Oracle® Identity Manager

Tools Reference

Release 9.1.0.1

E14067-01

February 2009

ORACLE®

Oracle Identity Manager Tools Reference, Release 9.1.0.1

E14067-01

Primary Author: Debapriya Datta

Contributing Author: Lyju Vadassery

Contents

Preface

This guide provides information that can be applied when creating adapters, assigning and removing process tasks, and compiling, importing, and exporting adapters.

Audience

This guide is intended for users of the Oracle Identity Manager Design Console. The information in this guide can be used by developers, who are responsible for creating adapters, and by Oracle Identity Manager administrators, who attach these adapters to process tasks, forms that are predefined in Oracle Identity Manager, and user-created forms.

Documentation Accessibility

Our goal is to make Oracle products, services, and supporting documentation accessible to all users, including users that are disabled. To that end, our documentation includes features that make information available to users of assistive technology. This documentation is available in HTML format, and contains markup to facilitate access by the disabled community. Accessibility standards will continue to evolve over time, and Oracle is actively engaged with other market-leading technology vendors to address technical obstacles so that our documentation can be accessible to all of our customers. For more information, visit the Oracle Accessibility Program Web site at http://www.oracle.com/accessibility/.

Accessibility of Code Examples in Documentation

Screen readers may not always correctly read the code examples in this document. The conventions for writing code require that closing braces should appear on an otherwise empty line; however, some screen readers may not always read a line of text that consists solely of a bracket or brace.

Accessibility of Links to External Web Sites in Documentation

This documentation may contain links to Web sites of other companies or organizations that Oracle does not own or control. Oracle neither evaluates nor makes any representations regarding the accessibility of these Web sites.

TTY Access to Oracle Support Services

To reach AT&T Customer Assistants, dial 711 or 1.800.855.2880. An AT&T Customer Assistant will relay information between the customer and Oracle Support Services at 1.800.223.1711. Complete instructions for using the AT&T relay services are available at http://www.consumer.att.com/relay/tty/standard2.html. After the AT&T Customer Assistant contacts Oracle Support Services, an Oracle Support

Services engineer will handle technical issues and provide customer support according to the Oracle service request process.

Related Documents

For more information, see the other documents in the Oracle Identity Manager documentation set for this release.

Documentation Updates

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

`http://www.oracle.com/technology/documentation/index.html`

Conventions

The following text conventions are used in this document:

Convention	Meaning
boldface	Boldface type indicates graphical user interface elements associated with an action, or terms defined in text or the glossary.
italic	Italic type indicates book titles, emphasis, or placeholder variables for which you supply particular values.
`monospace`	Monospace type indicates commands within a paragraph, URLs, code in examples, text that appears on the screen (or text that you enter), and names of files, directories, attributes, and parameters.

Online Help

Oracle Identity Manager comes equipped with a suite of online Help systems, designed to assist you with locating relevant information when you need it. The various books within the Help system library are available using the Help menu (located in the main screen). To open a particular book, select it from the Help menu.

In addition, Oracle Identity Manager provides content-sensitive Help for a number of forms. The contents of the content-sensitive Help topic reflect the form that is active within the Oracle Identity Manager main screen. To launch a context-sensitive Help topic for a particular form, ensure that it is active. Then, press F1.

Note: In addition to context-sensitive Help, the Oracle Identity Manager online Help system can also be invoked by pressing the F1 key.

1 Introduction to the Integration

This chapter describes the challenges of integrating access rights management applications with other software solutions, and the solution that Oracle Identity Manager offers. It discusses the following topics:

1.1 The Integration Problem

An access rights management application cannot be successful today unless it can be integrated with other software solutions. Not only are there many resources, but also there is no single integration standard for connecting to these resources.

One way to tackle this challenge is by using the common functionality that is supported by all the integrations. To do this, you need developers who can write this code. In addition, every time an existing software resource is modified, or a new one is added, you must write more code.

1.2 The Oracle Identity Manager Solution

The Adapter Factory is a code-generation tool provided by Oracle Identity Manager. It helps you create Java classes, known as adapters.

An adapter provides the following benefits:

- It extends the internal logic and functionality of Oracle Identity Manager.
- It interfaces with any software resource, by connecting to that resource by using the API of the resource.
- It enables the integration between Oracle Identity Manager and an external system.
- It can be generated without manually writing code.
- It is lightweight and specific to your needs.
- It can be maintained easily because all of the definitions for the adapter are stored in a repository. This repository can be edited through a GUI.
- One Oracle Identity Manager user can retain the domain knowledge about the integration, while another user can maintain the adapter.

- It can be modified and upgraded efficiently.

1.3 About Adapters

As stated earlier, an adapter is a Java class created by an Oracle Identity Manager user through the Adapter Factory. In addition, it is an aggregation of atomic function calls in a logical flow. These atomic steps are known as adapter tasks.

> **Note:** Oracle Identity Manager can connect to external systems such as databases and directory servers by using Java APIs for JDBC and LDAP. In addition, for all other APIs, such as C, C++, VB, and COM/DCOM, you can create a Java wrapper so that Oracle Identity Manager can communicate with the API directly.

There are five types of adapters:

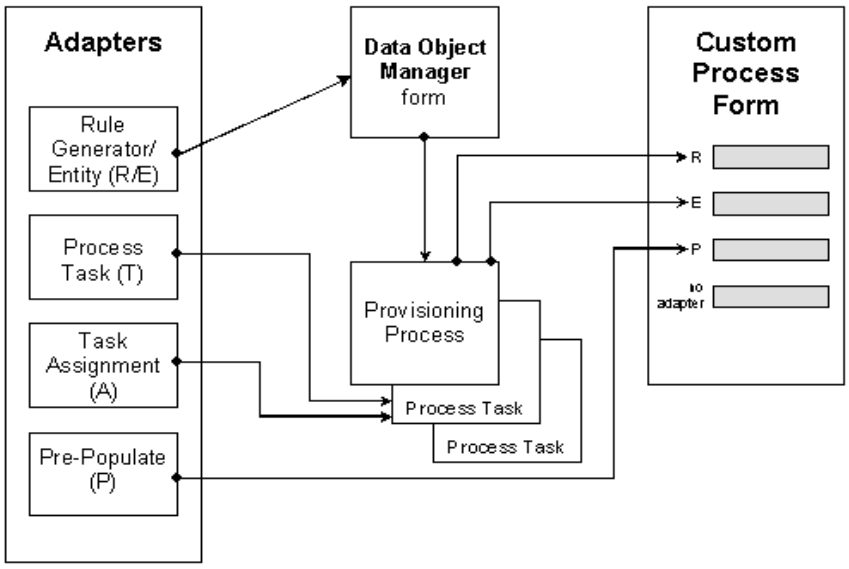
- A process task adapter, which allows Oracle Identity Manager to automate the completion of a process task.
- A task assignment adapter, which enables Oracle Identity Manager to automate the assignment of a process task to a user or group.
- A rule generator, which incorporates business rules to the fields of either an Oracle Identity Manager form or a user-defined form (created by using the Form Designer form), so these fields can be populated automatically and saved to the Oracle Identity Manager database.

> **Note:** For more information about the Form Designer form, refer to *Oracle Identity Manager Design Console Guide*

- A pre-populate adapter, which is a specific type of rule generator adapter that can be attached to a user-created form field. The data generated by this type of adapter can appear either automatically or manually. In addition, it uses criteria that enable Oracle Identity Manager to determine which pre-populate adapter will be applied to the designated form field. It then populates the designated form field without saving this information to the Oracle Identity Manager database.
- An entity adapter, which is attached to an Oracle Identity Manager or user-created form field. Oracle Identity Manager triggers an entity adapter on preinsert, preupdate, predelete, postinsert, postupdate, or postdelete. Once this occurs, the field to which the adapter is attached is populated automatically and saved to the Oracle Identity Manager database.

Figure 1–1 illustrates the functionality of five different types of adapters.

Figure 1–1 Adapter Functionality



These five types of adapters will be explained in greater detail in the chapters that follow.

1.4 Tabs of the Adapter Factory Form

The Adapter Factory form in the Design Console contains the following tabs:

- Adapter Tasks
- Execution Schedule
- Resources
- Variable List
- Usage Lookup
- Responses

Each of these tabs is explained in greater detail in the following sections.

1.4.1 Adapter Tasks

In the Adapter Tasks tab, you can create and manage the atomic function calls of an adapter. These function calls are known as adapter tasks.

1.4.2 Execution Schedule

The Execution Schedule tab lets you specify when you want Oracle Identity Manager to trigger a rule generator or an entity adapter. You can schedule Oracle Identity Manager to run a rule generator (Adapter Type *R*) on preinsert and/or preupdate. You can also configure Oracle Identity Manager to execute an entity adapter (Adapter Type *E*) on preinsert, preupdate, predelete, postinsert, postupdate, or postdelete.

Caution: **Process task adapters and task assignment adapters, which are attached to process tasks, are triggered once the process task's status becomes *Pending*. Therefore, you do not specify when Oracle Identity Manager will trigger these types of adapters, Oracle Identity Manager disables the Execution Schedule tab for them.**

Also, because Oracle Identity Manager always triggers pre-populate adapters on preinsert, Oracle Identity Manager disables the check boxes of this tab for pre-populate adapters.

1.4.3 Resources

From the Resources tab, you can:

- Click the Java APIs subtab to see the Java APIs that are being used by the adapter.
- Click the Other subtab to document a non-Java API file to the adapter, if necessary.

1.4.4 Variable List

From the Variable List tab, you can:

- Create, modify, and delete adapter variables.
- Set the data type and provide a description for each variable.
- Map an adapter variable to a literal or an adapter reference. You can also postpone the mapping until it is attached to a process task or a form field.

You also can resolve the value of the adapter variable at run time, when it is attached to a process task and the process task is run. As a result, process-specific data is available to map to this variable.

1.4.5 Usage Lookup

For a process task or task assignment adapter, the Usage Lookup tab displays the process task to which the adapter is attached, as well as the process of which this process task is a member.

For a rule generator or entity adapter, this tab shows the Oracle Identity Manager form and associated data object to which the adapter is attached. In addition, it displays the execution schedule of the adapter, along with a sequence number that represents the order in which Oracle Identity Manager will trigger the adapter.

For a pre-populate adapter, this tab displays the user-defined form and form field to which the adapter is attached. Also, it shows the pre-populate rule that is associated with the adapter.

Note: For more information about the different types of adapters, refer to the Chapter 3, "Creating Adapters" on page 3-1.

1.4.6 Responses

The Responses tab is used for defining meaningful responses to the process task. These responses depend on the execution result of the adapter. The various error messages returned by the external system can then be mapped to these responses in a way that they make sense in the context of the process task. On attaching the adapter to a process task, the status bucket, which consists of Pending, Completed, and Rejected, of

the process task (and subsequently the Object status) can be set, based on the adapter response code.

Tip: Oracle Identity Manager enables the Responses tab only for process task adapters. If an adapter is a task assignment, rule generator, pre-populate, or entity adapter, then Oracle Identity Manager disables this tab.

2

Getting Started

This chapter describes the configuration of Oracle Identity Manager and installation of the Remote Manager. It contains the following topics:

2.1 Overview of Oracle Identity Manager Configuration

To construct adapter tasks, ensure that Oracle Identity Manager has access to the target API JAR files and third-party applications to which you want to connect.

When your adapter uses Java tasks, you must configure Oracle Identity Manager to find the appropriate Java APIs. To do this, you must place the .jar files that contain these APIs into the JavaTasks subdirectory of the *OIM_HOME*`/xellerate` folder path, such as `C:\oracle\Xellerate\JavaTasks`. Then, you can access the Java classes associated with these Java APIs and use them in the Java task you are creating.

Sometimes, instead of directly communicating with the third-party system, Oracle Identity Manager must use an Oracle Identity Manager component that acts like a proxy. This component is known as Remote Manager.

The Remote Manager is used for:

- Invoking nonremotable APIs through Oracle Identity Manager
- Invoking APIs that do not support Secure Sockets Layer (SSL) over secure connections

The procedures in the following sections show you how to:

- Configure Oracle Identity Manager to reference JAR and class files for Java tasks.
- Configure the Remote Manager.

See Also:

"Creating a Java Task" on page 3-7 for more information about Java tasks

Oracle Identity Manager Administrative and User Console Guide to learn more about the Remote Manager

2.2 Configuring Oracle Identity Manager to Reference JAR and Class Files

To configure Oracle Identity Manager to reference JAR and class files:

1. Open the JavaTasks subdirectory, which can be found within the *OIM_HOME*/xellerate folder path. For example, C:\oracle\Xellerate\JavaTasks.
2. Place the JAR file or files into this subdirectory. You can then use these files to create Java tasks within an adapter without restarting the server.

2.3 Installing the Remote Manager

To configure the Remote Manager, follow the instructions described in Oracle Identity Manager installation guide for the application server that you use.

2.4 Adding the Trust Relation

To import a trusted certificate, at the command prompt, navigate to the following location:

JAVA_HOME\jre\lib\security

Enter the command:

```
JAVA_HOME/jre/bin/keytool -import -file
XLREMOTE_HOME\xlremote\config\xlserver.cert -keystore cacerts -trustcacerts -
alias rmtrust1 -storepass changeit

Owner: CN=Customer, OU=Customer, O=Customer, L=City, ST=NY, C=US
Issuer: CN=Customer, OU=Customer, O=Customer, L=City, ST=NY, C=US
Serial number: 41dee35a
Valid from: Fri Jan 07 11:30:34 PST 2005 until: Mon Jan 05 11:30:34 PST 2015
Certificate fingerprints:
         MD5:  B0:F2:33:C8:69:E4:25:A3:CB:59:E8:51:27:EE:5C:52
         SHA1: 3D:6A:6D:14:33:B5:5C:19:85:CC:EE:77:7F:7F:22:1D:56:48:47:4D
Trust this certificate? [no]:  yes
```

The certificate is added to the keystore.

3

Creating Adapters

This chapter describes how to create and define adapters. It contains the following topics:

3.1 Overview of Adapter Creation

Now that you have set up Oracle Identity Manager, you are ready to create the five types of Oracle Identity Manager adapters. These are:

- Process task adapters
- Task assignment adapters
- Rule generators
- Pre-populate adapters
- Entity adapters

> **See Also:** "About Adapters" on page 1-2 for a detailed description of these adapters

3.2 Defining Adapters

To define an adapter:

1. Log in to Oracle Identity Manager.
2. Open the Adapter Factory form. This form is in the Development Tools folder in the Design Console.

3. In the Adapter Name field, enter the name of the adapter, for example, `Create Solaris User`.

   Note: Although the adapter name can contain special characters, Oracle recommends that you do not use them because there might be run-time errors.

4. Double-click the Adapter Type lookup field.

   The Lookup window is displayed, displaying the five types of Oracle Identity Manager adapters. These are:

   - Process Task
   - Rule Generator
   - Pre-populate Rule Generator
   - Entity
   - Task Assignment

5. To enable the adapter to automate a process task, select **Process Task (T)**.

   To incorporate business rules into an Oracle Identity Manager or user-defined form field, select **Rule Generator (R)**. For example, for the User ID field of a form, you can configure Oracle Identity Manager to concatenate the initial letter of the user's first name with the user's last name.

   You can attach a type of rule generator adapter to a user-created form field, so that it can:

   - Display the data, which is generated by the adapter, automatically or manually.
   - Use criteria that enable Oracle Identity Manager to determine which adapter is applied to the designated form field.

   To attach the adapter to an Oracle Identity Manager or user-defined form field, and have Oracle Identity Manager trigger the adapter on preinsert, preupdate, predelete, postinsert, postupdate, or postdelete, select **Entity (E)**.

   To allow the adapter to automate the allocation of a process task to a user or group, select **Task Assignment (A)**.

6. Select the type of adapter you want, for example, Process Task (T). Then, click **OK**.

   See Also: *Oracle Identity Manager Administrative and User Console Guide* for more information about the Form Designer form

7. In the Description field, type a description for the adapter, for example, `This adapter is used to create a new user for the Solaris environment`.

8. From the toolbar, click **Save**.

   The adapter is now stored in the Oracle Identity Manager database.

3.3 Disabling and Re-enabling Adapters

To disable an adapter so that it cannot be used with a process task or form field, select the **Disable Adapter** option, and then save the adapter.

To re-enable it, clear the **Disable Adapter** option, and save the adapter.

3.4 About Adapter Variables

For a newly-created adapter to work, you can map data to the parameters of the adapter tasks. For this reason, you create placeholders, also known as adapter variables, to map the data at run time.

Note: An adapter variable can be reused for all adapter tasks.

Once an adapter variable is not needed for the adapter to run, you can remove it from the adapter. After you have deleted the adapter variable, ensure to recompile the adapter.

3.4.1 Creating an Adapter Variable

To create an adapter variable:

1. Select the adapter to which you wish to add an adapter variable, for example, the `Create Solaris User` adapter.
2. Select the Variable List tab.
3. Click **Add**.

   The Add a Variable window is displayed.
4. When you do not want Oracle Identity Manager to be able to change the adapter variable value after it is activated, select **Final**.
5. In the Variable Name field, enter the name of the adapter variable, for example, `SolarisUserID`.

   Caution: The adapter variable name cannot contain spaces.
6. From the Type menu, select the classification type of the adapter variable, such as String. The available items are:
   - Object
   - IT Resource
   - String
   - Boolean
   - Character
   - Byte
   - Date
   - Integer
   - Float
   - Long
   - Short
   - Double

7. Within the Description text area, you can enter explanatory information about the adapter variable.
8. From the Map To menu, you can map your adapter variable to one of the items listed in Table 3–1.

Table 3–1 Items on the Map To Menu

Name	Description
Literal	This adapter variable is mapped to a constant (or literal).
Resolve at Run time	This adapter variable's mapping occurs later, at run time. Selecting this option increases the reusability of the adapter.
Adapter References	This adapter variable gives access to an Oracle Identity Manager database reference or an Oracle Identity Manager data object reference.
System Date	When this adapter variable is triggered by Oracle Identity Manager, it is mapped to the current date and time of the Server. **Note**: This option appears only when you select the Date type.

