"/>
      <xsd:element name="addResponse" type="LDAPResult"/>
      <xsd:element name="delResponse" type="LDAPResult"/>
    </xsd:choice>
  </xsd:group>
</xsd:schema>
```
<xsd:element name="modDNResponse" type="LDAPResult"/>
<xsd:element name="compareResponse" type="LDAPResult"/>
<xsd:element name="extendedResponse" type="ExtendedResponse"/>
<xsd:element name="errorResponse" type="ErrorResponse"/>
</xsd:choice>
</xsd:group>

<!--  *************** Batch Envelopes ********************* -->
<xsd:element name="batchRequest" type="BatchRequest"/>
<xsd:element name="batchResponse" type="BatchResponse"/>

<!-- **** Batch Request Envelope **** -->
<xsd:complexType name="BatchRequest">
<xsd:sequence>
<xsd:element name="authRequest" type="AuthRequest" minOccurs="0" maxOccurs="1"/>
<xsd:group ref="BatchRequests" minOccurs="0" maxOccurs="unbounded"/>
</xsd:sequence>
<xsd:attribute name="requestID" type="RequestID" use="optional"/>
<xsd:attribute name="processing" use="optional" default="sequential">
<xsd:simpleType>
<xsd:restriction base="xsd:string">
<xsd:enumeration value="sequential"/>
<xsd:enumeration value="parallel"/>
</xsd:restriction>
</xsd:simpleType>
</xsd:attribute>
<xsd:attribute name="responseOrder" use="optional" default="sequential">
<xsd:simpleType>
<xsd:restriction base="xsd:string">
<xsd:enumeration value="sequential"/>
<xsd:enumeration value="unordered"/>
</xsd:restriction>
</xsd:simpleType>
</xsd:attribute>
<xsd:attribute name="onError" use="optional" default="exit">
<xsd:simpleType>
<xsd:restriction base="xsd:string">
<xsd:enumeration value="resume"/>
</xsd:restriction>
</xsd:simpleType>
</xsd:attribute>
</xsd:complexType>
<xsd:enumeration value="exit"/>
</xsd:restriction>
</xsd:simpleType>
</xsd:attribute>
</xsd:complexType>

<!-- **** Batch Response Envelope **** -->
<xsd:complexType name="BatchResponse">
  <xsd:sequence>
    <xsd:group ref="BatchResponses" minOccurs="0" maxOccurs="unbounded"/>
  </xsd:sequence>
  <xsd:attribute name="requestID" type="RequestID" use="optional"/>
</xsd:complexType>

<!-- **** Batch Responses **** -->
<xsd:group name="BatchResponses">
  <xsd:choice>
    <xsd:element name="searchResponse" type="SearchResponse"/>
    <xsd:element name="authResponse" type="LDAPResult"/>
    <xsd:element name="modifyResponse" type="LDAPResult"/>
    <xsd:element name="addResponse" type="LDAPResult"/>
    <xsd:element name="delResponse" type="LDAPResult"/>
    <xsd:element name="modDNResponse" type="LDAPResult"/>
    <xsd:element name="compareResponse" type="LDAPResult"/>
    <xsd:element name="extendedResponse" type="ExtendedResponse"/>
    <xsd:element name="errorResponse" type="ErrorResponse"/>
  </xsd:choice>
</xsd:group>

<!-- **** Search Response **** -->
<xsd:complexType name="SearchResponse">
  <xsd:sequence>
    <xsd:element name="searchResultEntry" type="SearchResultEntry" minOccurs="0" maxOccurs="unbounded"/>
    <xsd:element name="searchResultReference" type="SearchResultReference" minOccurs="0" maxOccurs="unbounded"/>
    <xsd:element name="searchResultDone" type="LDAPResult"/>
  </xsd:sequence>
  <xsd:attribute name="requestID" type="RequestID" use="optional"/>
</xsd:complexType>