Note: When you select the object type, a Qualifier menu is displayed within the Add a Variable window. From this menu, you can select either of the following:

- Database Reference. If you select this item, then the adapter variable is mapped to the reference of the database that the Oracle Identity Manager is currently running against.
- Data Object Reference. If you select this item, then the adapter variable is mapped to an Oracle Identity Manager data object.

Note: If you select the IT Resource type, then a Resource Type menu is displayed within the Add a Variable window. From this menu, you can select one of the IT resource types that have been created by using the IT Resource Type Definition form. By doing so, you can map the adapter variable to a parameter of this IT resource type.

9. On the toolbar in the Add a Variable window, click **Save**. The information for your adapter variable is stored in the Oracle Identity Manager database.

   Close the Add a Variable window to activate the main screen. The name, classification type, mapping selection, and description of the adapter variable you created appear in the child table of the Variable List tab.

This adapter variable now belongs to the adapter in the Adapter Factory form. It is saved to the Oracle Identity Manager database, and the adapter variable is ready to use.

3.4.2 Modifying an Adapter Variable

To modify an adapter variable:

1. Select the adapter that contains the adapter variable you want to edit, for example, the `Create Solaris User` adapter.

2. Click the Variable List tab and double-click the row header of the adapter variable you want to modify. The Edit a Variable window is displayed, showing information about the adapter variable.
3. Make the necessary edits, for example, changing the adapter variable's data type from String to Character.
4. On the Edit a Variable toolbar, click **Save**. The modified information about the adapter variable is stored in the Oracle Identity Manager database.
5. Close the Edit a Variable window to activate the main screen. The adapter variable you modified appears within the child table of the Adapter Factory form.

Note: Ensure that you check your data mappings and recompile the adapter, especially if you change the adapter variable's data type.

3.4.3 Deleting an Adapter Variable

When an adapter variable is no longer necessary for the adapter to run, you can remove it from the adapter. To do this:

1. Select the adapter that contains an adapter variable you want to remove, for example, the `Create Solaris User` adapter.
2. Select the Variable List tab.
3. From the list of this tab, select the adapter variable you want to delete.
4. Click **Delete**.
5. Recompile the adapter after deleting any variable.

The adapter variable disappears from the child table. The adapter variable has been deleted.

3.5 Creating Adapter Tasks

After you construct the adapter and create its variables, you can create the atomic function calls of an adapter. These function calls are known as adapter tasks.

Oracle Identity Manager allows you to create the following adapter tasks:

- A Java task, which allows an adapter to communicate with an external source by invoking Java API.
- A remote task, which enables an adapter to call a method on an API. This API resides on a computer that is external to Oracle Identity Manager.

  This type of task is used mostly with integrations of third-party APIs that are not network-enabled. A remote manager executes the remote API method, which is located on a remote computer. In addition, if the third-party API does not use Secure Sockets Layer (SSL), then you can use the remote manager to invoke third-party APIs over SSL-protected communication. Remote tasks can also be used with integrations of third-party APIs, which are network-enabled, but are not located on the Oracle Identity Manager server for scalability purposes. The remote API method is still executed by a remote manager. However, because the third-party API is network-enabled, the remote manager does not have to reside on the target system.
- A stored procedure task, which allows Oracle Identity Manager to map to and execute SQL programs located within a particular database schema. These

programs are known as stored procedures. They contain information, such as SQL statements, which are pre-compiled for greater efficiency.

By incorporating a stored procedure task into an adapter, and attaching this adapter to a process task, Oracle Identity Manager can incorporate stored procedures on any Oracle Database or Microsoft SQL Server database that is accessible on its network. This includes retrieving primitive values from stored procedures.

- A utility task, which enables you to populate an adapter with methods and APIs that come packaged with Oracle Identity Manager. In addition, this type of task provides you with access to the Java Standard Library APIs.
- An Oracle Identity Manager API task, which enables access to Oracle Identity Manager published APIs from adapter tasks. This allows for enhanced portability of adapter code.
- A set variable task, which allows you to set a variable within an adapter.
- An error handler task, which lets you display any errors associated with an adapter that occur at run time. In addition, you can see the reasons for the errors, along with possible solutions.
- A logic task, which lets you build a conditional statement within an adapter.

You can create the following types of logic tasks:

- FOR loops
- WHILE loops
- IF statements
- ELSE statements
- ELSE IF statements
- BREAK statements
- RETURN statements
- CONTINUE statements
- SET VARIABLE statements
- Handle Error statements

See Also: "Creating a Logic Task" on page 3-20 for more information about the types of logic tasks you can build

For classification purposes, Oracle Identity Manager represents each type of adapter task by an icon. The icon, which precedes the task name, is a visual indicator of the type of task it is. For example, "J" represents a Java task, and "LT" represents a logic task.

To see a list of these icons, select the Adapter Tasks tab, and click **Legend**. The Legend window appears, displaying the following list of icons:

- Functional Task
  - Java
  - Remote
  - Stored Procedure
- Utility Task

  - Utility
  - Oracle Identity Manager API
- Logical Task

3.5.1 Creating a Java Task

Oracle Identity Manager can handshake with an external source through a Java API. To make this happen, you must add a task to an adapter, which, when triggered by Oracle Identity Manager, initiates communications with the external source. This type of task is called a Java task.

To create a Java task:

1. Select the adapter to which you want to add a Java task, for example, the Update Solaris Password adapter.
2. Select the Adapter Tasks tab.
3. Click **Add**.

   After the Adapter Task Selection window is displayed, select the **Functional Task** option.
4. From the display area to the right of this option, select the Java item, and click **Continue**.

   The Object Instance Selection window is displayed.

   Table 3–2 explains the options in the Object Instance Selection window.

Table 3–2 Options in the Object Instance Selection Window

Option	Description
New Object Instance	When you click this option, you are creating a new Java object instance.
Persistent Instance	You can call the method on a persistent object by clicking this option, clicking the adjacent combo box, and selecting an object instance from the drop-down menu.
Task Return Value Instance	You can call this method on an object returned by an adapter task defined earlier by clicking this option, clicking the combo box, and selecting an adapter task from the drop down list.

Note: When the Persistent Instance option is grayed out, you have not defined any persistent objects for your adapter. Similarly, if the Task Return Value Instance option is grayed out, none of the tasks have Java Object return values associated with them.

5. Click an option—for example, New Object Instance—and click **Continue**. The Add an Adapter Factory Task window is displayed.

   Table 3–3 lists and describes the various regions of the Add an Adapter Factory Task window:

Table 3–3 Regions of the Add an Adapter Factory Task Window

Name	Description
Task Name	This field displays the name of the Java task.

Table 3–3 (Cont.) Regions of the Add an Adapter Factory Task Window

Name	Description
Persistent Instance	If this Java object is to be used again, the check box is selected, and the name of the task instance is entered in the adjacent field.
API Source	This combo box contains a list of all JAR and class files to which you have access.
Application API	This combo box contains a list of all class files to which you have access, and which belong to the JAR file that has been selected from the API Source list.
Constructors	This text area displays all the constructors, which are available for the Java object.
Methods	This text area shows a list of all the methods, which are available for the Java object.
Application Method Parameters	This area contains the parameters of the selected constructor and method. These parameters are mapped to the adapter variables and Oracle Identity Manager components.

6. In the Task Name field, enter the name of the task you are creating, for example, `Update Password`.
7. (Optional.) To make your Java object reusable, select **Persistent Instance**, then type the name of the instance of this task in the text field located to the right of the check box.

Caution: Ensure that name of the instance contains no spaces.

Note: To reference a session with the target resource multiple times during the life of the adapter, and not just once, select **Persistent Instance**.

Tip: By setting the Java object to be persistent, the next time you create a Java object, it appears in the Persistent Instance list of the Object Instance Selection window. In addition, you do not have to map the constructor to all adapter tasks of the same Java object.

8. Select the API Source. The JAR files appear, which Oracle Identity Manager references from the JavaTasks subdirectory of the *`OIM_HOME`*`/xellerate` folder path—for example, `C:\oracle\Xellerate\JavaTasks`.

See Also: "Configuring Oracle Identity Manager to Reference JAR and Class Files" on page 2-2 for instructions on how to enable Oracle Identity Manager to use third-party JAR files with a Java task

9. Select the Application API. The class files, which belong to the JAR file you selected in the API Source, appear.
10. From the Constructors area, select the method to be used to initialize the Java class you selected.
11. From the Methods area, select the method that will be used with your Java task.
12. From the toolbar, click **Save**.

The information pertaining to the Java task is stored in the Oracle Identity Manager database. You can now access the parameters of your Java task's constructors and methods. These parameters appear in the Application Method Parameters region of the Add an Adapter Factory Task window.

13. To display the Java class constructors and methods for which you must set mappings, click the plus icons displayed to the left of the Constructor and Method icons.
14. Select the parameter of the constructor or method for which you must set a mapping.
15. In the Description text area, you can enter a description for this mapping.
16. Click the Map to combo box, and select an item that you can map to the parameter of the constructor or method, for example, Adapter Variables.
17. Set the appropriate mappings.

    See Also: The "Adapter Mapping Information" section on *Oracle Identity Manager Reference* for more information about which mappings to set

18. Click **Set**.

    The parameter of the selected constructor or method now appears in blue. This signifies that it has been mapped.

    Tip: To remove a parameter mapping, right-click the appropriate parameter, and select Un-Map Parameter from the popup menu that appears.

19. Repeat steps 15 through 18 for all parameters of the constructors and methods that appear in the Application Method Parameters region.
20. On the Add an Adapter Factory Task window toolbar, click **Save**. The information pertaining to the Java task is stored in the Oracle Identity Manager database.
21. On the toolbar, click **Close**. The Add an Adapter Factory Task window disappears, and the main screen is active once again. The Java task that you created—for example, Update Password—appears within the Adapter Factory form.
22. (Optional.) To create additional Java tasks for the adapter, repeat steps 3-21.

    Tip: You can create different types of adapter tasks, and add them to the adapter.

    If the adapter is logically complete, and all variables on the adapter tasks are mapped, you can compile it to use with a process task or form field.

23. To compile the adapter, click **Build**.

    The text in the Compile Status field changes from Recompile to OK. This indicates that Oracle Identity Manager compiled the adapter and found no errors. You can now attach the adapter to a process task or form field.

24. (Optional.) To see the code that Oracle Identity Manager generates, from the toolbar, click **Notes**.

    The Notes window is displayed, containing the code that Oracle Identity Manager generated.

Note: If, after clicking Build, CODE GEN ERROR appears in the Compile Status field, it means that Oracle Identity Manager encountered one of two types of errors while validating and compiling the adapter:

- Validation Error

  While Oracle Identity Manager is checking the adapter to verify that it is valid, an error is found. This error can result from a parameter of an adapter task not being mapped, a parameter being mapped improperly, or an adapter task being placed out of order.

  Because Oracle Identity Manager generates code for an adapter only after it is validated, if Oracle Identity Manager encounters a validation error, it does not create any code.

- Java Compilation Error

  Oracle Identity Manager has verified that the adapter is valid. However, while Oracle Identity Manager is compiling the adapter, an error is found. This error can result from assigning an incorrect data type to an adapter task parameter.

  Because Oracle Identity Manager has validated the adapter, it generates code. However, Oracle Identity Manager stops building code at the point of the compilation where it encounters the error.

Tip: Once you create a Java task, and add it to an adapter, you can see the following information by accessing the Resources tab of the Adapter Factory form:

- The JAR and class files used to create the Java task.
- The name, which represents the directory path that contains these JAR and class files.

3.5.2 To Create a Remote Task

A remote task enables an adapter to invoke an API method by using the Remote Manager. This API resides on a computer that is external to Oracle Identity Manager. This section explains how to create a remote task.

Note: Before creating a remote task, ensure that you define an adapter variable with a classification type of IT Resource, as well as select one of the IT resources that have been created by using the IT Resource Type Definition form.

1. Select the adapter to which you wish to add a remote task.
2. Click the Adapter Tasks tab.
3. Click **Add**.

   The Adapter Task Selection window is displayed.
4. Select the Functional Task option.
5. From the display area to the right of the button, select the Remote item to create a remote task. Then, click **Continue**.

   The Object Instance Selection window is displayed.

Note: To learn more about the choices of this window, refer to "Creating a Java Task" on page 3-7.

6. Click **Continue**.

   The Add an Adapter Factory Task window is displayed.

7. In the Task Name field, enter the name of the remote task you are creating.

8. (Optional.) If you want your remote task to be reusable, select the **Persistent Instance** option. Then, type the name of the instance of this task in the text field, located to the right of the check box.

   Caution: Ensure that the name of the instance contains no spaces.

   See Also:

   "Creating a Java Task" on page 3-7 for more information about the regions of the Add an Adapter Factory Task window

   "Configuring Oracle Identity Manager to Reference JAR and Class Files" on page 2-2 for information about how to enable Oracle Identity Manager to use third-party JAR files with a Java task

9. From the Add an Adapter Factory Task window, select a JAR file, class file, constructor, and method. Then, set the mappings for the parameters of the constructor and method.

   Note: One of the input parameters will have a classification type of IT Resource. You must associate this parameter with an adapter variable of type IT Resource.

   See Also: The "Adapter Mapping Information" section in *Oracle Identity Manager Reference* for more information about which mappings to select

10. From the Add an Adapter Factory Task window toolbar, click **Save**.

    The information pertaining to the remote task is stored in the Oracle Identity Manager database.

11. From this window toolbar, click **Close**.

    The Add an Adapter Factory Task window disappears, and the main screen is active once again. The remote task that you created appears within the Adapter Factory form.

12. (Optional.) To create additional remote tasks for the adapter, repeat Steps 3 through 11.

    You are now ready to compile the adapter, so it can be used with a process task or form field.

13. To compile the adapter, click **Build**.

    The text in the Compile Status field changes from Recompile to OK. This indicates that Oracle Identity Manager compiled the adapter and did not find any errors.

You can now attach the adapter to a process task or form field, so Oracle Identity Manager can communicate with the external API.

3.5.3 To Create a Stored Procedure Task

Through Oracle Identity Manager, you can map to and execute SQL programs that are located within a particular database schema. These SQL programs are known as stored procedures. Stored procedures contain information, such as SQL statements, which are precompiled for greater efficiency.

For this to occur, you must add a stored procedure task to an adapter. When triggered by Oracle Identity Manager, this task incorporates stored procedures on any Oracle Database or Microsoft SQL Server database that is accessible on its network. This includes retrieving primitive values from stored procedures.

To create a stored procedure task:

Note: The parameter values and server type for the database schema are set within the IT Resources form.

The server type of the schema must be set to Database. Otherwise, Oracle Identity Manager cannot reference the database schema during the creation of a stored procedure task, the execution of a stored procedure task, or both.

In addition, Oracle Identity Manager uses values, which are represented by parameters—for example, Database Name or *URL*—to connect to the schema. As a result, the stored procedures contained within the schema, can be executed.

---

1. For Oracle Identity Manager Installations that use Oracle Database, copy the ojdbc14.jar file from the *OIM_HOME*/xellerate/ext/ directory to the *OIM_DC_HOME*/xlclient/ext directory.

   For Oracle Identity Manager Installations that use Microsoft SQL Server, you must obtain the following files from Microsoft and copy them to the *OIM_DC_HOME*/xlclient/ext directory:

   - msbase.jar
   - mssqlserver.jar
   - msutil.jar

2. Select the adapter to which you wish to add a stored procedure task, for example, the Update User ID adapter.
3. Click the Adapter Tasks tab.
4. Click **Add**.

   The Adapter Task Selection window is displayed.

5. Select the **Functional Task** option.
6. From the display area to the right of the option, select **Stored Procedure**, and click **Continue**. The Add an Adapter Factory Task window is displayed.

   The following table lists and describes the regions of the Add an Adapter Factory Task window.

Table 3–4 Regions of the Add an Adapter Factory Task Window

Name	Description
Task Name	Displays the name of the stored procedure task.
Description	Displays explanatory information about the stored procedure task.
Database	Lists the databases defined in the IT Resources form. Important: Only those IT resources with a server type of Database appear in the Database list.
Schema	Lists the schemas, which are associated with the database that appears in the Database list.
Procedure	Lists the stored procedures, which reside within the database schema that is displayed in the Schema list.

Connection Status	Displays the status of the connection between Oracle Identity Manager and the database that contains the target stored procedure.
	When Oracle Identity Manager can connect to the database, Connection Established is displayed in the Connection Status region.
	Note: If Oracle Identity Manager cannot connect, Connection Failed appears in the display area. In addition, the **Notes** button of the Add an Adapter Factory Task window is enabled. Clicking this button shows you why a connection could not be established, for example:
	`Exception Type:`
	`java.lang.ClassNotFoundExceptionMessage:`
	`java.lang.ClassNotFoundException:`
	`oracle.jdbc.driver.OracleDriver`
	In this example, Oracle Identity Manager could not connect to the designated database because it could not find a particular Java class.

Parameters	Contains parameters that can be mapped to the stored procedure. These parameters appear after you select a database, schema, and stored procedure and save this information to the Oracle Identity Manager database.

7. In the Task Name field, enter the name of the stored procedure task you are creating (for example, Update ID).
8. In the Description text area, you can enter a description for this stored procedure task.
9. Click the Database list. The databases, which are defined in the IT Resources form, appear.

Note: If Oracle Identity Manager cannot connect to the database you selected, *Connection Failed* appears in the display area. In addition, the **Notes** button of the Add an Adapter Factory Task window is enabled. Clicking this button shows you why a connection could not be established.

Tip: Schemas and stored procedures appear only after you select a database to which Oracle Identity Manager can connect. Based on this selection, related schemas and stored procedures appear in the corresponding combo boxes.

See Also: *Oracle Identity Manager Administrative and User Console Guide* for more information about using the IT Resources form to define the settings for databases that contain stored procedures

10. Click the **Schema list**. The schemas appear, which are associated with the database you selected.
11. Click the **Procedure list**. The stored procedures, which reside within the database schema that you selected from the Schema combo box, appear.
12. From the Add an Adapter Factory Task window's toolbar, click **Save**. The information pertaining to the stored procedure task is saved into the Oracle Identity Manager database.

    You can now set the mappings for the parameter(s) of the stored procedure. These parameters appear in the Parameters region of the Add an Adapter Factory Task window.

    Note: Oracle Identity Manager automatically maps the database and schema of the selected stored procedure. However, Oracle Identity Manager enables you to override these mappings.

    See Also: The "Adapter Mapping Information" section in *Oracle Identity Manager Reference* for more information about which mappings to select

13. From the Add an Adapter Factory Task window's toolbar, click **Save**. The mappings that you have set for the parameter(s) of the stored procedure task are stored in the Oracle Identity Manager database.
14. From this window's toolbar, click **Close**.

    The Add an Adapter Factory Task window disappears, and the main screen is active once again. The stored procedure task you created (for example, Update ID) appears within the Adapter Factory form.
15. (Optional.) Repeat steps 3 through 13 to create additional stored procedure tasks for the adapter.
16. To compile the adapter, click **Build**.

    The text in the Compile Status field changes from Recompile to OK. This indicates that Oracle Identity Manager compiled the adapter and did not find any errors. You can now attach the adapter to a process task or form field, so Oracle Identity Manager can map to and execute the stored procedure you selected.

3.5.4 To Create a Utility Task

The Adapter Factory is shipped with a library of utility classes and methods, which increase the efficiency of developing adapters.

These utility classes and methods are contained within the **xlUtils.jar**, **xlIntegration.jar**, and **rt.jar** files. A Java task you create by using a class or method from one of these JAR files is called a **utility task**.

> **See Also:** *Java API Documentation* for more information about the class files that contains the *xlUtils.jar*, *xlAPI.jar*, *xlIntegration.jar*, and `rt.jar` files.

1. Select the adapter to which you wish to add a utility task, for example, the Update Solaris User Group adapter.
2. Click the Adapter Tasks tab.
3. Click **Add**.

   The Adapter Task Selection window is displayed.
4. Select the **Utility Task** option.
5. From the display area to the right of the option, select **Utility**, and click **Continue**. The Object Instance Selection window is displayed.

   > **See Also:** "Creating a Java Task" on page 3-7 to learn more about the choices of this window
6. Click **Continue**. The Add an Adapter Factory Task window is displayed
7. In the **Task Name** field, enter the name of the utility task you are creating, for example, Update User Group.
8. (Optional.) If you want your utility task to be reusable, select **Persistent Instance**, then type the name of the instance of this task in the text field to the right of the check box.

   Caution: Ensure that name of the instance does not contain any spaces.

   See Also:

   "Creating a Java Task" on page 3-7 for more information about the regions of the Add an Adapter Factory Task window

   "Configuring Oracle Identity Manager to Reference JAR and Class Files" on page 2-2
9. Click the Application API list. The class files appear, which belong to the `xlUtils.jar`, `xlIntegration.jar`, and `rt.jar` files.

   Note: The xlUtils.jar, xlIntegration.jar, and rt.jar files contain all of the class files that you can use for a utility task. Therefore, you do not have to access the API Source list.

   ---
10. From the Add an Adapter Factory Task window, select a constructor and method. Then, set the mappings for the parameters of the constructor and method.

> **See Also:** The "Adapter Mapping Information" section in *Oracle Identity Manager Reference* for more information about which mappings to select

11. From the Add an Adapter Factory Task window's toolbar, click **Save**. The information pertaining to the utility task is stored in the Oracle Identity Manager database.
12. From this window's toolbar, click **Close**.

    The Add an Adapter Factory Task window disappears, and the main screen is active once again. The utility task that you created (for example, Update User Group) appears within the Adapter Factory form.
13. (Optional.) Repeat steps 3 through 12 to create additional utility tasks for the adapter.

    You are now ready to compile the adapter, so it can be used with a process task or form field.
14. To compile the adapter, click **Build**.

    The text in the Compile Status field changes from Recompile to OK. This indicates that Oracle Identity Manager compiled the adapter and did not find any errors. You can now attach the adapter to a process task or form field.

3.5.5 To Create an Oracle Identity Manager API Task

For greater portability of the adapter code, an Oracle Identity Manager API task enables Adapter tasks to call APIs published by Oracle Identity Manager. This is better than accessing Oracle Identity Manager data directly through hardcoded SQL statements.

The Adapter Factory is shipped with a library of utility classes and methods, which increase the efficiency of developing adapters that contain Oracle Identity Manager API tasks. These utility classes and methods are contained within the xlAPI.jar file.

> **See Also:** Java API Documentation for more information about the class files that contain the `xlUtils.jar`, `xlAPI.jar`, `xlIntegration.jar`, and `rt.jar` files

To create this type of adapter task:

1. Select the adapter to which you wish to add an Oracle Identity Manager API task, for example, the Get User's Password adapter.
2. Click the Adapter Tasks tab.
3. Click **Add**.

   The Adapter Task Selection window is displayed.
4. Select the **Utility Task** option.
5. From the display area to the right of the option, select Xellerate API, and click **Continue**. The Object Instance Selection window is displayed.

   > **See Also:** "Creating a Java Task" on page 3-7 to learn more about this window
6. Click **Continue**. The Add an Adapter Factory Task window is displayed.

7. In the Task Name field, enter the name of the Oracle Identity Manager API task you are creating, for example, Retrieve Password).
8. (Optional.) If you want your Oracle Identity Manager API task to be reusable, select Persistent Instance. Then, type the name of the instance of this task in the text field to the right of the check box.

   Tip: Ensure that name of the instance contains no spaces.

   See Also:

   "Creating a Java Task" on page 3-7 for more information about the regions of the Add an Adapter Factory Task window

   "Configuring Oracle Identity Manager to Reference JAR and Class Files" on page 2-2 to learn how to enable Oracle Identity Manager to use third-party JAR files with a Java task

9. Click the Application API list. The class files appear, which belong to the `xlAPI.jar` file.

   Note: `The xlAPI.jar` file contains all of the class files that you can use for an Oracle Identity Manager API task. Therefore, you do not have to access the API Source list.

10. From the Add an Adapter Factory Task window, select a class file, constructor, and method. Then, set the mappings for the parameters of the constructor and method.

    See Also: The "Adapter Mapping Information" section in *Oracle Identity Manager Reference* for more information about which mappings to select

11. From the Add an Adapter Factory Task window's toolbar, click **Save**. The information pertaining to the Oracle Identity Manager API task is stored in the Oracle Identity Manager database.
12. Close the Add an Adapter Factory window to activate the main screen. The Oracle Identity Manager API task that you created—for example, *Retrieve Password*—appears within the Adapter Factory form.
13. (Optional.) To create additional Oracle Identity Manager API tasks for the adapter, repeat steps 3 through 12.

    You are now ready to compile the adapter, so it can be used with a process task or form field.
14. To compile the adapter, click **Build**.

    The text in the **Compile Status** field changes from Recompile to OK. This indicates that Oracle Identity Manager compiled the adapter and did not find any errors. You can now attach the adapter to a process task or form field, so Oracle Identity Manager can communicate with a third-party application.

3.5.6 Reassigning the Value of an Adapter Variable

Sometimes, for an adapter to accomplish its required objective, you must reassign the value of one adapter variable to another adapter variable, a different type of adapter

task, or a constant (or literal). The task that enables you to reallocate an adapter variable value is known as a set variable task.

> **See Also:** "About Adapter Variables" on page 3-3

For example, you can create a set variable task to set the adapter variable return value to equal the output of an adapter task (UserName) if the User ID length is fewer than 11 characters.

To create a set variable task:

1. Select the adapter to which you wish to add a set variable task (for example, the Check the Solaris User ID adapter).
2. Click the Adapter Tasks tab.
3. Click **Add**. The Adapter Task Selection window is displayed.
4. Select the Logic Task option.
5. From the display area, select SET VARIABLE, and click **Continue**. The Add Set Variable Task Parameters window is displayed.
6. From the Variable Name list, select the adapter variable that has a value you want to reassign—for example, Adapter return value.
7. From the Operand Type list, select the type of operand that will provide the value for the variable.

   > **Tip:** You can reassign an adapter variable's value to another adapter variable, a different type of adapter task, or a literal.

   Use Table 3–5 to understand the various types of operands.

Table 3–5 Types of Operands

Operand Name	Description
Variable	If you select this operand type, then adapter variables appear in the Operand Qualifier list. From this list, select the specific adapter variable that will provide the reassigned value. **Note**: The only adapter variables that will appear in the Operand Qualifier combo box will be those variables that have the same data type as the adapter variable that is displayed within the Variable Name combo box.
Adapter Task	By selecting this operand type, adapter tasks are displayed in the Operand Qualifier combo box. From this combo box, select the particular adapter task that will provide the reallocated value. **Note**: The only adapter tasks that will appear in the Operand Qualifier combo box will be those tasks that have the same data type as the adapter variable that is displayed within the Variable Name combo box.
Literal	When you select this operand type, types of literals appear in the Operand Qualifier combo box. From this combo box, select the type of literal that will provide the reallocated value. Then, type the specific literal into the field that appears underneath the combo box.

The following task sets the adapter variable's return value to be equal to the UserName adapter variable.

1. On the toolbar in the Add Set Variable Task Parameters window, click **Save**. The set variable task you created is stored in the Oracle Identity Manager database.
2. On the Add Set Variable Task Parameters window toolbar, click **Close**. The Add Set Variable Task Parameters window disappears, and the main screen is active once again. The set variable task that you created, for example, `Set Adapter return value = UserName`, appears in the Adapter Factory form.
3. (Optional.) Repeat Steps 3-9 to create additional set variable tasks for the adapter.

   You are now ready to compile the adapter, so it can be used with a process task or form field.
4. To compile the adapter, click **Build**. The text in the Compile Status field changes from Recompile to OK. Oracle Identity Manager compiled the adapter and found no errors. You can attach the adapter to a process task or form field.

3.5.7 Adding an Error Handler Task

To add an error handler task:

1. An adapter task can return errors. When this occurs, the process task or form field to which the adapter is attached gets rejected.

   You can attach your own customizable error messages, which will be displayed to the user. These messages are known as error handler tasks.

   For example, you can attach an error handler task to an adapter that will display an error message when the length of a User ID is greater than 10 characters.
2. Select the adapter to which you wish to add an error handler task (for example, the *Check the Solaris User ID* adapter).
3. Click the Adapter Tasks tab.
4. Click **Add**.

   The Adapter Task Selection window is displayed.
5. Select the **Logic Task** option.
6. From the display area, select Handle Error, and click **Continue**. The Add an Adapter Factory Task window is displayed.
7. Double-click this window's lookup field. The Lookup window is displayed, displaying the error handler tasks you can add to the adapter.

   Note: The only error handler tasks that appear in this Lookup window are the ones that begin with ADAPTER—such as ADAPTER.USERIDLENERR).

   If you do not see the error handler task that you want to incorporate into the adapter, you can create one by accessing the Error Message Definition form. Refer to *Oracle Identity Manager Design Console Guide* for information about the Error Message Definition form.
8. Select the error handler task you want, for example, ADAPTER.USERIDLENERR.
9. Click **OK**. The Lookup window disappears, and the Add an Adapter Factory Task window is active. In addition, the error handler task you selected appears in the field of this window.

10. From the Add an Adapter Factory Task window toolbar, click **Save**. The error handler task you incorporated into the adapter is stored in the Oracle Identity Manager database.
11. From this window's toolbar, click **Close**.

    The Add an Adapter Factory Task window disappears, and the main screen is active once again. The error handler task you added, for example, `Handle Error.ADAPTER.USERIDLENERR`, appears within the child table of the Adapter Factory form.
12. (Optional.) Repeat Steps 3-10 to create additional error handler tasks for the adapter.

    If the adapter is logically complete and all variables on the adapter tasks are mapped, you can compile it to use with a process task or form field.
13. To compile the adapter, click **Build**.

    The text in the Compile Status field changes from *Recompile* to *OK*. This indicates that Oracle Identity Manager compiled the adapter and did not find any errors. You can now attach the adapter to a process task or form field.

See Also:

- *Oracle Identity Manager Design Console Guide* for information about creating error messages by using the Error Message Definition form
- *Oracle Identity Manager Globalization Guide* for information about localizing user-defined data by using custom resource bundles

3.5.8 Creating a Logic Task

While defining the adapter, you can add conditional statements to the adapter to control its logic flow. These conditional statements are known as logic tasks. For example, you can create a logic task that will trigger an action if the length of a User ID is greater than 10 characters.

To create a logic task:

1. Select the adapter to which you wish to add a logic task (for example, the Check the Solaris User ID adapter).
2. Click the Adapter Tasks tab.
3. Click **Add**. The Adapter Task Selection window is displayed.
4. Select the **Logic Task** option.
5. From the display area, select the type of logic task you want to create. Then, click **Continue**.

Note: If you select a conditional expression, and click **Continue**, one of the following actions occurs:

Oracle Identity Manager adds the conditional statement to the adapter directly; or

A secondary window is displayed, containing fields about the conditional expression that you can configure.

To see what happens when you select a particular conditional statement, refer to Table 3–6.

Table 3–6 Actions Resulting from Particular Conditional Statements

Conditional Statement	Statement Is Added to the Adapter Directly	Secondary Window Appears
FOR		X
WHILE		X
IF		X
ELSE	X	
ELSE IF		X
BREAK	X	
RETURN	X	
CONTINUE	X	

Table 3–7 explains the various regions of the Add Adapter Factory Logic Task Parameters window:

Table 3–7 Regions of the Add Adapter Factory Logic Task Window

Name	Description
Operand Type	These combo boxes contain types of operands, such as adapter tasks and adapter variables.
Comparator Combo Box	From this combo box, you can set the relationship between two operands (for example, <, =, >).
Operand Qualifier	These combo boxes contain the qualifiers for the operands.
Literal Text Box	When you select the Literal operand type, enter the specific literal into this field.

Note: By selecting the FOR conditional statement, an Add Adapter Factory Logic Task Parameters window is displayed. However, it contains different text and combo boxes.

For the FOR conditional expression, use Table 3–8 to understand the various regions of this Add Adapter Factory Logic Task Parameters window.

Table 3–8 Add Adapter Factory Logic Task Parameters for FOR Conditional Statement

Name	Description
Operand Type	These combo boxes contain types of operands, such as adapter tasks and adapter variables.
Comparator Combo Box	From this combo box, you can set the relationship between two operands (for example, <, =, >).
Operand Qualifier	These combo boxes contain the qualifiers for the operands.
Increment Combo Box	Within this area, you can set whether the initial value will increase or decrease, and by how much.

Note: If you select the ELSE, BREAK, RETURN, or CONTINUE conditional expressions, proceed to Step 8.

6. Set the parameters for your conditional expression.

   This logic task will check to see if the length of the User ID is greater than 10 characters.

7. From the Add Adapter Factory Logic Task Parameters window toolbar, click **Save**.

   The logic task you created is stored in the Oracle Identity Manager database.

8. From this window toolbar, click **Close**. The Add Adapter Factory Logic Task Parameters window disappears, and the main screen is active once again. The logic task that you created—for example, If (Check ID Length > 10)—appears within the Adapter Factory form.

9. (Optional.) Repeat Steps 3-8 to create additional logic tasks for the adapter.

   Caution: All adapter tasks that can be executed for a condition of a logic task should be nested properly under that logic task.

   See Also: "Changing the Order and Nesting of Tasks" on page 3-23 for more information about nesting tasks

   You are now ready to compile the adapter, so it can be used with a process task or form field.

10. To compile the adapter, click **Build**.

    The text in the Compile Status field changes from *Recompile* to *OK*. This indicates that Oracle Identity Manager compiled the adapter and did not find any errors. You can now attach the adapter to a process task or form field.

3.6 Modifying Adapter Tasks

The following procedure will show you how to edit an adapter task, in case you must make changes to it. To modify an adapter task

1. Select the adapter that contains the adapter task you wish to edit (for example, the *Update Solaris User Group* adapter).
2. Click the **Adapter Tasks** tab.
3. Double-click the adapter task that you want to modify.

   The Edit Adapter Factory Task Parameters window is displayed, displaying information that relates to the adapter task you selected. Within this window, make the necessary modifications.

4. On the Edit Adapter Factory Task Parameters window toolbar, click **Save**.

   The information you modified is stored in the Oracle Identity Manager database.

5. On the toolbar, click **Close**.

   The Edit Adapter Factory Task Parameters window disappears. The main screen is active again. The modified task appears within the child table of the **Adapter**

Factory form. You must re-compile the adapter, so it can be used with a process task or form field.

6. To recompile the adapter, click **Build**.

   The text in the **Compile Status** field changes from *Recompile* to *OK*. This indicates that Oracle Identity Manager compiled the adapter and did not find any errors. You can now attach the adapter to a process task or form field.

   Caution: You cannot modify the API call inside a Java, Xellerate API, or Utility task. The adapter task has to be deleted and re-created.

   In addition, if *CODE GEN ERROR* appears in the Compile Status field, Oracle Identity Manager encountered errors while compiling the adapter. Rectify the errors, if necessary re-do the adapter task modifications, and compile the adapter again.

3.7 Changing the Order and Nesting of Tasks

If you add multiple tasks to an adapter, you can either change the order in which the tasks are executed, or place one task inside of another task for the adapter to work.

The following procedure will show you how to change the order and nesting of tasks.

Caution: You should not change the order and nesting of adapter tasks unless you understand the mapping dependencies of the adapter tasks.

To change the order and nesting of tasks:

1. Select the adapter that contains tasks of which you want to change the order and/or nest (for example, the *Check the Solaris User ID* adapter).
2. Click the **Adapter Tasks** tab.

   The tasks appear, which belong to the current adapter.

   In this example, the following changes must occur:

   - The error handler task must be nested inside of the *IF (Check ID Length > 10)* logic task.
   - The set variable task has to be nested inside of the **ELSE** logic task.
   - The **IF** logic task precedes the **ELSE** logic task.