<!-- ***** DsmlDN ***** -->
<xsd:simpleType name="DsmlDN">
    <xsd:restriction base="xsd:string"/>
</xsd:simpleType>

<!-- ***** DsmlRDN ***** -->
<xsd:simpleType name="DsmlRDN">
    <xsd:restriction base="xsd:string"/>
</xsd:simpleType>

<!-- ***** Request ID ***** -->
<xsd:simpleType name="RequestID">
    <xsd:restriction base="xsd:string"/>
</xsd:simpleType>

<!-- ***** AttributeDescriptionValue ***** -->
<xsd:simpleType name="AttributeDescriptionValue">
    <xsd:restriction base="xsd:string">
        <xsd:pattern value="(((\[0-2]\.[0-9]+)+|\[a-zA-Z]+(\[a-zA-Z0-9\]-\d-\d-\d-\d))|([a-zA-Z0-9\d-\d-\d-\d])")/">
    </xsd:restriction>
</xsd:simpleType>

<xsd:simpleType name="NumericOID">
    <xsd:restriction base="xsd:string">
        <xsd:pattern value="(\[0-2]\.[0-9]+)+|([a-zA-Z]+(\[a-zA-Z0-9\]-\d-\d-\d-\d))|([a-zA-Z0-9\d-\d-\d-\d])"/>
    </xsd:restriction>
</xsd:simpleType>

<!-- ***** MAX Integer ***** -->
<xsd:simpleType name="MAXINT">
    <xsd:restriction base="xsd:unsignedInt">
        <xsd:maxInclusive value="2147483647"/>
    </xsd:restriction>
</xsd:simpleType>

<!-- **** DSML Value **** -->
<xsd:simpleType name="DsmlValue">
    <xsd:union memberTypes="xsd:string xsd:base64Binary xsd:anyURI"/>
</xsd:simpleType>

<!-- **** DSML Control **** -->
<xsd:complexType name="Control">
</xsd:complexType>
<xsd:element name="controlValue" type="xsd:anyType" minOccurs="0"/>
</xsd:sequence>
<xsd:attribute name="type" type="NumericOID" use="required"/>
<xsd:attribute name="criticality" type="xsd:boolean" use="optional" default="false"/>
</xsd:complexType>
<!-- **** DSML Filter **** -->
<xsd:complexType name="Filter">
<xsd:group ref="FilterGroup"/>
</xsd:complexType>
<xsd:group name="FilterGroup">
<xsd:sequence>
<xsd:choice>
<xsd:element name="and" type="FilterSet"/>
<xsd:element name="or" type="FilterSet"/>
<xsd:element name="not" type="Filter"/>
<xsd:element name="equalityMatch" type="AttributeValueAssertion"/>
<xsd:element name="substrings" type="SubstringFilter"/>
<xsd:element name="greaterOrEqual" type="AttributeValueAssertion"/>
<xsd:element name="lessOrEqual" type="AttributeValueAssertion"/>
<xsd:element name="present" type="AttributeDescription"/>
<xsd:element name="approxMatch" type="AttributeValueAssertion"/>
<xsd:element name="extensibleMatch" type="MatchingRuleAssertion"/>
</xsd:choice>
</xsd:sequence>
</xsd:group>
<xsd:complexType name="FilterSet">
<xsd:sequence>
<xsd:group ref="FilterGroup" minOccurs="0" maxOccurs="unbounded"/>
</xsd:sequence>
</xsd:complexType>
<xsd:complexType name="AttributeValueAssertion">
<xsd:sequence>
<xsd:element name="value" type="DsmlValue"/>
</xsd:sequence>
<xsd:attribute name="name" type="AttributeDescriptionValue" use="required"/>
</xsd:complexType>
<xsd:complexType name="AttributeDescription"/>
<xsd:attribute name="name" type="AttributeDescriptionValue" use="required"/>
</xsd:complexType>
<xsd:complexType name="SubstringFilter">
<xsd:sequence>
<xsd:element name="initial" type="DsmlValue" minOccurs="0" maxOccurs="unbounded"/>
<xsd:element name="any" type="DsmlValue" minOccurs="0" maxOccurs="unbounded"/>
<xsd:element name="final" type="DsmlValue" minOccurs="0" maxOccurs="unbounded"/>
</xsd:sequence>
<xsd:attribute name="name" type="AttributeDescriptionValue" use="required"/>
</xsd:complexType>
<xsd:complexType name="MatchingRuleAssertion">
<xsd:sequence>
<xsd:element name="value" type="DsmlValue"/>
</xsd:sequence>
<xsd:attribute name="dnAttributes" type="xsd:boolean" use="optional" default="false"/>
<xsd:attribute name="matchingRule" type="xsd:string" use="optional"/>
<xsd:attribute name="name" type="AttributeDescriptionValue" use="optional"/>
</xsd:complexType>
<!-- *************** DSML MESSAGE **************** -->
<xsd:complexType name="DsmlMessage">
<xsd:sequence>
<xsd:element name="control" type="Control" minOccurs="0" maxOccurs="unbounded"/>
</xsd:sequence>
<xsd:attribute name="requestID" type="RequestID" use="optional"/>
</xsd:complexType>
<!-- *************** LDAP RESULT ************************ -->
<xsd:simpleType name="LDAPResultCode">
<xsd:restriction base="xsd:string">
<xsd:enumeration value="success"/>
<xsd:enumeration value="operationsError"/>
<xsd:enumeration value="protocolError"/>
<xsd:enumeration value="timeLimitExceeded"/>
<xsd:enumeration value="sizeLimitExceeded"/>
<xsd:enumeration value="compareFalse"/>
<xsd:enumeration value="compareTrue"/>
</xsd:restriction>
</xsd:simpleType>
<xsd:enumeration value="authMethodNotSupported"/>
<xsd:enumeration value="strongAuthRequired"/>
<xsd:enumeration value="referral"/>
<xsd:enumeration value="adminLimitExceeded"/>
<xsd:enumeration value="unavailable CriticalExtension"/>
<xsd:enumeration value="confidentiality Required"/>
<xsd:enumeration value="saslBindInProgress"/>
<xsd:enumeration value="noSuchAttribute"/>
<xsd:enumeration value="undefinedAttributeType"/>
<xsd:enumeration value="inappropriateMatching"/>
<xsd:enumeration value="constraintViolation"/>
<xsd:enumeration value="attributeOrValueExists"/>
<xsd:enumeration value="invalidAttributeSyntax"/>
<xsd:enumeration value="noSuchObject"/>
<xsd:enumeration value="aliasProblem"/>
<xsd:enumeration value="invalidDNSyntax"/>
<xsd:enumeration value="aliasDereferencingProblem"/>
<xsd:enumeration value="inappropriateAuthentication"/>
<xsd:enumeration value="invalidCredentials"/>
<xsd:enumeration value="insufficientAccessRights"/>
<xsd:enumeration value="busy"/>
<xsd:enumeration value="unavailable"/>
<xsd:enumeration value="unwillingToPerform"/>
<xsd:enumeration value="loopDetect"/>
<xsd:enumeration value="namingViolation"/>
<xsd:enumeration value="objectClassViolation"/>
<xsd:enumeration value="notAllowedOnNonLeaf"/>
<xsd:enumeration value="notAllowedOnRDN"/>
<xsd:enumeration value="entryAlreadyExists"/>
<xsd:enumeration value="objectClassModsProhibited"/>
<xsd:enumeration value="affectMultipleDSAs"/>
<xsd:enumeration value="other"/>
</xsd:restriction>
</xsd:simpleType>
</xsd:complexType>
<xsd:complexType name="ResultCode">
<xsd:attribute name="code" type="xsd:int" use="required"/>
<xsd:attribute name="descr" type="LDAPResultCode" use="optional"/>
</xsd:complexType>
<xsd:complexType name="LDAPResult">
<xsd:complexContent>
<xsd:extension base="DsmlMessage">
<xsd:sequence>
<xsd:element name="resultCode" type="ResultCode"/>
<xsd:element name="errorMessage" type="xsd:string" minOccurs="0"/>