   Therefore, you must first reorganize the logic tasks. Then, you must nest the error handler task and set variable task inside of the **IF** and **ELSE** logic tasks, respectively. To reorganize tasks:

3. Select the task that must run before another task, and click the **Up** arrow button. The selected task will switch places with the task that precedes it.

   or

   Select the task that must be executed after another task, and click the **Down** arrow button. The highlighted task is displayed below the task that previously followed it.

   To nest tasks/remove task nestings:

4. Select the task that must be placed inside of another task, and click the **Right** arrow button. The selected task will be nested inside of the task that appears above it.

   or

   Select the task that no longer be nested inside of another task, and click the **Left** arrow button. The highlighted task will not be nested inside of the task that is displayed above it.

5. On the toolbar, click **Save**.

   The order and nesting of the adapter's tasks is stored in the Oracle Identity Manager database. If the adapter is logically complete and all variables on the adapter tasks are mapped, you can compile it to use with a process task or form field.

6. To compile the adapter, click **Build**.

   The text in the Compile Status field changes from *Recompile* to *OK*. This indicates that Oracle Identity Manager compiled the adapter and did not find any errors. You can now attach the adapter to a process task or form field.

Caution: If you see *CODE GEN ERROR* in the Compile Status field, Oracle Identity Manager found errors while compiling the adapter. Rectify the errors, if necessary re-do the adapter task modifications, and compile the adapter again.

3.8 Deleting Adapter Tasks

When an adapter task is no longer necessary for the adapter to run, you must remove it from the adapter. To delete an adapter task:

1. Select the adapter that contains the task you wish to remove (for example, the *Update Solaris User Group* adapter).
2. Click the **Adapter Tasks** tab.
3. Select the task that you want to remove (for example, the *CONTINUE* logic task).
4. Click **Delete**.

   The selected task is deleted and disappears from the child table.

5. On the toolbar, click **Save**.
6. Recompile the adapter.

Caution: While deleting adapter tasks, ensure that the logic of the adapter is consistent and maintained.

3.9 Working with Responses

Adapters can have different outcomes, called **responses**. Based on these responses, adapters can trigger other process tasks.

For example, if the adapter returns a *True* response, the process task's status can be set automatically to *Completed*. However, if the adapter returns a *False* response, the process task's status can be set automatically to *Rejected*, and another process task can be triggered.

These responses can be added, modified, or removed on the Responses tab of the Adapter Factory form.

The following procedures will show you how to create, modify, and delete responses.

Note: Responses are used only with process task adapters, because these adapters are attached to process tasks. Rule generators, pre-populate adapters, and entity adapters are not connected to processes. In addition, task assignment adapters are not associated with responses. Therefore, if the active adapter is a task assignment adapter, rule generator, pre-populate adapter, or entity adapter, Oracle Identity Manager disables the Responses tab.

3.9.1 To Create a Response

1. Select the adapter to which you want to add responses (for example, the *Create Solaris User* adapter).
2. Click the **Responses** tab.
3. Click **Add**.

   An empty row is inserted into the **Responses** tab.
4. Click the field that appears within the **Code Name** column.
5. Enter a code, which represents a response type that can be generated (for example, *True*).
6. Click the field that appears within the **Description** column.
7. Enter a description for this response (for example, *The user was created successfully*.).
8. Double-click the field that appears within the **Status** column.

   The Lookup popup window is displayed, containing the different status levels that you can associate with the response.

   Note: For more information about Oracle Identity Manager's status levels, refer to Chapter 4, "About Process Task Adapters" on page 4-1.
9. Click the desired status level (for example, *Completed (C)*). Then, click **OK**.

   The Lookup window disappears, and the **Responses** tab is active once again.
10. Create another response, by clicking the **Add** button, and entering *False* and *The user was not created successfully.* into the Code Name and Description fields, respectively. Then, access the Lookup window, and assign the *Rejected (R)* status level to this response.
11. On the toolbar, click **Save**.

    The responses that you created for this adapter have been stored in the Oracle Identity Manager database. After you attach this adapter to a process task, these responses will appear in the Responses tab of the Editing Task window of the Process Definition form.

3.9.2 To Modify a Response

The following procedure demonstrates how to edit a response.

1. Select the adapter that contains the response you want to edit (for example, the *Create Solaris User* adapter).
2. Click the **Responses** tab.
3. Double-click the field of the response, which contains information that you want to modify.
   a. If the field is a text field, Oracle Identity Manager enables it. You can now edit the contents within this field.
   b. When the field is a lookup field, the Lookup popup window is displayed, containing the different status levels that you can associate with the response. Click the desired **status level**, then click **OK**.

   For example, double-click the **Status** column of the *False* response, select the *Suspended (S)* status level, and click **OK**.
4. On the toolbar, click **Save**.

   The information that you modified for the response is stored in the Oracle Identity Manager database.

3.9.3 To Delete a Response

When a response is no longer necessary, you can delete it from the adapter.

1. Select the adapter, which contains a response that you want to remove.
2. Click the **Responses** tab.
3. Select the response that you want to delete.
4. Click **Delete**.

The response disappears. This indicates that Oracle Identity Manager has deleted the response.

3.10 Scheduling Rule Generators and Entity Adapters

Oracle Identity Manager triggers a process task adapter or a task assignment adapter automatically if it is attached to a process task, and the process task's status is *Pending*. In addition, Oracle Identity Manager always triggers pre-populate adapters on pre-insert. Therefore, you do not schedule when process task adapters, task assignment adapters, or pre-populate adapters will be executed.

On the other hand, a rule generator and an entity adapter are attached to a form field. The only way that Oracle Identity Manager will be able to execute the rule generator or entity adapter is for you to specify when it will be triggered. You do this through the Execution Schedule tab.

> **Note:** If an entity adapter is attached to a process form or an object form for validation of field values, then these adapters will trigger if we edit data in these forms after completing direct or request provisioning.

Using this tab, you can determine that Oracle Identity Manager will trigger the rule generator or entity adapter on preinsert or preupdate. In addition, you can also schedule an entity adapter to be executed on predelete, postinsert, postupdate, and postdelete.

This procedure demonstrates how to configure Oracle Identity Manager to trigger a rule generator or entity adapter.

3.10.1 Scheduling Rule Generators and Entity Adapters

To schedule rule generator and entity adapters:

1. Select the rule generator or entity adapter that you want Oracle Identity Manager to trigger (for example, *Solaris User ID Generator*).

   Note: When you work with process task adapters or pre-populate adapters, you do not use the Execution Schedule tab. As a result, this tab, as well as its contents, are grayed out.

2. Click the **Execution Schedule** tab.

   The contents of the Execution Schedule tab appear.

   The following table will help you understand the various check boxes of the **Execution Schedule** tab:

Name	Description
Pre-Insert	By clicking this check box, Oracle Identity Manager can trigger the rule generator or entity adapter before the record is inserted into the database.
Pre-Update	When you click this check box, Oracle Identity Manager can trigger the rule generator or entity adapter before the record is updated in the database.
Pre-Delete	By clicking this check box, Oracle Identity Manager can trigger the entity adapter before the record is deleted from the database.
Post-Insert	When you click this check box, Oracle Identity Manager can trigger the entity adapter after the record is inserted into the database.
Post-Update	By clicking this check box, Oracle Identity Manager can trigger the entity adapter after the record is updated in the database.
Post-Delete	When you click this check box, Oracle Identity Manager can trigger the entity adapter after the record is deleted from the database.

   Note: By clicking the check boxes of the Execution Schedule tab, you are defining the times when Oracle Identity Manager *can* trigger the rule generator or entity adapter. The Data Object Manager form allows you to specify when Oracle Identity Manager *will* trigger the rule generator or entity adapter.

   For more information about the Data Object Manager form, refer to "Mapping Rule Generator Adapter Variables" on page 6-2.

3. Enable the desired check boxes. Then, from the toolbar, click **Save**.

   The criteria you set for Oracle Identity Manager to execute the rule generator or entity adapter is stored in the Oracle Identity Manager database.

4

About Process Task Adapters

This chapter describes the process task adapters. It contains the following topics:

4.1 Introduction to Processes and Process Tasks

A process task adapter is Java code created through the adapter factory. It enables the Oracle Identity Manager to automatically execute process tasks in provisioning processes.

Each process and process task has a status, which indicates the stage of its completion. The statuses for a process or process task are listed in the following table in the order of importance.

> **Note:** The UT, UCR, and XLR statuses are not mentioned in the following table. Do not use these status values because they have been deprecated and will be removed in a future release of Oracle Identity Manager.

Task Status	Description
C	Completed: This process/process task has been completed successfully.
MC	Manually Completed: This process task has been completed successfully by an Oracle Identity Manager user (that is, manually).
P	Pending: This process/process task is in the process of being completed. All preceding tasks and processes, respectively, have been completed.
PX	Pending Cancellation: This process task will be canceled, but this task has to be completed first before it can be canceled.
R	Rejected: This process/process task has not been completed successfully or has not been approved. The status of rejected process tasks can only be changed to *Canceled* or *Unsuccessfully Completed*.

Task Status	Description
S	Suspended: This process/process task has been put on hold temporarily.
UC	Unsuccessfully Completed: This process task has been set to *Completed*. However, it had been rejected before.
W	Waiting: This process/process task cannot be completed until all preceding process tasks or processes are completed.
X	This process/process task has been stopped. Its status cannot change anymore

The status level of a process represents the most important status level of its process tasks, which must be completed for the process to be completed. Suppose a process has three process tasks, each process task has a different status level (*Completed*, *Waiting*, and *Rejected*), and all three process tasks must be completed for the process to be completed. Because the highest task status level is *Rejected*, the status level of the process is also *Rejected*.

A process task can be managed in one of the following ways:

- It can be handled manually by using the Object Process Console tab of the Organizations or Users forms, or the Oracle Identity Manager Web Application.
- An Oracle Identity Manager process can be configured so that one (or more) of its tasks is triggered automatically once it achieves a status of *Pending*.

The Java code that enables Oracle Identity Manager to automatically execute a process task is referred to as a process task adapter. The Oracle Identity Manager tool that allows developers to create and manage process task adapters is known as the Adapter Factory.

Note: For more information about handling process tasks manually, refer to the *Oracle Identity Manager Administrative and User Console Guide*.

4.2 How a Process Task Adapter Works

After you create a process task adapter, you attach it to the appropriate process task by using the Integration tab of the Process Definition form. From this tab, you can also map any variables of the adapter to their proper locations, which were designated as either *Resolve at Run time* or as an adapter return variable.

For example, the adapter named *adpSOLARISPASSWORDUPDATED* is connected to the *Password Updated* task of the *Solaris* process.

After you attach an adapter to a process task, for the adapter to be functional, it might need data from fields of other forms. For this example, the *adpSOLARISPASSWORDUPDATED* adapter cannot work unless it obtains the following information:

- The user's Oracle Identity Manager ID and password.
- The user's Solaris ID and password.
- The IP address where Solaris is located.

Therefore, it must get this information from the *UserID*, *Passwd*, *SolarisUserID*, *SolarisUserPasswd*, and *ServerAddress adapter* variables respectively. These five variables

are created by using the Adapter Factory form. The "Y" that precedes each adapter variable signifies that it has been mapped correctly.

The form that enables you to create process-specific fields, which will be used by a process to obtain the information it needs, is called the Form Designer. When you create these fields, Oracle Identity Manager stores them into a table. Then, by associating this table with a process (through the Table Name lookup field of the Process Definition form), the adapter, which you attach to a task of this process, will use the table to retrieve the appropriate data.

If you want to modify this table, you can do so through the Form Designer form.

The *UD_SOLARIS* table contains two fields: *UD_SOLARIS_USERID* and *UD_SOLARIS_PASSWD*. By accessing this record of the Form Designer form, you can edit the fields of the table.

Note: For more information about using the Form Designer form to create and modify process-specific fields, refer to the *Oracle Identity Manager Administrative and User Console Guide*.

Once you attach the process task adapter to a dependent process task, and the status of this process task is *Pending* (the status of the previous process task is *Completed*), Oracle Identity Manager will trigger the adapter automatically. When the process task is an independent task, Oracle Identity Manager will execute the adapter as soon as the process is requested.

The result of the adapter being executed represents the state of the process task. When the adapter is finished successfully, the process task to which this adapter is attached will have a status of *Completed*.

On the other hand, if the adapter cannot perform its designated function, the process task to which this adapter is attached will have a status of *Rejected*. By discovering the cause of the error, you can modify the process task and/or adapter so it can run successfully.

Note: To determine why a process task might have failed:

Find the process task. When the process task has not yet been provisioned to the target user or organization, it is located in the To Do List or Pending Approvals. To find the task:

1. Log in as the user.
2. Select the To Do List link or the Pending Approvals links in the left side of the window.

4.3 Attaching Process Task Adapters to Process Tasks

In the previous chapter, you learned how to create a process task adapter. You must attach it to a process task to execute that process task automatically.

To connect an adapter to a process task, access the Integration tab (from the Process Definition form). From this tab, you can also map any adapter variables to their proper locations.

The following procedure shows you how to attach a process task adapter to a process task:

1. Open the Process Definition form, which is located in the Process Management folder.

   In the Oracle Identity Manager Workspace, the Process Definition form appears.

2. Select the process, which contains a task to which you want to attach an adapter. The selected process, along with its tasks, appears in the Process Definition form. For this example, the Solaris process has been selected.

3. Double-click the row header of the task to which you want to attach an adapter. The Editing Task window appears, containing information about the task (for example, the *Password Updated* process task).

4. Click the **Integration** tab.

5. Click **Add**.

   The Handler Selection window appears.

6. To access Oracle Identity Manager adapters, click the **Adapter** option.

   The adapters appear, which you can attach to the process task.

7. From this region, select the adapter that you want to attach to the process task, for example, the adp*SOLARISPASSWORDUPDATED* adapter.

   Tip: For classification purposes, the first three letters of each adapter's name are adp. For classification purposes, the first three letters of each adapter's name are *adp*.

8. From the Handler Selection window's toolbar, click **Save**.

   A dialog box appears, stating that the adapter was successfully added to the process task.

9. Click **OK**.

   The dialog box disappears, and the **Integration** tab is now active. This tab now displays the following:

   - The name of the adapter that is attached to the process task;
   - The status of the adapter; and
   - The names, descriptions, and mapping statuses of the adapter's variables.

   Note: An adapter can have one of three mapping statuses:

   Ready. This adapter has been successfully compiled, and all of its variables have been mapped correctly.

   Mapping Incomplete. This adapter has been successfully compiled, but at least one of its variables have not been mapped correctly.

   Adapter Unavailable. After this adapter had been compiled successfully, it was modified, and recompiled.

   Note: If an adapter does not have any mappable variables, the Adapter Variables region is empty. In addition, the Status field will display either *Ready* or *Adapter Unavailable*, depending on whether the adapter has to be recompiled.

Note: A mappable adapter variable either has been designated as *Resolve at Run time* or it is an adapter return variable.

Note: Once you attach the adapter to the process task, any responses that you defined for the adapter appear in the Responses tab of the Editing Task window.

10. Set the mappings for each variable that appears in the Adapter Variables region of the Integration tab. To do so, double-click the row header of the variable you want to map (for example, *SolarisUserID*).

    The Data Mapping for Variable window is displayed.

    Table 4–1 describes the fields of the Data Mapping for Variable window.

Table 4–1 Fields of the Data Mapping for Variable Window

Field Name	Description
Variable Name	This field displays the name of the adapter variable for which you are setting a mapping (for example, *SolarisUserID*).
Data Type	This field shows the data type of the adapter variable (for example, *String* is the data type for the *SolarisUserID* variable).
Map To	This field contains the types of mappings that you can set for the adapter variable (for example, *IT Resources*).When you map the adapter variable to a location or a contact, Oracle Identity Manager enables the adjacent combo box. From this combo box, select the specific type of location or contact to which you are mapping the adapter variable. In addition, if you map the adapter variable to a custom process form, and this form contains child table(s), Oracle Identity Manager enables the adjacent combo box. From this combo box, select the child table to which you are mapping the adapter variable. If you are not mapping the adapter variable to a location, contact, or child table of a custom process form, this combo box is grayed out.
Qualifier	This field contains the qualifiers for the mapping you selected in the **Map to** combo box (for example, *IT Asset*).
IT Asset Type	This field enables you to select a specific IT Resource (for example, *Solaris*) when you map an adapter variable to an IT Resource, and this variable's data type is *String*. If you are not mapping the adapter variable to an IT Resource, or the variable's data type is not *String*, this field does not appear.
IT Asset Property	This field enables you to select a specific field that will receive the results of the mapping (for example, *User Name*), when you map an adapter variable to an IT Resource, and this variable's data type is *String*. If you are not mapping the adapter variable to an IT Resource, or the variable's data type is not String, this field does not appear. **Important**: The **IT Asset Type** and **IT Asset Property** fields are included within this window for backward compatibility. The preferred way is to create an adapter variable with a data type of *IT Resource*, in which case these fields will not appear.

Table 4–1 (Cont.) Fields of the Data Mapping for Variable Window

Field Name	Description
Literal Value	When you map the adapter variable to a literal, use this field to specify the specific literal value. If you are not mapping the adapter variable to a literal, this field does not appear.
Old Value	By selecting this check box, you map the adapter variable to the value that was originally in the selected Qualifier field before modification. Process task adapters associated with process tasks are conditionally triggered when some field on the process form gets changed. If you click the Old Value option, and the process task is marked Conditional, then the value that is passed to the adapter is the previous value of the field, before it got modified. This is useful in cases of fields that accept passwords. For example, if you want to disallow setting the password to the same value, you can use the old value for comparison. If you are not mapping the adapter variable to a field that belongs to a child table of a custom process form, this check box is grayed out.

11. Complete the Map To, Qualifier, IT Asset Type, IT Asset Property, Literal Value, and Old Value fields.

    Note: For more information about which mappings to select, refer to the "Adapter Variable Mapping" section in *Oracle Identity Manager Reference*

12. On the toolbar, click **Save**. Then, click **Close**.

    The Data Mapping for Variable window disappears. The **Integration** tab is active again.

13. On the Editing Task window toolbar, click **Save**.

    The contents in the **Status** field change from *Mapping Incomplete* to *Ready*. In addition, the mapping statuses for the adapter's variables change from *No (N)* to *Yes (Y)*.

14. On the toolbar, click **Close**.

    The Editing Task window disappears, and the main screen is active once again. The adapter you added to the *Password Updated* task (*adpSOLARISPASSWORDUPDATED*) appears in the **Process Definition** form.

    This signifies that the *adpSOLARISPASSWORDUPDATED* process task adapter was attached to the *Password Updated* process task.

    Tip: Once you attach a process task adapter to a process task, a quick way to see the process and task to which it is connected is by accessing the Usage Lookup tab of the Adapter Factory form.

4.4 Removing Process Task Adapters from Process Tasks

If a process task adapter is no longer necessary for Oracle Identity Manager to complete the process task automatically, or when you wish to attach a different adapter to a process task, you must first remove the adapter that is attached to the

This procedure will show you how to remove a process task adapter from a process task.

4.4.1 To Remove a Process Task Adapter from a Process Task

1. Open the Process Definition form.

   In the Design Console workspace, the **Process Definition** form appears.

2. Select the process, which contains a task from which you want to remove an adapter (for example, the *Solaris* process).

   The selected process, along with its tasks, appears in the **Process Definition** form.

3. Double-click the row header of the process task from which you want to remove the adapter (for example, the *Password Updated* task).

   The Editing Task window appears, containing information about the process task. Click the Integration tab.

4. Click the **Integration** tab.

   The Integration tab displays information about the adapter that is attached to the process task.

5. Click **Remove**.

   A dialog box appears, asking if you want to remove the adapter from the process task.

6. Click **OK**.

   A dialog box appears, signifying that the adapter has been removed from the process task.

7. Click **OK**.

   The contents of the adapter no longer appear in the Integration tab.

8. On the toolbar, click **Close**.

   The Editing Task window disappears, and the main screen is active once again. The adapter that was once linked to the *Password Updated* task *(adpSOLARISPASSWORDUPDATED)* no longer appears in the child table of the **Process Definition** form.

   This signifies that you have removed the adapter from the process task.

5

Applying Task Assignment Adapters

This chapter describes the process of applying task assignment adapters. It contains the following topics:

5.1 Overview

For a process task that must be completed manually, you can configure Oracle Identity Manager to automate the assignment of the task to either a specific user or to a user who belongs to a particular user group.

The Java code that enables Oracle Identity Manager to assign a process task to a user or group automatically is known as a task assignment adapter.

For efficiency purposes, a task assignment adapter can be used repeatedly for different process tasks. In addition, multiple task assignment adapters can be designated to be associated with a particular task. Therefore, Oracle Identity Manager selects the task assignment adapter for the process task. This occurs through task assignment rules.

Task assignment rules are criteria. These standards enable Oracle Identity Manager to select the task assignment adapter that is to be used to allocate the process task to the target user or group.

> **Note:** In other words, the task assignment rule allows Oracle Identity Manager to decide whether to assign a process task to a user or group. The task assignment adapter enables Oracle Identity Manager to determine which user or group will be the recipient of the process task.

Each task assignment adapter has a task assignment rule associated with it. In addition, every rule has a priority number, which indicates the order in which Oracle Identity Manager will trigger it.

For this example, Oracle Identity Manager will trigger the Associate Adapter with User rule first (because it has the highest priority). If the condition of this rule is TRUE, then it is successful. As a result, Oracle Identity Manager will associate the related task assignment adapter (the Assign Task to User adapter) with the Get Solaris UUID process task.

On the other hand, when the condition of a rule is FALSE, then the rule has failed. Oracle Identity Manager will then trigger the rule with the next highest priority. If this rule is successful, then Oracle Identity Manager will assign the designated adapter to the target process task.

So, in this example, if the Associate Adapter with User rule fails, then Oracle Identity Manager will trigger the Associate Adapter with Group rule. If this rule is successful, then Oracle Identity Manager will attach the related task assignment adapter (the Assign Task to Group adapter) to the Get Solaris UUID process task.

After assigning a rule to a task assignment adapter, if this type of adapter contains adapter variables, then you must map these variables to their proper locations. Otherwise, the adapter will not be functional.

Finally, when a task assignment adapter either becomes invalid, or is no longer necessary for Oracle Identity Manager to allocate the process task to a user or group, you must remove the adapter from the task.

The following sections demonstrate how to attach a task assignment adapter to a process task, and remove a task assignment adapter from a process task.

5.2 Attaching Task Assignment Adapters to Process Tasks

In the Chapter 3, "Creating Adapters", you learned how to create a task assignment adapter. You must attach it to a process task so that Oracle Identity Manager can automate the assignment of the task to a user or group.

To connect a task assignment adapter to a process task, access the Assignment tab (from the Process Definition form). From this tab, you can also map any adapter variables to their proper locations.

This procedure will show you how to attach a task assignment adapter to a process task.

5.2.1 To Attach a Task Assignment Adapter to a Process Task

1. Open the Process Definition form, which is located in the Process Management folder.

   Within the Oracle Identity Manager workspace, the Process Definition form appears.

2. Select the process, which contains a task to which you want to attach an adapter.

   The selected process, along with its tasks, appears in the Process Definition form.

3. Double-click the row header of the task to which you want to attach a task assignment adapter.

   The Editing Task window appears, containing information about the task (for example, the Get Solaris UUID process task).

4. Click the **Assignment** tab. The Assignment dialog box is displayed.

5. From this tab, click **Add**.

   A blank row appears within the Assignment tab.

   The following table lists the relevant fields of the Assignment tab:

Field Name	Description
Priority	From this field, set the priority number for the associated task assignment rule.
Rule	From this lookup field, select the rule that will determine if the associated adapter will be used to automate the assignment of the process task to a user or group.
Target Type	From this lookup field, specify whether the task is to be assigned to an Oracle Identity Manager user or group.
Adapter	From this lookup field, select the adapter that is to be associated with the designated task assignment rule.
Adapter Status	This field displays the mapping status of the adapter's variables. To learn more about the various mapping statuses for an adapter, refer to the "Attaching Process Task Adapters to Process Tasks" on page 4-3.

6. Double-click the **Priority** field. From this field, set the priority number for the associated task assignment rule.
7. Double-click the **Rule** lookup field. From the Lookup dialog box that is displayed, select the rule that will determine if the associated adapter will be used to automate the assignment of the process task to a user or group.
8. Double-click the **Target Type** lookup field. From the Lookup dialog box that is displayed, specify whether the task is to be assigned to an Oracle Identity Manager user or group.
9. Double-click the **Adapter** lookup field. From the Lookup dialog box that is displayed, specify the task assignment adapter that is to be associated with the rule you selected in Step 7 of this procedure.
10. On the toolbar that is displayed within the Assignment tab, click **Save**.

    The mapping status of the task assignment adapter variables is displayed within the Adapter Status field. Use the following table to decide which action to perform, based on the adapter's mapping status.

Mapping Status	Action
Ready	The adapter does not have any variables that can be mapped. In other words, none of the adapter variables are return variables or have been designated as Resolve at Run time. So, proceed to Step 14 of this procedure.
Mapping Incomplete	At least one of the adapter's variable must be mapped. So, proceed to Step 11 of this procedure.
Adapter Unavailable	After the adapter had been compiled successfully, it was modified. As a result, you must recompile the adapter.

Note: To learn more about the various mapping statuses for an adapter, refer to the "Attaching Process Task Adapters to Process Tasks" on page 5-2.

11. Click **Map**.

    The Adapter Variables window appears. It displays the following information:

- The name of the task assignment adapter that is attached to the process task;
- The status of the adapter; and
- The mapping statuses, names, and descriptions of the adapter's variables.

12. Set the mappings for each variable that appears in the **Adapter Variables** region of this window. To do so, double-click the row header of the variable you want to map (for example, UUID).

    The Edit Data Mapping for Variable dialog box is displayed.

    Table 5–1 lists the fields of the Edit Data Mapping for Variable dialog box is displayed.

Table 5–1 Fields of the Edit Data Mapping for Variable Dialog Box

Field Name	Description
Variable Name	This field displays the name of the adapter variable for which you are setting a mapping (for example, UUID).
Data Type	This field shows the data type of the adapter variable (for example, *String* is the data type for the UUID variable).
Map To	This field contains the types of mappings that you can set for the adapter variable (for example, IT Resources). When you map the adapter variable to a location or a contact, Oracle Identity Manager enables the adjacent combo box. From this combo box, select the specific type of location or contact to which you are mapping the adapter variable. In addition, if you map the adapter variable to a custom process form, and this form contains child table(s), then Oracle Identity Manager enables the adjacent combo box. From this combo box, select the child table to which you are mapping the adapter variable. If you are not mapping the adapter variable to a location, contact, or child table of a custom process form, this combo box is grayed out.
Qualifier	This field contains the qualifiers for the mapping you selected in the Map To combo box (for example, IT Asset).
IT Asset Type	This field enables you to select a specific IT Resource (for example, *Solaris*) when you map an adapter variable to an IT Resource, and this variable's data type is String. If you are not mapping the adapter variable to an IT Resource, or the variable's data type is not String, this field does not appear.
IT Asset Property	This field enables you to select a specific field that will receive the results of the mapping (for example, Unique ID), when you map an adapter variable to an IT Resource, and this variable's data type is *String*. If you are not mapping the adapter variable to an IT Resource, or if the variable's data type is not String, then this field does not appear. I**mportant**: The IT Asset Type and IT Asset Property fields are included within this window for backward compatibility. The preferred way is to create an adapter variable with a data type of IT Resource, in which case these fields will not appear.
Literal Value	When you map the adapter variable to a literal, use this field to specify the specific literal value. If you are not mapping the adapter variable to a literal, then this field does not appear.

Table 5–1 (Cont.) Fields of the Edit Data Mapping for Variable Dialog Box

Field Name	Description
Old Value	By selecting this check box, you map the adapter variable to the value that was originally in the selected Qualifier field before modification. Process task adapters associated with process tasks are conditionally triggered when some field on the process form is changed. If you click the Old Value option, and the process task is marked Conditional, then the value that is passed to the adapter is the previous value of the field. This is useful in cases of fields that accept passwords. For example, if you want to disallow setting the password to the same value, you can use the old value for comparison. If you are not mapping the adapter variable to a field that belongs to a child table of a custom process form, this check box is grayed out.

13. Complete the Map To, Qualifier, IT Asset Type, IT Asset Property, Literal Value, and Old Value fields.

    Note: For more information about which mappings to select, refer to the "Adapter Mapping Information" section in *Oracle Identity Manager Reference*.

14. On the toolbar, click **Save**. Then, click **Close**.

    The Edit Data Mapping for Variable window disappears. The Adapter Variables dialog box is active again.

    The contents in the Status field change from Mapping Incomplete to Ready. In addition, the mapping statuses for the adapter's variables change from No (N) to Yes (Y).

15. Click **Save**. Then, click **Close**.

    The Adapter Variable dialog box disappears, and the Assignment tab is active once again.

    The adapter that you assigned to the process task (for example, *Assign Solaris Task*) now has a status of Ready.

16. From the toolbar that appears within the Assignment tab, click **Save** and **Close**

    The Assignment tab disappears, and the main screen is active once again. This signifies that the task assignment adapter is attached to the process task.

    Note: Once you attach a task assignment adapter to a process task, a quick way to see the process and the task to which it is connected is by accessing the Usage Lookup tab of the Adapter Factory form.

5.3 Removing Task Assignment Adapters from Process Tasks

When a task assignment adapter either becomes invalid, or is no longer necessary for Oracle Identity Manager to allocate the process task to a user or group, you must remove the adapter from the task.

5.3.1 To Remove a Task Assignment Adapter from a Process Task

To detach a task assignment adapter from a process task, perform the following tasks:

1. Open the Process Definition form.

   The Process Definition form appears in the Design Console workspace.

2. Select the process, which contains a task from which you want to remove an adapter (for example, the Solaris 8 process).

   The selected process, along with its tasks, appears in the Process Definition form.

3. Double-click the row header of the process task from which you want to remove the adapter (for example, the Get Solaris UUID task).

   The Editing Task dialog box is displayed, containing information about the process task.

4. Click the **Assignment** tab.

   The Assignment tab appears, displaying information about the adapter that is attached to the process task.

5. Highlight the row, containing the adapter that you want to remove from the process task.

6. Click **Delete**. The adapter no longer appears within the Assignment tab.

7. Click **Save**. Then, click **Close**.

   The Assignment tab disappears, and the Main Screen is active once again. This signifies that the task assignment adapter is removed from the process task.

6

Understanding Rule Generators

This chapter describes the rule generators. It contains the following topics:

6.1 Overview

To perform field validations and enter default values into forms, which either come packaged with Oracle Identity Manager or are created by Oracle Identity Manager users, certain business rules must be applied. For example, for the Users form, you might want Oracle Identity Manager to generate the User ID automatically by concatenating the user's first name and last name.

To do this, you must create a specific type of adapter, which is designed to modify the field value in a form. You can do this by using the Adapter Factory form. Then, when you attach this adapter to the form, Oracle Identity Manager will automatically update the field value for all records of that form, and save this information to the Oracle Identity Manager database.

This type of adapter, which can generate, modify, or verify the value of a form field automatically, is called a rule generator. Oracle Identity Manager triggers a rule generator on preinsert and preupdate.

If you create a rule generator and it contains adapter variables, then you must map these adapter variables to their proper locations. Otherwise, the adapter will not be functional.

Also, you can attach this adapter to a provisioning process for the adapter to work. Then, once the process is provisioned to a target user or organization, Oracle Identity Manager will trigger the associated rule generator.

On occasion, a rule generator, which has been assigned to a provisioning process, might no longer be needed for the process to be completed. If this happens, then you can remove the rule generator from the provisioning process. Similarly, after you attach one rule generator to a form field, you can connect a different rule generator to that form field. When this occurs, you must first remove the rule generator that is currently attached to the form field.

The following sections will show you how to:

- Map the adapter variables of a rule generator.

- Associate rule generators with provisioning processes.
- Remove a rule generator from a form field.

6.2 Mapping Rule Generator Adapter Variables

In the Chapter 3, "Creating Adapters", you learned how to create a rule generator. Now, you must map the adapter variables of the rule generator to their proper locations. This will ensure that the adapter will work.

To map these adapter variables, access the Data Object Manager form from the Development Tools/Business Rule Definition folder of the Design Console.

To map the adapter variables of a rule generator to their proper locations:

1. Open the Data Object Manager form. In the Design Console workshops, the Data Object Manager form is displayed.

   The following table lists and describes the various regions of the Data Object Manager form:

Name	Description
Form Description Field	From this lookup field, select the form that contains the field to which you are attaching the rule generator.
Data Object Field	This field displays the name of the data object, which is represented by the selected form.
Attach Handlers Tab	This tab displays: ■ The rule generators that are attached to the selected form. ■ The execution schedule of the rule generators associated with this form. ■ The order in which Oracle Identity Manager will run the rule generators. ■ Insert, update, and delete permissions for user groups.
Map Adapters Tab	This tab displays: ■ The names of the rule generators that are associated with the form; ■ The status of these adapters. ■ The names, descriptions, and mapping statuses of the rule generators' adapter variables. **Note**: The Map Adapters tab is grayed out until an adapter is assigned to the current data object. For more information about assigning an adapter to a data object, refer to *Oracle Identity Manager Administrative and User Console Guide*.

2. Double-click the **Form Description** field. A Lookup dialog box appears with the forms to which you can attach rule generators.
3. Select the form you want (for example, *Solaris*). Then, click **OK**.
4. On the toolbar, click **Save**.

   The selected form, the form's data object, and the rule generator adapters associated with the form appear. In addition, Oracle Identity Manager enables the Map Adapters tab.

For this example, the Solaris form has been selected. Its data object `Thor.CarrierBase.tcUD_SOLARIS` appears, along with the four rule generator adapters associated with it (`adpCONVERTTOLOWERCASE`, `adpSOLARISHMDSTRINGGEN`, `adpSETSOLARISASSET`, and `adpSETPASSWORDFROMMAIN`). Oracle Identity Manager will trigger these four rule generators on preinsert.

Based on the sequence numbers of these adapters, Oracle Identity Manager will trigger the `adpCONVERTTOLOWERCASE` adapter first, followed by the `adpSOLARISHMDSTRINGGEN`, `adpSETSOLARISASSET`, and `adpSETPASSWORDFROMMAIN` adapters respectively.

Tip: To change the sequence of triggering a rule generator:

1. Click **Assign**. The Event Handlers dialog box is displayed.
2. Select the rule generator from.
3. Click the up arrow and down arrow buttons to modify the order of the rule generator.

For these rule generators to work properly, you must map the adapter variables to their proper locations.

5. Click the **Map Adapters** tab.
6. From the Name combo box, select the rule generator, which has adapter variables that can be mapped (for example, the `adpCONVERTTOLOWERCASE` rule generator).

   The Map Adapters tab now displays the following:

   - The name of the rule generator that is to be attached to the form.
   - The status of the rule generator.
   - The names, descriptions, and mapping statuses of the rule generator's adapter variables.

   Note: To learn more about the various mapping statuses for an adapter, refer to the "Attaching Process Task Adapters to Process Tasks" on page 4-3.

7. Set the mappings for each variable that appears in the Adapter Variables region of the Map Adapters tab. To do so, double-click the row header of the variable you want to map (for example, *Data*). The Data Mapping for Variable dialog box is displayed.

   Table 6–1 describes the various fields of the Data Mapping for Variable dialog box.

Table 6–1 Fields of the Data Mapping for Variable Dialog Box

Field Name	Description
Variable Name	This field displays the name of the adapter variable for which you are setting a mapping (for example, Data).
Data Type	This field shows the data type of the adapter variable (for example, String is the data type for the Data adapter variable).