<xsd:element name="referral" type="xsd:anyURI" minOccurs="0" maxOccurs="unbounded"/>
</xsd:sequence>
<xsd:attribute name="matchedDN" type="DsmlDN" use="optional"/>
</xsd:extension>
</xsd:complexContent>
</xsd:complexType>
<xsd:complexType name="ErrorResponse">
<xsd:sequence>
<xsd:element name="message" type="xsd:string" minOccurs="0"/>
<xsd:element name="detail" minOccurs="0">
<xsd:complexType>
<xsd:sequence>
<xsd:any/>
</xsd:sequence>
</xsd:complexType>
</xsd:element>
</xsd:sequence>
<xsd:attribute name="requestID" type="RequestID" use="optional"/>
<xsd:attribute name="type">
<xsd:simpleType>
<xsd:restriction base="xsd:string">
<xsd:enumeration value="notAttempted"/>
<xsd:enumeration value="couldNotConnect"/>
<xsd:enumeration value="connectionClosed"/>
<xsd:enumeration value="malformedRequest"/>
<xsd:enumeration value="gatewayInternalError"/>
<xsd:enumeration value="authenticationFailed"/>
<xsd:enumeration value="unresolvableURI"/>
<xsd:enumeration value="other"/>
</xsd:restriction>
</xsd:simpleType>
</xsd:attribute>
</xsd:complexType>
<xsd:element>
</xsd:sequence>
<xsd:attribute name="requestID" type="RequestID" use="optional"/>
<xsd:attribute name="type">
<xsd:simpleType>
<xsd:restriction base="xsd:string">
<xsd:enumeration value="notAttempted"/>
<xsd:enumeration value="couldNotConnect"/>
<xsd:enumeration value="connectionClosed"/>
<xsd:enumeration value="malformedRequest"/>
<xsd:enumeration value="gatewayInternalError"/>
<xsd:enumeration value="authenticationFailed"/>
<xsd:enumeration value="unresolvableURI"/>
<xsd:enumeration value="other"/>
</xsd:restriction>
</xsd:simpleType>
</xsd:attribute>
</xsd:element>
</xsd:sequence>
<xsd:attribute name="matchedDN" type="DsmlDN" use="optional"/>
<xsd:complexType name="AuthRequest">
  <xsd:complexContent>
    <xsd:extension base="DsmlMessage">
      <xsd:attribute name="principal" type="xsd:string" use="required"/>
    </xsd:extension>
  </xsd:complexContent>
</xsd:complexType>

<!-- *************** Search ********************* -->
<xsd:complexType name="AttributeDescriptions">
  <xsd:sequence minOccurs="0" maxOccurs="unbounded">
    <xsd:element name="attribute" type="AttributeDescription"/>
  </xsd:sequence>
</xsd:complexType>

<xsd:complexType name="SearchRequest">
  <xsd:complexContent>
    <xsd:extension base="DsmlMessage">
      <xsd:sequence>
        <xsd:element name="filter" type="Filter"/>
        <xsd:element name="attributes" type="AttributeDescriptions" minOccurs="0"/>
      </xsd:sequence>
      <xsd:attribute name="dn" type="DsmlDN" use="required"/>
      <xsd:attribute name="scope" use="required">
        <xsd:simpleType>
          <xsd:restriction base="xsd:string">
            <xsd:enumeration value="baseObject"/>
            <xsd:enumeration value="singleLevel"/>
            <xsd:enumeration value="wholeSubtree"/>
          </xsd:restriction>
        </xsd:simpleType>
      </xsd:attribute>
      <xsd:attribute name="derefAliases" use="required"/>
    </xsd:extension>
  </xsd:complexContent>
</xsd:complexType>
<xsd:restriction base="xsd:string">
    <xsd:enumeration value="neverDerefAliases"/>
    <xsd:enumeration value="derefInSearching"/>
    <xsd:enumeration value="derefFindingBaseObj"/>