Table 6–1 (Cont.) Fields of the Data Mapping for Variable Dialog Box

Field Name	Description
Map To	This field contains the source and target locations of the mappings you can set for the adapter variable (for example, User Definition). When you map the adapter variable to a location or a contact, Oracle Identity Manager enables the adjacent combo box. From this combo box, select the specific type of location or contact to which you are mapping the adapter variable. If you are not mapping the adapter variable to a location or contact, this combo box is grayed out.
Qualifier	This field contains the qualifiers for the mapping you selected in the Map To combo box (for example, User Login).
IT Asset Type	This field enables you to select a specific IT Resource (for example, *Solaris*) when you map an adapter variable to an IT Resource, and this variable's data type is String. If you are not mapping the adapter variable to an IT Resource, or the variable's data type is not String, this field does not appear.
IT Asset Property	This field enables you to select a specific field that will receive the results of the mapping (for example, *User Name*), when you map an adapter variable to an IT Resource, and this variable's data type is String. If you are not mapping the adapter variable to an IT Resource, or the variable's data type is not *String*, this field does not appear. **Important**: The IT Asset Type and IT Asset Property fields are included within this window for backward compatibility. The preferred way is to create an adapter variable with a data type of IT Resource, in which case these fields will not appear.
Literal Value	When you map the adapter variable to a literal, type the name of the specific literal in this field (for example, IBM). If you are not mapping the adapter variable to a literal, this field does not appear.

Complete the Map To, Qualifier, IT Asset Type, IT Asset Property, and Literal Value fields.

Note: For more information about which mappings to select, refer to the "Adapter Mapping Information" section in *Oracle Identity Manager Reference*.

8. Click **Save**. Then, click **Close**

   The Data Mapping for Variable window disappears. The Map Adapters tab is active again.

9. On the main screen toolbar, click **Save**.

Repeat Steps 7 and 8 for all adapter variables that can be mapped.

The contents in the Status field change from Mapping Incomplete to Ready. In addition, the mapping statuses for the adapter variables change from *No (N)* to *Yes (Y)*.

This signifies that all the adapter variables for the rule generator adapter have been mapped correctly. You are now ready to attach this rule generator to a provisioning process, so it can be triggered after the process is provisioned to a target user or organization.

> **Tip:** When you map all the adapter variables for a rule generator that is associated with a form, a quick way to see the form to which it is attached as well as the execution schedule of the rule generator, is by accessing the Usage Lookup tab of the Adapter Factory form.
>
> After the rule generator is assigned to a process, and the process is provisioned, the rule generator will be executed by Oracle Identity Manager.

6.3 Associating Rule Generators with Processes

After you map the adapter variables of a rule generator to their proper locations, you must attach it to a provisioning process. Then, once the process is provisioned to a target user or organization, Oracle Identity Manager will trigger the associated rule generator.

Similarly, when a rule generator, which has been assigned to a provisioning process, is no longer needed for the process to be completed, you must remove the rule generator from the provisioning process.

To assign a rule generator to a provisioning process or remove a rule generator from a provisioning process, access the Event Handlers/Adapters tab in the Process Definition form. This form can be found in the Process Management folder.

6.4 Removing Rule Generators from Form Fields

Sometimes, after you attach a rule generator to a form field, you can connect a different rule generator to that form field. When this occurs, you must first remove the rule generator that is currently attached to the form field.

> **Caution: If you remove a rule generator from a form and if the class name of the form's data object matches the table name of a provisioning process, then you will not be able to assign the rule generator to that provisioning process.**
>
> **For example, suppose the adpCONVERTTOLOWERCASE rule generator is removed from the Solaris form. If the class name of the form's associated data object is UD_SOLARIS, then the rule generator cannot be assigned to any provisioning process with a table name of UD_SOLARIS.**

To remove a rule generator from a form field, perform the following steps:

1. Open the Data Object Manager form.
2. Select the form that contains a rule generator you want to remove.
3. The selected form, along with its rule generators, appear in the Data Object Manager form.
4. Click the rule generator that you want to remove from the form field.
5. Click **Delete**.

The selected rule generator no longer appears in the Data Object Manager form. This indicates that you have removed the rule generator from the form field.

Caution: If you attempt to remove a rule generator from a form field, and if an error box appears, then the adapter has already been associated with a provisioning process. First, detach the rule generator from the process. Then, you can remove it from the form field.

7

Using Prepopulate Adapters

This chapter describes the prepopulate adapters. It contains the following topics:

7.1 Overview

Sometimes, a user-created form contains fields that can be populated by Oracle Identity Manager and fields into which an Oracle Identity Manager user must enter data. When the information that the user types into a field is contingent upon the data that appears in a system-generated field, Oracle Identity Manager must first populate this field. When the form is displayed, an Oracle Identity Manager user can view the system-generated data, and as a result, enter information into the appropriate fields.

For this to happen, you must create a specific type of rule generator. Then, by attaching it to the field that is designated to be system-generated, Oracle Identity Manager will automatically populate this field with the appropriate information, without saving this information to the Oracle Identity Manager database.

This type of rule generator is known as a prepopulate adapter. The data, which is generated by a prepopulate adapter can appear automatically or manually entering it. Oracle Identity Manager displays this information automatically when the Auto-prepopulate check box is selected for a provisioning process. When this check box is cleared, an Oracle Identity Manager user must manually generate the displaying of the data that is generated by the prepopulate adapter. To do this, click the prepopulate button on the form section of the Direct Provisioning wizard in the Web client, while provisioning the form to a user.

You can repeatedly use a prepopulate adapter for different form fields. In addition, you can designate multiple prepopulate adapters to be associated with a particular field. As a result, Oracle Identity Manager must know which prepopulate adapter it must select for the form field. This occurs by using the prepopulate rules.

prepopulate rules are criteria. These rules enable Oracle Identity Manager to select one prepopulate adapter, which is associated with a form field, when this prepopulate adapter is assigned to the field.

Each prepopulate adapter has a prepopulate rule associated with it. In addition, every rule has a priority number, which indicates the order in which Oracle Identity Manager will trigger it.

For this example, Oracle Identity Manager will trigger the Rule for Uppercase User ID rule first because it has the highest priority. If the condition of this rule is TRUE, then it

is successful. As a result, Oracle Identity Manager will attach the related prepopulate adapter (the Display Uppercase Letters for User ID adapter) to the User ID field.

On the other hand, when the condition of a rule is FALSE, then the rule has failed. Oracle Identity Manager will then trigger the rule with the next highest priority. If this rule is successful, then Oracle Identity Manager will attach the associated adapter to the designated field.

So, in this example, if the Rule for Uppercase User ID rule fails, Oracle Identity Manager will trigger the Rule for Lowercase User ID rule. If this rule is successful, Oracle Identity Manager will attach the related prepopulate adapter (the Display Lowercase Letters for User ID adapter) to the User ID field.

After assigning a rule to a prepopulate adapter, if this type of adapter contains adapter variables, then you must map these adapter variables to their proper locations. Otherwise, the adapter will not be functional.

Finally, when a prepopulate adapter, which has been associated with a field, is no longer valid, you must remove the adapter from the field.

7.2 Attaching Prepopulate Adapters to Form Fields

To attach a prepopulate adapter to a form field, perform the following steps:

1. Select the field to which a prepopulate adapter will be attached.
2. Select the rule that will determine if the adapter will be used to populate the designated field with information.
3. Select the adapter that will be associated with the designated field.
4. Set the priority number of the selected rule.
5. Map the adapter variables of the prepopulate adapter to their proper locations.

   Note: To attach a prepopulate adapter to a form field, you must ensure the following:

   - The form is not in an active state. Otherwise, create a new form version.
   - After attaching the adapter, you must activate the form to be able to use it.

6. Open the Form Designer form.
7. Query for the form to which you want to attach a prepopulate adapter (for example, Solaris).
8. Click the **prepopulate** tab.

   The prepopulate adapters, which have already been attached to the form you queried, appear within this tab.

   Note: If no adapters have been attached to a form field, then the prepopulate tab will be empty.

9. Click **Add**.

   The prepopulate Adapters dialog box is displayed.

Table 7–1 lists and describes the fields of the prepopulate Adapters dialog box.

Table 7–1 Fields of the Prepopulate Adapters Dialog Box

Name	Description
Field Name	This combo box contains a list of all of the form fields to which a prepopulate adapter can be attached.
Rule	From this lookup field, select the rule that will determine if the associated adapter will be used to populate the designated form field with information.
Adapter	From this lookup field, select the adapter that will be associated with the designated field.
Order	From this field, set the priority number of the selected rule.
Adapter Status	This field displays the mapping status of the adapter variables. To learn more about the various mapping statuses for an adapter, refer to the "Attaching Process Task Adapters to Process Tasks" on page 4-3.
Adapter Variables	This area displays the following: ■ Mapped: The mapping statuses of the adapter's variables. "Y" indicates that an adapter variable has been mapped properly; "N" indicates that this variable has not been mapped correctly. ■ Name: The names of the adapter variables. ■ Mapped to: The form fields to which the variables are mapped If an adapter variable is not yet mapped, then the corresponding cell in this column will be empty.

10. From the **Field Name** combo box, select the form field, such as User ID, to which the prepopulate adapter will be attached.
11. Double-click the **Rule** lookup field. From the Lookup dialog box that is displayed, select the rule that will determine if the associated adapter will be used to populate the designated form field with information (for example, Rule for Lowercase User ID).
12. Double-click the **Adapter** lookup field. From the Lookup dialog box that is displayed, choose the adapter that will be associated with the field you selected in Step 10, for example, Display Lowercase Letters for User ID.
13. In the **Order** field, enter the priority number of the rule you selected in Step 11, for example, 2.
14. On the prepopulate Adapters window toolbar, click **Save**.
15. Mapping Incomplete appears within the Adapter Status field. This signifies that the adapter you selected contains variables that have not been mapped correctly. These variables can be mapped to their proper locations. Otherwise, the adapter will not work.
16. Set the mappings for each variable that appears in the Adapter Variables region of the prepopulate Adapters window. To do so, double-click the row header of the variable you want to map, for example, UserID.

    The Map Adapter Variables window is displayed.

    Table 7–2 describes the fields of the Map Adapter Variables window.

Table 7–2 Felds of the Map Adapter Variables Window

Field Name	Description
Variable Name	This field displays the name of the adapter variable for which you are setting a mapping (for example, UserID).
Data Type	This field shows the data type of the adapter variable (for example, *String* is the data type for the UserID adapter variable).
Map To	This field contains the types of mappings that you can set for the adapter variable (for example, Process Data). When you map the adapter variable to a location or a contact, Oracle Identity Manager enables the adjacent combo box. From this combo box, select the specific type of location or contact to which you are mapping the adapter variable. If you are not mapping the adapter variable to a location or contact, this combo box is grayed out.
Qualifier	This field contains the qualifiers for the mapping you selected in the **Map to** combo box (for example, User ID).
IT Asset Type	This field enables you to select a specific IT Resource (for example, Solaris) when you map an adapter variable to an IT Resource, and this variable's data type is String. If you are not mapping the adapter variable to an IT Resource, or the variable's data type is not String, this field does not appear.
IT Asset Property	This field enables you to select a specific field that will receive the results of the mapping (for example, *User Name*), when you map an adapter variable to an IT Resource, and this variable's data type is String. If you are not mapping the adapter variable to an IT Resource, or the variable's data type is not String, then this field does not appear. **Important**: The IT Asset Type and IT Asset Property fields are included within this window for backward compatibility. The preferred way is to create an adapter variable with a data type of IT Resource, in which case these fields will not appear.
Literal Value	When you map the adapter variable to a literal, use this field to specify the specific literal value. If you are not mapping the adapter variable to a literal, this field does not appear.

17. Complete the Map To, Qualifier, IT Asset Type, IT Asset Property, and Literal Value fields.

> **Note:** For more information about which mappings to select, refer to the "Adapter Mapping Information" section in *Oracle Identity Manager Reference*.

18. On the Map Adapter Variable window toolbar, click **Save**. Then, click **Close**.

The Map Adapter Variables window disappears. The prepopulate Adapters window is active again.

The text in the Adapter Status field changes from Mapping Incomplete to Ready. In addition, the mapping statuses for the adapter's variables change from No (N) to Yes (Y).

19. On the prepopulate Adapters window toolbar, click **Close**.

The prepopulate Adapters window disappears, and the Form Designer form is active again. The prepopulate adapter, which you attached to the User ID form field (Display Lowercase Letters for User ID), appears in the prepopulate tab of the Results of 1Q Sales 2003 form.

After a process, which references this form, is provisioned to a target user or organization, the form will appear. Oracle Identity Manager will then check to see if the prepopulate rule, which has the highest priority, is valid. If so, then Oracle Identity Manager will assign the associated prepopulate adapter to the designated field (User ID), and execute it. At this point, one of the following actions occur:

- If the Auto-prepopulate check box is selected for the provisioning process, Oracle Identity Manager will display the data that is generated by the prepopulate adapter automatically.
- If the Auto-prepopulate check box is cleared, an Oracle Identity Manager user must manually trigger the displaying of the data that is generated by the prepopulate adapter. To do this, the administrator must click the prepopulate button on the form section of the direct provisioning wizard in the Web client, while provisioning the form to a user.

Tip: Once you allocate a prepopulate adapter to a form field, and assign a prepopulate rule to the adapter, a quick way to see the association among the adapter, the form field, and the rule is by accessing the Usage Lookup tab of the Adapter Factory form.

7.3 Removing Prepopulate Adapters from Form Fields

If a prepopulate adapter, which has been associated with a form field, is no longer valid, then you must remove the adapter from the field.

Note: Before removing the prepopulate adapter from a form field, you must create a new version of the form.

To remove a prepopulate adapter from a form field:

1. Select the prepopulate adapter that you want to remove.
2. Click **Delete**. The prepopulate adapter is removed from the form field. It cannot be triggered when the form is launched.
3. After removing the adapter, you must activate the form.

8

Managing Entities

Similar to rule generator adapters, entity adapters are also responsible for generating, modifying, or verifying the value of a form field automatically, and saving this information to the Oracle Identity Manager database. However, the differences between rule generators and entity adapters are as follows:

- **Execution schedule**. Entity adapters can be triggered by Oracle Identity Manager on preinsert, preupdate, predelete, postinsert, postupdate, and postdelete. A rule generator adapter can be executed only on preinsert and preupdate.
- **Manual field value modification**. The adapter populates the form field to which an entity adapter is attached. An Oracle Identity Manager user should not edit this value because the entity adapter will overwrite this modification. As a result, the modification will not be saved to the database.

  Similarly, the adapter also populates the form field to which a rule generator adapter is attached. However, an Oracle Identity Manager user can edit this value because this modification will take precedence over the value that the rule generator adapter generates. Because of this, the modification will be saved to the database.
- **Background color of form field**. If a rule generator is attached to a form field, then the field will appear in a particular background color such as pink. This is a visual indicator that the field has a rule generator attached to it. On the other hand, when an entity adapter is attached to a form field, the field will not have a distinct background color.

Note: For more information about mapping the variables of an entity adapters with Oracle Identity Manager forms and/or provisioning processes, refer to the procedures in the Chapter 6, "Understanding Rule Generators" on page 6-1.

9

Compiling Adapters

When you are creating multiple adapters and compiling them, you can do so in one of two ways. First, you can access each adapter, and compile it individually. However, this method consumes a lot of time and effort.

A more efficient approach is to either select the adapters that you want to compile, and compile them in one shot, or, when necessary, to compile all adapters that exist in the Oracle Identity Manager database, by clicking a button.

9.1 Automatic Compilation of Adapters

From release 9.1.0 onward, adapters are compiled automatically when you import connector files by using the Deployment Manager. The compiled adapter classfiles are stored in the Oracle Identity Manager database, as opposed to the filesystem, from where they are loaded at run time. The following two APIs have been added to compile adapters programmatically:

- `public void compileAdapter (String adapterName)`: This API compiles a single adapter and stores the compiled classfile in the database. It takes the name of the adapter as a parameter. If the adapter is not found or if there are any errors, then the API throws an appropriate exception.
- `public void compileAll`: This API compiles all adapters in a system. If it encounters any errors during compilation, then it throws an exception of the type `tcBulkException`. This exception comprises all the individual errors that the API encounters during compilation.

However, you can modify the adapters manually if you make changes.

Note: You must set the path of the JDK directory in the `XL.CompilerPath` system property. If this is not done, then an error is encountered during the adapter compilation stage of the process performed when you import an XML file by using the Deployment Manager.

Refer to the "The System Configuration Form" section in *Oracle Identity Manager Design Console Guide* for information about setting values of system properties.

9.2 Compiling Adapters Manually

The Adapter Manager form is located in the Development Tools folder. It is used to compile multiple adapters simultaneously.

To manually compile multiple adapters, perform the following steps:

1. Open the Adapter Manager form.
2. To compile every adapter that resides within the Oracle Identity Manager database, select the **Compile All** option.

   To compile multiple adapters, select the adapters you want to compile. Then, select the **Compile Selected** option.

   To compile all adapters that do not have an OK status, select the **Compile Previously Failed** option.
3. Click the **Start** button.

   Oracle Identity Manager will compile the adapters that match the criteria you specified in Step 2.

   Tip: Oracle Identity Manager lets you look at the record of any adapter that appears within the Adapter Manager form. By doing so, you can see detailed information about the adapter.

   To view an adapter's record, select the desired adapter. Then, either double-click its row header, or right-click the adapter, and select the Launch Adapter command from the menu that appears.

10

Exporting and Importing Adapters

This chapter describes the exporting and importing adapters.

Sometimes, when you create an adapter for one database, you might want to use it with another database. One approach is to manually reconstruct the adapter for that database. However, besides this method being unproductive in terms of time, money, and resources, the developer might inadvertently introduce an error while re-creating the adapter. As a result, after the process with which this adapter is associated is provisioned to the target user or organization, the adapter will not be able to perform its designated function, and Oracle Identity Manager will not be able to provision the corresponding resource to that user or organization.