    <xsd:enumeration value="derefAlways"/>
</xsd:restriction>
</xsd:simpleType>
</xsd:attribute>

<xsd:attribute name="sizeLimit" type="MAXINT" use="optional" default="0"/>
<xsd:attribute name="timeLimit" type="MAXINT" use="optional" default="0"/>
<xsd:attribute name="typesOnly" type="xsd:boolean" use="optional" default="false"/>
</xsd:extension>
</xsd:complexContent>
</xsd:complexType>

<!-- ***** Search Result Entry ***** -->
<xsd:complexType name="SearchResultEntry">
    <xsd:complexContent>
        <xsd:extension base="DsmlMessage">
            <xsd:sequence>
                <xsd:element name="attr" type="DsmlAttr" minOccurs="0" maxOccurs="unbounded"/>
            </xsd:sequence>
            <xsd:attribute name="dn" type="DsmlDN" use="required"/>
        </xsd:extension>
    </xsd:complexContent>
</xsd:complexType>
<xsd:complexType name="DsmlAttr">
    <xsd:sequence>
        <xsd:element name="value" type="DsmlValue" minOccurs="0" maxOccurs="unbounded"/>
    </xsd:sequence>
    <xsd:attribute name="name" type="AttributeDescriptionValue" use="required"/>
</xsd:complexType>
<xsd:complexType name="DsmlModification">
    <xsd:sequence>
        <xsd:element name="value" type="DsmlValue" minOccurs="0" maxOccurs="unbounded"/>
    </xsd:sequence>
    <xsd:attribute name="name" type="AttributeDescriptionValue" use="required"/>
</xsd:complexType>
<xsd:complexType name="DsmlModification">
<xsd:complexType name="DsmlModification">
  <xsd:complexContent>
    <xsd:extension base="DsmlElement">
      <xsd:attribute name="operation" use="required">
        <xsd:simpleType>
          <xsd:restriction base="xsd:string">
            <xsd:enumeration value="add"/>
            <xsd:enumeration value="delete"/>
            <xsd:enumeration value="replace"/>
          </xsd:restriction>
        </xsd:simpleType>
      </xsd:attribute>
    </xsd:extension>
  </xsd:complexContent>
</xsd:complexType>

<!--  *************** ADD ********************* -->
<xsd:complexType name="AddRequest">
  <xsd:complexContent>
    <xsd:extension base="DsmlMessage">
      <xsd:sequence>
        <xsd:element name="add" type="DsmlModification" minOccurs="0" maxOccurs="unbounded"/>
      </xsd:sequence>
    </xsd:extension>
  </xsd:complexContent>
</xsd:complexType>

<!--  *************** MODIFY ********************* -->
<xsd:complexType name="ModifyRequest">
  <xsd:complexContent>
    <xsd:extension base="DsmlMessage">
      <xsd:sequence>
        <xsd:element name="modification" type="DsmlModification" minOccurs="0" maxOccurs="unbounded"/>
      </xsd:sequence>
      <xsd:attribute name="dn" type="DsmlDN" use="required"/>
    </xsd:extension>
  </xsd:complexContent>
</xsd:complexType>

<!--  *************** REMOVE ********************* -->
<xsd:complexType name="RemoveRequest">
  <xsd:complexContent>
    <xsd:extension base="DsmlMessage">
      <xsd:sequence>
        <xsd:element name="remove" type="DsmlModification" minOccurs="0" maxOccurs="unbounded"/>
      </xsd:sequence>
      <xsd:attribute name="dn" type="DsmlDN" use="required"/>
    </xsd:extension>
  </xsd:complexContent>
</xsd:complexType>

<!--  *************** REPLACE ********************* -->
<xsd:complexType name="ReplaceRequest">
  <xsd:complexContent>
    <xsd:extension base="DsmlMessage">
      <xsd:sequence>
        <xsd:element name="replace" type="DsmlModification" minOccurs="0" maxOccurs="unbounded"/>
      </xsd:sequence>
      <xsd:attribute name="dn" type="DsmlDN" use="required"/>
    </xsd:extension>
  </xsd:complexContent>
</xsd:complexType>

<!--  *************** SEARCH RESULT REFERENCE ********************* -->
<xsd:complexType name="SearchResultReference">
  <xsd:complexContent>
    <xsd:extension base="DsmlMessage">
      <xsd:sequence>
        <xsd:element name="ref" type="xsd:anyURI" maxOccurs="unbounded"/>
      </xsd:sequence>
    </xsd:extension>
  </xsd:complexContent>
</xsd:complexType>

<!--  *************** ADD ********************* -->
<xsd:complexType name="AddRequest">
  <xsd:complexContent>
    <xsd:extension base="DsmlMessage">
      <xsd:sequence>
        <xsd:element name="add" type="DsmlModification" minOccurs="0" maxOccurs="unbounded"/>
      </xsd:sequence>
      <xsd:attribute name="dn" type="DsmlDN" use="required"/>
    </xsd:extension>
  </xsd:complexContent>
</xsd:complexType>
<xsd:extension base="DsmlMessage">
  <xsd:sequence>
    <xsd:element name="attr" type="DsmlAttr" minOccurs="0" maxOccurs="unbounded"/>
  </xsd:sequence>
  <xsd:attribute name="dn" type="DsmlDN" use="required"/>
</xsd:extension>
</xsd:complexContent>
</xsd:complexType>

<!-- *************** DELETE ********************* -->
<xsd:complexType name="DelRequest">
  <xsd:complexContent>
    <xsd:extension base="DsmlMessage">
      <xsd:attribute name="dn" type="DsmlDN" use="required"/>
    </xsd:extension>
  </xsd:complexContent>
</xsd:complexType>

<!-- *************** MODIFY DN ********************* -->
<xsd:complexType name="ModifyDNRequest">
  <xsd:complexContent>
    <xsd:extension base="DsmlMessage">
      <xsd:attribute name="dn" type="DsmlDN" use="required"/>
      <xsd:attribute name="newrdn" type="DsmlRDN" use="required"/>
      <xsd:attribute name="deleteoldrdn" type="xsd:boolean" use="optional" default="true"/>
      <xsd:attribute name="newSuperior" type="DsmlDN" use="optional"/>
    </xsd:extension>
  </xsd:complexContent>
</xsd:complexType>

<!-- ************* COMPARE ******************** -->
<xsd:complexType name="CompareRequest">
  <xsd:complexContent>
    <xsd:extension base="DsmlMessage">
      <xsd:sequence>
        <xsd:element name="assertion" type="AttributeValueAssertion"/>
      </xsd:sequence>
      <xsd:attribute name="dn" type="DsmlDN" use="required"/>
    </xsd:extension>
  </xsd:complexContent>
</xsd:complexType>
<!-- ***** ABANDON ***** -->
<xsd:complexType name="AbandonRequest">
  <xsd:complexContent>
    <xsd:extension base="DsmlMessage">
      <xsd:attribute name="abandonID" type="RequestID" use="required"/>
    </xsd:extension>
  </xsd:complexContent>
</xsd:complexType>

<!-- ************* EXTENDED OPERATION ******************** -->
<xsd:complexType name="ExtendedRequest">
  <xsd:complexContent>
    <xsd:extension base="DsmlMessage">
      <xsd:sequence>
        <xsd:element name="requestName" type="NumericOID"/>
        <xsd:element name="requestValue" type="xsd:anyType" minOccurs="0"/>