An alternate method, which is more efficient, and eliminates the chance for error, is to:

1. Create a bug-free adapter and apply it properly within one database. The Deployment Manager, internally, builds a `*.xml` file, which contains this adapter and all of its internal variable mappings.

   The Deployment Manager also exports this file into a designated location.

2. Retrieve the file from the location, and import it into the target database.

You can use Deployment Manager to transfer adapters from one database to another. Refer to the *Oracle Identity Manager Administrative and User Console Guide* for instructions about how to use Deployment Manager.

Note: Any adapter that you import into a target database must be recompiled within that database. Also, you should copy the third-party JAR files, after migrating the adapter. Otherwise, the adapter will not work.

11

Creating and Testing a Remote Manager IT Resource

This chapter describes the tasks for creating and testing a Remote Manager IT Resource. It contains the following topics:

Remote Manager is an Oracle Identity Manager component that acts like a proxy in directly communicating with a third-party system. The Remote Manager is used to invoke nonremotable APIs through Oracle Identity Manager and APIs that do not support Secure Sockets Layer (SSL) over secure connections.

After installing the Remote Manager and establishing the trust relation between the Oracle Identity Manager Server and the Remote Manager (trusting the certificate), you can create an IT Resource for the Remote Manager and then test it.

11.1 Postinstallation Configuration

After installing the Remote Manager, you can ensure that the certificate is trusted between the application server and the Remote Manager. To do so, first open the Remote Manager form in the Administration folder of the Design Console. The Remote Manager form shows all Remote Managers that are connected but not necessarily "trusted".

Perform the following steps to ensure that the trust relation between the application server and the Remote Manager is established through the certificate. In this procedure, the JBoss Application Server is used as an example. The keytool utility is used to import/export the certificates.

1. Using a command prompt, open the directory path and use the keytool utility to list the certificate fingerprints:

   XLREMOTE_HOME\xlremote

2. Enter the command:

   JAVA_HOME\jre\bin\keytool -list -keystore config\.xlkeystore

3. Enter the default password for xellerate keystore: `xellerate`

   ```
   Your keystore contains 1 entry

          xell, Jan 7, 2005, keyEntry,

          Certificate fingerprint (MD5):
   ```

B0:F2:33:C8:69:E4:25:A3:CB:59:E8:51:27:EE:5C:52

The certificate fingerprint is marked in bold. Compare this to the list of certificates in the keystore.

4. Open the Java SDK folder used for the application server. Again, enter the path and use the keytool to list the certificates in the keystore:

   JAVA_HOME\jre\lib\security\cacerts

5. Enter the following command to see the list of trusted certificates:

   JAVA_HOME\bin\keytool -keystore cacerts -storepass changeit -storetype jks -provider *provider_name*

   The output showing the keystore entries are as follows:

```
Your keystore contains 25 entries
equifaxsecureebusinessca1, Jul 23, 2003, trustedCertEntry,
Certificate fingerprint (MD5): 64:9C:EF:2E:44:FC:C6:8F:52:07:D0:51:73:8F:CB:3D
verisignclass4ca, Jun 29, 1998, trustedCertEntry,
Certificate fingerprint (MD5): 1B:D1:AD:17:8B:7F:22:13:24:F5:26:E2:5D:4E:B9:10
entrustglobalclientca, Jan 9, 2003, trustedCertEntry,
Certificate fingerprint (MD5): 9A:77:19:18:ED:96:CF:DF:1B:B7:0E:F5:8D:B9:88:2E
gtecybertrustglobalca, May 10, 2002, trustedCertEntry,
Certificate fingerprint (MD5): CA:3D:D3:68:F1:03:5C:D0:32:FA:B8:2B:59:E8:5A:DB
entrustgsslca, Jan 9, 2003, trustedCertEntry,
Certificate fingerprint (MD5): 9D:66:6A:CC:FF:D5:F5:43:B4:BF:8C:16:D1:2B:A8:99
verisignclass1ca, Jun 29, 1998, trustedCertEntry,
Certificate fingerprint (MD5): 51:86:E8:1F:BC:B1:C3:71:B5:18:10:DB:5F:DC:F6:20
thawtepersonalbasicca, Feb 12, 1999, trustedCertEntry,
Certificate fingerprint (MD5): E6:0B:D2:C9:CA:2D:88:DB:1A:71:0E:4B:78:EB:02:41
entrustsslca, Jan 9, 2003, trustedCertEntry,
Certificate fingerprint (MD5): DF:F2:80:73:CC:F1:E6:61:73:FC:F5:42:E9:C5:7C:EE
thawtepersonalfreemailca, Feb 12, 1999, trustedCertEntry,
Certificate fingerprint (MD5): 1E:74:C3:86:3C:0C:35:C5:3E:C2:7F:EF:3C:AA:3C:D9
verisignclass3ca, Oct 24, 2003, trustedCertEntry,
Certificate fingerprint (MD5): 10:FC:63:5D:F6:26:3E:0D:F3:25:BE:5F:79:CD:67:67
gtecybertrustca, May 10, 2002, trustedCertEntry,
Certificate fingerprint (MD5): C4:D7:F0:B2:A3:C5:7D:61:67:F0:04:CD:43:D3:BA:58
thawteserverca, Feb 12, 1999, trustedCertEntry,
Certificate fingerprint (MD5): C5:70:C4:A2:ED:53:78:0C:C8:10:53:81:64:CB:D0:1D
equifaxsecureca, Jul 23, 2003, trustedCertEntry,
Certificate fingerprint (MD5): 67:CB:9D:C0:13:24:8A:82:9B:B2:17:1E:D1:1B:EC:D4
thawtepersonalpremiumca, Feb 12, 1999, trustedCertEntry,
Certificate fingerprint (MD5): 3A:B2:DE:22:9A:20:93:49:F9:ED:C8:D2:8A:E7:68:0D
thawtepremiumserverca, Feb 12, 1999, trustedCertEntry,
Certificate fingerprint (MD5): 06:9F:69:79:16:66:90:02:1B:8C:8C:A2:C3:07:6F:3A
entrust2048ca, Jan 9, 2003, trustedCertEntry,
Certificate fingerprint (MD5): BA:21:EA:20:D6:DD:DB:8F:C1:57:8B:40:AD:A1:FC:FC
verisignserverca, Jun 29, 1998, trustedCertEntry,
Certificate fingerprint (MD5): 74:7B:82:03:43:F0:00:9E:6B:B3:EC:47:BF:85:A5:93
entrustclientca, Jan 9, 2003, trustedCertEntry,
Certificate fingerprint (MD5): 0C:41:2F:13:5B:A0:54:F5:96:66:2D:7E:CD:0E:03:F4
baltimorecybertrustca, May 10, 2002, trustedCertEntry,
Certificate fingerprint (MD5): AC:B6:94:A5:9C:17:E0:D7:91:52:9B:B1:97:06:A6:E4
geotrustglobalca, Jul 23, 2003, trustedCertEntry,
Certificate fingerprint (MD5): F7:75:AB:29:FB:51:4E:B7:77:5E:FF:05:3C:99:8E:F5
gtecybertrust5ca, May 10, 2002, trustedCertEntry,
Certificate fingerprint (MD5): 7D:6C:86:E4:FC:4D:D1:0B:00:BA:22:BB:4E:7C:6A:8E
equifaxsecureglobalebusinessca1, Jul 23, 2003, trustedCertEntry,
Certificate fingerprint (MD5): 8F:5D:77:06:27:C4:98:3C:5B:93:78:E7:D7:7D:9B:CC
baltimorecodesigningca, May 10, 2002, trustedCertEntry,
```

```
Certificate fingerprint (MD5): 90:F5:28:49:56:D1:5D:2C:B0:53:D4:4B:EF:6F:90:22
equifaxsecureebusinessca2, Jul 23, 2003, trustedCertEntry,
Certificate fingerprint (MD5): AA:BF:BF:64:97:DA:98:1D:6F:C6:08:3A:95:70:33:CA
verisignclass2ca, Oct 24, 2003, trustedCertEntry,
Certificate fingerprint (MD5): B3:9C:25:B1:C3:2E:32:53:80:15:30:9D:4D:02:77:3E
```

For clarity, the certificate fingerprints are highlighted in bold. The certificate fingerprint that is required is Certificate fingerprint (MD5.

`B0:F2:33:C8:69:E4:25:A3:CB:59:E8:51:27:EE:5C:52)` is not in the trusted certificates. Therefore, you can import the certificate.

6. Perform the procedure described in the "Trusting the Remote Manager Certificate" section in the installation guide for the application server that you use.

11.2 To Create and Test a Remote Manager IT Resource

To create and test a Remote Manager IT resource, perform the following steps:

Note: If you want to test the Remote Manager IT Resource over a non-SSL connection, then set the `<RMIOverSSL>` property in the following file to `false`:

OIM_HOME\xellerate\config\xlconfig.xml

---

1. In the Design Console, open the Resource Object form.
2. Create a resource object. In this example, the following parameters are set:
   - The name is `MyObj`
   - The option, `Order for User` is enabled
   - The Type is `Application`
   - The following check boxes are available:
     - Allowed Multiple
     - Auto Save
     - Self Request Allowed
     - Allow All
     - Auto Launch
3. Create an IT resource type for the resource object. Open the IT Resource Type Definition form. In this example, the following parameters are set:
   - Server Type: `MyObjServer`.

   Note: While defining the IT Resource Type parameter in the Design Console, you can specify which fields will be encrypted.

   ---

4. Create an IT resource for the Remote Manager. In this example, the following parameters are set:
   - The name of the IT Resource is `remote`.
   - The name of the Type is `Remote Manager`.

Ensure that the IT resource has the proper URL and service name, and that the Remote Manager is installed at the location indicated by the URL.

Note: Check to see if the name itself is not present in the URL. For example, the Remote Manager is composed of the service name and URL, as follows:

service name: RManager url: rmi://w2kevandanwkstn:12346

---

5. Create an instance of the `MyObjServer` IT Resource Type created previously. Open the IT Resource Information Form. In the Remote Manager field, ensure that the Remote Manager created in Step 4 (`remote`) is selected.
6. After saving the information in the IT Resources Information form, you can provide any additional details required for that IT resource. In this example, the user name and password are entered.
7. Create a JAR file for the following code:

```
package testme;
import java.io.PrintStream;
public class test
{
    public test ()
    {
    }
    public static int addme(int i, int j)
    {
     /*6*/System.out.println(i + "+" + j + "=" + (i + j));
     /*7*/return i + j;
     }
     public static void main(String args[])
     {
      /*  11*/addme(5, 10);
     }
}
```

   This code will be run on the Remote Manager.

8. Copy the JAR file into the *`xlremote_home`*`/JavaTasks` and *`OIM_HOME`*`/xellerate/JavaTasks` directories.
9. Create an adapter that will be run in the Remote Manager. Open the Adapter Factory form. In this example, the following parameters are set:
   - The Adapter Name is `remotetest`.
   - The Adapter Type is `Process Task`.

     For this example, you can create three variables for this adapter (based on example code in the .jar file). Click **Add.** The Java code takes two integers as arguments and the IT resource as the third variable.
10. In the first variable, the following parameters are set:
    - The Variable name is `var1`.
    - The Variable type is `Integer`.
    - The Map To option is set to Resolve at Run time.

11. Create the second variable in the same way you did the first. The following parameters are set:
    - The Variable name is `var2`.
    - The Variable type is `Integer`.
    - The Map To option is set to Resolve at Run time.
12. Create the third variable for IT Resource. The parameters are set as follows:
    - The Variable name is `ITRes`.
    - The Variable type is `ITResource`.
    - The Map To option is set to Resolve at Run time.
    - The Resource Type is `MyObjServer`.

    Note: The Resource Type field must be the same "ITResource Type" created in Step 5 and not Remote Manager.

13. Add a New Remote Java Task. In the Adapter Factory Form, click **Add**. Ensure that the Functional Task option is active. Select the **Remote** option. Click **Continue**.
14. The Object Instance Selection dialog box is displayed. Create a new Object Instance. Ensure that the New Object Instance option is active. Click **Continue**.
15. The Remote window is displayed. In this example, the following parameters are set:
    - The Task Name is `remote`.
    - The API Source references the .jar file in the JavaTask folder.
    - The Application API is `Testme.test`.
    - The Constructor is set to `0 public testme.test ()`.
    - The Method is set to `testme.test.addme (int, int)`.

      After clicking **Save**, the IT Resource is automatically added as an argument. The Application Method Parameters are ready for mapping.
16. Begin mapping the parameters by highlighting the first item in the Parameter Data Mapping list. This output parameter is an integer. The following mapping is set:
    - Map To: `Adapter Variables`
    - Name: `Return variable`
17. Click **Set**.
18. Highlight the second parameter to map. This input parameter is an integer. The following mapping is set:
    - Map: `Adapter Variables`
    - Name: `var1`
19. Click **Set**.
20. Select the third parameter to map. This input parameter is an integer. The following mapping is set:
    - Map To: `Adapter Variables`

- Name: `var2`

21. Click **Set**.
22. Select the final parameter to map. Map this ITResource to the variable passed as input to the adapter. The following mapping is set:
    - Map To: `Adapter Variables`
    - Name: `ITRes`
23. Click **Set**.
24. Click **Set**. Then click **Save**. The Adapter Factory form is displayed.
25. Compile the adapter by clicking **Build**.

To invoke the adapter, you can create a provisioning process that calls this adapter as one task. To do this:

1. Open the Process Definition Form. In this example, the following parameters are set:
    - The Name field is `MyObjProv`
    - The Type field is `Provisioning`
    - The Object name is `MyObj`

      The following check boxes are available:
      - Default Process
      - Auto Pre-populate
      - Auto Save Form
2. Click the **Save** icon. The provisioning tasks automatically appear in the Tasks tab.
3. Click **Add** to create a new task. In this example, the parameters are set:
    - The Task Name field is `Call Remote Adapter`.
    - The Task Description field explains the task's function.
4. Click the **Save** icon. Then click the **Integration** tab. Next, click **Add** to add an adapter to this task. The Handler Type window is displayed.
5. Enable the **Adapter** option and select the adapter to be executed.
6. Click the **Save** icon. In the Integration tab, the adapter name appears in the Name field. The Status field shows that the Mapping is incomplete. The Adapter Variables pane shows the variables are not mapped.
7. Select the first variable, Adapter return value, then click **Map**. The Edit Data Mapping for Variable window is displayed. The parameters are set to:
    - Data Type: `Object`
    - Map To: `Response Code`
8. Select the second variable, `var1` then click **Map**. The **Edit Data Mapping for Variable** window appears. The parameters are set to:
    - Data Type: `Integer`
    - Map To: `Literal`
    - Qualifier: `Integer`
    - Literal Value: `10`

9. Select the third variable, var2, then click **Map**. The Edit Data Mapping for Variable window is displayed. The parameters are set to:
   - Data Type: `set to Integer`
   - Map To: `Literal`
   - Qualifier: `Integer`
   - Literal Value: `20`
10. Select the fourth variable, ITRes, and then click **Map**. The Edit Data Mapping for Variable window is displayed. The parameters are set to:
   - Data Type: `IT Resource (MyObjServer)`
   - Map To: `IT Resource`
   - Qualifier: `MyObjServerInstance`
11. Click the **Responses** tab of the Editing Task window. Click **Add** to add the possible responses from the adapter. In this example, the only possible response is 30. Set Description to Completed and Status to C.
12. Click the **Task to Object Status Mapping** tab. In the Completed category, set **Object Status** to Provisioned.
13. At this point, you are ready to directly provision a user with the newly created resource to test the execution of the remote task. However, you must first ensure that the Remote Manager is running. Open the Remote Manager Form and verify that the service is available.
14. Start the Oracle Identity Manager Administrative and User Console and login as Administrator. Navigate to **Users, Manage** and select a user to provision this resource *(MyObj)*. The User Detail page appears with the selected user. In the View Additional Details About This User pull-down option, select **Resource Profile**.
15. The User Detail, Resource Profile page is displayed. Click **Provision New Resource** and select the newly created resource (*MyObj*).
16. The Provision Resource to User wizard is displayed. Click **Continue** to complete the provisioning process.
17. Continue with the provisioning process until you come to the **Resource Successfully Provisioned** page is displayed.
18. Check the Remote Manager log file to see if the code is executed. The Remote Manager log file is located in the *`OIM_HOME`*`/xlremote/log` directory. The last line in the log should be similar to the following:

    ```
    DONE5+10=15
    ```

    The preceding line shows that the two input integers are added to equal 15. This indicates that the code executed correctly and that the resource object was provisioned.

12

SPML Web Service

This chapter discusses the SPML Web Service interface of Oracle Identity Manager.

The following sections of this chapter provide basic information about the SPML Web Service:

- Introduction to the SPML Web Service
- Provisioning Operations Supported by the SPML Web Service

To deploy the SPML Web Service, you can follow the approach described in *one* of the following sections:

- Deploying the SPML Web Service
- Enabling Security by Using Oracle Web Services Manager and Then Deploying the SPML Web Service

The following sections describe procedures to be performed after you deploy the SPML Web Service:

- Postdeployment Tasks
- Enabling SSL Communication

The following section provides an overview of the procedure to develop a client for the SPML Web Service:

- Developing the Client for the SPML Web Service

12.1 Introduction to the SPML Web Service

Organizations can have multiple provisioning systems that exchange information about the modification of user records. In addition, there can be applications that interact with multiple provisioning systems. Connectors can enable the interaction between two provisioning systems or between an application and a provisioning system so that each application synchronizes with the other. However, configuring a custom connector for each combination of these systems leads to a lot of overhead.

The solution to this problem is the application of one common language or protocol that all these systems understand. The answer is SPML.

The SPML Web Service provides a layer over Oracle Identity Manager to interpret SPML requests and convert them to Oracle Identity Manager calls.

> **See Also:** Refer to the following Web page for information about the SPML v2.0 specification:
>
> `http://www.oasis-open.org/specs/index.php#spmlv2.0`

The SPML Web Service supports only inbound provisioning requests. It does not support any type of reconciliation because it does not generate reconciliation events. For example, if you request a resource to be provisioned to an OIM User in which data must be populated in the resource's process form and child table, the request will not be supported. This is because the SPML Web Service is not aware of any information associated with a resource object.