      </xsd:sequence>
    </xsd:extension>
  </xsd:complexContent>
</xsd:complexType>

<xsd:complexType name="ExtendedResponse">
  <xsd:complexContent>
    <xsd:extension base="LDAPResult">
      <xsd:sequence>
        <xsd:element name="responseName" type="NumericOID" minOccurs="0"/>
        <xsd:element name="response" type="xsd:anyType" minOccurs="0"/>
      </xsd:sequence>
    </xsd:extension>
  </xsd:complexContent>
</xsd:complexType>

<!-- ********************END base SCHEMA ********************* -->
</xsd:schema>
A
architecture, 1-3

C
certified components, 1-2
e changing input locale, 3-11
c child form data
adding, 5-29
clearing server cache, 3-11
configuring
 connector for multiple installations of the target
 system, 5-32
 SOA composite, 2-4
target system as trusted source, 4-3
 webservice security policy, 5-5
connection pooling, 3-10
 connector
 preinstallation, 2-1
 security, 5-1
 connector architecture, 1-3
 connector customization, 5-1
 Connector Installer, 3-1
 connector testing, 6-1
 custom attributes
 provisioning, 5-12
 reconciliation, 5-15
 custom child forms, 5-24
 customizing connector, 5-1

D
defining
 IT resources, 3-3
direct provisioning, 4-8
disabling accounts, 1-11

E
enabling accounts, 1-11
enabling logging, 3-8
Enterprise Manager, 2-51
errors, 6-1
errors and faults, 2-41
exclusion lists, 5-36

F
filtered reconciliation, 4-2

G
globalization features, 1-2

I
input locale, changing, 3-11
installation, 3-1
installing connector, 3-1, 3-6
 preinstallation, 2-1
 prerequisites, 2-1
 issues, 7-1
 IT resources
 defining, 3-3
 parameters, 3-3

J
 JDeveloper, 2-1

L
limited reconciliation, 4-2
logging enabling, 3-8
lookup
 configuration, 1-7
 provisioning attribute mapping, 1-8
 reconciliation attribute mapping, 1-8
 user configuration, 1-7
 lookup field synchronization, 1-6, 4-3
 lookup fields, 1-6, 4-3

M
multilanguage support, 1-2

P
parameters of IT resources, 3-3
 preinstallation, 2-1
 build utility, 2-2
 problems, 6-1, 7-1
 provisioning, 4-7, 4-12
connector objects
  module, 1-11
  custom attributes, 5-12
  direct provisioning, 4-8
  fields, 1-11
  identity fields, 1-11
  provisioning triggered by policy changes, 4-7
  request-based provisioning, 4-7
  user provisioning functions, 1-11

R
reconciliation
  batched, 4-2
  connector objects, 1-9
  custom attributes, 5-15
  full, 4-2
  limited, 4-2
  scheduled tasks, 4-3
reconciliation, user attributes, 1-9
request-based provisioning, 4-8
requirements for deploying, 1-2

S
scheduled tasks, 4-3
security, 5-1
  importing SSL certificate, 5-11
  outbound policy, 5-2
  passcode guidelines, 5-2
  passing credentials, 5-5, 5-7
  passwords, 5-2
  WSS policy, 5-5
server cache, clearing, 3-11
SOA composite, 2-1
  creating, 2-4
  deploying, 2-51
  partner link, 2-5
  testing, 2-51
stages of connector deployment
  postinstallation, 3-6
supported
  languages, 1-2
  releases of Oracle Identity Manager, 1-2
  target systems, 1-2

T
target resource reconciliation, 1-9
target system, multiple installations, 5-32
target systems supported, 1-2
testing, 6-1
timestamp attribute
  mapping, 5-30
  transformation, 5-35
  troubleshooting, 6-1

V
validation, 5-32