Note: Outbound provisioning requests can be sent by using a generic technology connector containing the SPML Provisioning Format Provider. Refer to *Oracle Identity Manager Administrative and User Console Guide* for information about generic technology connectors.

12.1.1 Functional Architecture of the SPML Web Service

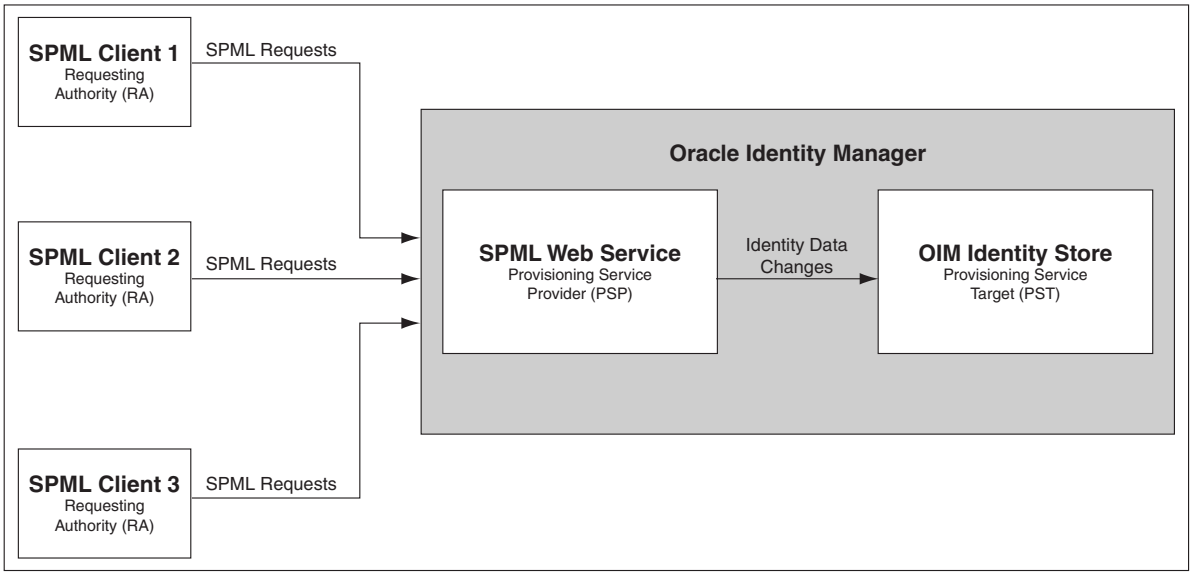
The SPML Web Service sends and receives SPML requests in the form of SOAP messages. The SPML model consists of the following entities that participate in an end-to-end provisioning scenario.

- Requesting Authority (RA): An RA or requestor is the component that issues well-formed SPML requests to a provisioning service provider.
- Provisioning Service Provider (PSP): A PSP or provider is the component that listens for, processes, and returns the results for well-formed SPML requests from a known requestor.
- Provisioning Service Target (PST): A PST or target represents a destination or an endpoint that a provider makes available for provisioning actions.

The implementation of the SPML protocol allows for the reliable exchange of provisioning requests and a model on which you can build a more complex application-level provisioning functionality. SPML is the language of exchanging the management requests used by provisioning systems to manage and control an identity.

Figure 12–1 illustrates the functional architecture of the SPML Web Service.

Figure 12–1 SPML Architecture



The provisioning application can play the role of both an RA and a PSP. Consider the following scenarios:

Provisioning Application as PSP

In this scenario, a client application sends an SPML request to the provisioning application. The provisioning application carried out the request and returns an SPML response to the client application. The request-response exchange is either synchronous or asynchronous. This is typically described as the "inbound" scenario. In Oracle Identity Manager, this is implemented through the SPML Web Service.

Provisioning Application as RA

In this scenario, the provisioning application plays the role of the SPML client and sends an SPML request to a PST, which carries the request and returns an SPML response. The request-response exchange is synchronous or asynchronous. This is typically described as the "outbound" scenario. In Oracle Identity Manager, this is implemented through the generic technology connector containing the SPML Format Provisioning Provider.

Provisioning Application as RA and PSP

Note: This feature is not supported in Oracle Identity Manager release 9.1.0.

In this scenario, a client application sends an SPML request to the provisioning application that cannot itself fulfill the request. Here, the provisioning application forwards the request to the provisioning target that fulfills the request and returns an SPML response. The provisioning application then returns an SPML response to the client application. The request-request-response-response exchange is synchronous or asynchronous.

12.2 Provisioning Operations Supported by the SPML Web Service

The SPML Web Service supports capabilities that meet the minimum conformance criteria described in the SPML v2.0 specification. This section discusses the various operations supported by the SPML Web Service.

Note: The SPML Web Service supports requests in UTF-8 encoding only.

You can use the psoID to uniquely identify an entity (User/Group/Organization) in a provisioning target. In Oracle Identity Manager, to specify an entity, the combination of an objectclass and an entity key is required. An entity key is the database key returned when you create an entity in Oracle Identity Manager. The psoID in Oracle Identity Manager is in the following format:

`objectclass:`*`entitykey`*

For example, objectclass can be Users and entity key can be 3. In this case, the psoID you specify is `Users:3`.

The SPML Web Service supports the following provisioning operations or requests:

- **Add Operations**

The Add Request operation creates an entry for the requested target type in Oracle Identity Manager. The supported target types are User, Group, and Organization. This operation checks if an entry exists and reports errors, if any. This operation can also include capability data for the User and Group target types when it associates a user or group with one or more groups by using the `memberOf` relationship. This function also supports the `administrator` reference capability, which you can use to assign one or more groups as the administrator of a new group.

> **See Also:** The "Add Request" section on page B-1 for a sample Add request

- **Modify Operations**

  The Modify Request operation updates the specified target entry for the requested target type in Oracle Identity Manager. The supported target types are User, Group, and Organization. This operation checks if an entry exists and reports errors, if any. If the entry exists, then the target is updated with the given data. This operation can also modify capability data. For the User target type, the supported reference capability is `memberOf`. This operation can also be used to update the references of the user. The user can be added to or removed from a group.

  Similarly, for the Group target type, the supported reference capabilities are `memberOf` and `administrator`. Modify requests can be used to create, replace, or delete these references.

  > **See Also:** The "Modify Request" section on page B-4 for a sample Modify request

- **Delete Operations**

  The Delete Request operation deletes the specified target entry from Oracle Identity Manager. The supported target types are User, Group, and Organization. This operation checks if an entry exists and reports errors, if any. If the entry exists, it is deleted. The recursive delete option is not supported.

- **Add, Replace, or Delete References**

  References are entities that can be referred to by other entities. For example, user John Doe is a member of a group. Now, you want to remove John Doe from that group and make him a member of another group. You can specify the details of the user in the new group through a reference, and the user becomes the member of the new group.

  The following references are supported by SPML:

  – Add/Replace/Delete Group membership references for users

  – Add/Replace/Delete Group membership references for groups

  – Add/Replace/Delete Group administrator references for groups

- **Lookup Operations**

  The Lookup Request operation looks up the specified target entry in Oracle Identity Manager. The supported target types are User, Group, and Organization. This operation checks if the entry exists, and it returns the lookup data or reports back errors. The user can be looked up in Oracle Identity Manager based on the user ID. The lookup operation also returns any *capability-specific* data that is

associated with the object if you specify everything as the value of `returnData` type.

- **Search Operations**

  The Search Request operation searches for the specified target entry in Oracle Identity Manager. The supported target types are User, Group, and Organization. This operation checks if the entry exists and returns the search response based on whether or not the entry is available.

  Search requests do not support query scope. Even if there is a query scope, it is ignored. The search query is formed by applying the AND operator to the search criteria specified in the request. Different search criteria can be specified in an SPML search request by using the Directory Services Markup Language (DSML) filter. The only operation supported in the DSML filter is `equalityMatch`.

  The `includeDataForCapability` element is not supported. If you specify everything as the value of `returnData`, then the search operation returns all the associated references. The search operation requires object class information based on which it can query the User, Group, and Organization containers. Therefore, it expects an object class as one of the query elements in the search query.

  For example, if you want to perform a search on the Users object class, then this information must be embedded in the request. This is described in the following code sample:

```
<searchRequest
returnData="identifier"xmlns="urn:oasis:names:tc:SPML:2:0:search"
xmlns:dsml="urn:oasis:names:tc:DSML:2:0:core">
 <query scope="pso">
  <basePsoID ID=""/><and>
  <dsml:filter><dsml:equalityMatch name="Users.LastName">
  <dsml:values>Doe
  </dsml:values>
  </dsml:equalityMatch>
  </dsml:filter>
  <dsml:filter><dsml:equalityMatch name="Users.XellerateType">
  <dsml:values>End-User
  </dsml:values>
  </dsml:equalityMatch>
  </dsml:filter>
  <dsml:filter>
  <dsml:equalityMatch name="ObjectClass"><dsml:values>Users
  </dsml:values>
  </dsml:equalityMatch>
  </dsml:filter>
  </and>
  </query>
 </searchRequest>
```

The `basePsoID` in which the search is performed must be `Organization` or an empty string. (The `basepsoID` is either the same as the psoID format `Organization:key` or it can be an empty string.)

For example: `<basePsoID ID="Organization:7"/>`, `<basePsoID ID=""/>`

- **Password Operations**

  There are two `passwordRequest` operations, `setPasswordRequest` and `resetPasswordRequest`. The former enables a user to change the password while the latter generates a new, randomly generated password for the user, based on the password policy defined for that user in Oracle Identity Manager. The

supported target type is User. By default, `resetPasswordRequest` sets the password to `default`.

- **Suspend, Resume, or Active User Operations**

  The `suspend` capability specified in the SPML v2.0 specification has three operations:

  - `suspend`: Disables the state of an active user in Oracle Identity Manager.
  - `resume`: Enables the state of a disabled user in Oracle Identity Manager.
  - `active`: Returns the state of a user (that is, specifies whether or not a user is in the `active` state).

- **ListTargets Operations**

  - The `listTargets` operation can be used to determine the targets that are available for provisioning. This operation can also be used to determine the attributes that are supported by the provisioning system for each supported target type. This operation returns the schema information of all three supported target types: User, Group, and Organization.

The SPML Web Service does *not* support the following operations:

- Search (iterate/closelterator/hasReference)
- Update (updates/iterate/closelterator)
- Asynchronous request (status/cancel)
- Batch processing
- Bulk update (bulkModify/bulkDelete)
- Password expiry or validation

Note: If you want to include the date attribute in a provisioning request, then you must use the following format:

```
yyyy-MM-dd hh:mm:ss.fffffffff
```

No other date format is supported. Refer to the "Add Request With Date Format" section on page B-8 for a sample Add request with the date attribute assigned.

12.3 Deploying the SPML Web Service

Note: If you are using SPML Web service along with Oracle Identity Manager, then you must redeploy the SPML Web service whenever you upgrade Oracle Identity Manager.

If you have customized the EAR file, then you must redo those changes in the EAR file and then redeploy it.

The SPML Web Service is packaged in a deployable Enterprise Archive (EAR) file named `OIMSpmlWS.ear`. This file is generated when you install Oracle Identity Manager. This file is stored at the following location:

OIM_HOME/SPMLWS

There is a separate EAR file for each application server, and each file is stored in its respective application server folder in the `SPMLWS` directory. This EAR file is generated when you install Oracle Identity Manager.

> **Note:** Oracle Identity Manager and the SPML Web Service must be deployed on the same application server. This is known as **collocated deployment**. In a clustered environment, ensure that the SPML Web Service is installed on each node on which Oracle Identity Manager is installed.

Use the following batch file to run the scripts that deploy the SPML Web Service on the application server on which Oracle Identity Manager is running:

OIM_HOME/xellerate/setup/spml_*AppServerName*

To run this batch file, perform the steps that correspond to your operating environment:

> **Note:** The following log file is created when you run the batch file:
>
> *OIM_HOME*/xellerate/logs/spml-*AppServerName*.log

- Deploying the SPML Web Service on Oracle WebLogic Server
- Deploying the SPML Web Service on IBM WebSphere Application Server
- Deploying the SPML Web Service on JBoss Application Server
- Deploying the SPML Web Service on Oracle Application Server

12.3.1 Deploying the SPML Web Service on Oracle WebLogic Server

On a nonclustered Oracle WebLogic Server installation:

Enter the following command:

For UNIX:

OIM_HOME/setup/spml_*weblogic*.sh *appserver_admin_password oim_db_user_password*

For Microsoft Windows:

OIM_HOME\setup\spml_*weblogic*.cmd *appserver_admin_password oim_db_user_password*

On a clustered Oracle WebLogic Server installation:

1. Enter the following command on the administrator node:

   For UNIX:

   OIM_HOME/xellerate/setup/spml_*weblogic*.sh *appserver_admin_password oim_db_user_password*

   For Microsoft Windows:

   OIM_HOME\xellerate\setup\spml_*weblogic*.cmd *appserver_admin_password oim_db_user_password*

2. Perform the procedure described in the "Configuring the Apache Proxy Plug-in" appendix of *Oracle Identity Manager Installation and Configuration Guide for Oracle WebLogic Server*.

12.3.2 Deploying the SPML Web Service on IBM WebSphere Application Server

On a nonclustered IBM WebSphere Application Server installation:

Enter the following command:

For UNIX:

```
OIM_HOME/setup/spml_websphere.sh appserver_admin_password oim_db_user_password
```

For Microsoft Windows:

```
OIM_HOME\setup\spml_websphere.cmd appserver_admin_password oim_db_user_password
```

On a clustered IBM WebSphere Application Server installation:

1. Enter the following command on the administrator node:

   For UNIX:

   ```
   OIM_HOME/xellerate/setup/spml_websphere.sh appserver_admin_password
   oim_db_user_password
   ```

   For Microsoft Windows:

   ```
   OIM_HOME\xellerate\setup\spml_websphere.cmd appserver_admin_password
   oim_db_user_password
   ```

2. Regenerate the `plugin-cfg.xml` file by performing the procedure described in the "Configuring the IIS Plug-in" section of *Oracle Identity Manager Installation and Configuration Guide for IBM WebSphere Application Server*.

12.3.3 Deploying the SPML Web Service on JBoss Application Server

Note: Deployment of the SPML Web Service in clustered JBoss Application Server environments is not supported.

On a nonclustered JBoss Application Server installation:

Enter the following command:

For UNIX:

```
OIM_HOME/xellerate/setup/spml_jboss.sh oim_db_user_password
```

For Microsoft Windows:

```
OIM_HOME\xellerate\setup\spml_jboss.cmd oim_db_user_password
```

12.3.4 Deploying the SPML Web Service on Oracle Application Server

On a nonclustered Oracle Application Server installation:

1. Enter the following command:

   For UNIX:

OIM_HOME/setup/spml_*oc4j*.sh *appserver_admin_password oim_db_user_password*

For Microsoft Windows:

OIM_HOME\setup\spml_*oc4j*.cmd *appserver_admin_password oim_db_user_password*

2. Enter the following command script:

   For UNIX:

   OIM_HOME\xellerate\setup\spml_oc4j.sh *appserver_admin_password oim_db_user_password*

   For Microsoft Windows:

   OIM_HOME\xellerate\setup\spml_oc4j.cmd *appserver_admin_password oim_db_user_password*

3. Open the following file in a text editor:

   OC4J_HOME/j2ee/*OC4J_instance*/config/application.xml

4. In the `<imported-shared-libraries>` section of the `application.xml` file, change `<import-shared-library name="apache.commons.logging"/>` to `<remove-inherited name="apache.commons.logging"/>`.

   In other words, the `<imported-shared-libraries>` section must appear as follows

```
<imported-shared-libraries>
  <import-shared-library name="adf.oracle.domain"/>
  <import-shared-library name="oracle.ifs.client"/>
  <remove-inherited name="apache.commons.logging"/>
</imported-shared-libraries>
```

On a clustered Oracle Application Server installation:

Perform the following steps on each node of the cluster:

1. Enter the following command script:

   For UNIX:

   OIM_HOME\xellerate\setup\spml_oc4j.sh *appserver_admin_password oim_db_user_password*

   For Microsoft Windows:

   OIM_HOME\xellerate\setup\spml_oc4j.cmd *appserver_admin_password oim_db_user_password*

2. Open the following file in a text editor:

   OC4J_HOME/j2ee/*OC4J_instance*/config/application.xml

3. In the `<imported-shared-libraries>` section of the `application.xml` file, change `<import-shared-library name="apache.commons.logging"/>` to `<remove-inherited name="apache.commons.logging"/>`.

   In other words, the `<imported-shared-libraries>` section must appear as follows

```
<imported-shared-libraries>
  <import-shared-library name="adf.oracle.domain"/>
  <import-shared-library name="oracle.ifs.client"/>
```

```
    <remove-inherited name="apache.commons.logging"/>
</imported-shared-libraries>
```

12.4 Enabling Security by Using Oracle Web Services Manager and Then Deploying the SPML Web Service

Note: Perform the procedure described in this section only if you want to secure the SPML Web Service by using Oracle Web Services Manager.

Oracle Web Services Manager (WSM) provides features that ease the installation, configuration, and management of Web services across a wide range of deployment environments.

See Also: *Oracle Web Services Manager Administrator's Guide* for detailed information about Oracle WSM

You use Oracle WSM to secure the SPML Web Service. When a request is sent to the SPML Web Service in a SOAP message, it is intercepted by Oracle WSM. The SOAP message contains the Web Services Security (wsse) tag in the SOAP header. This `wsse:security` tag contains the user credentials that must be authenticated. For securing the SPML Web Service, you can use either the Oracle WSM Server Agent or the Oracle WSM Gateway.

See Also:

- *Oracle Web Services Manager Installation Guide* for detailed information about installing Oracle WSM

- *Oracle Web Services Manager Deployment Guide* for detailed information about the Oracle WSM Server Agent and the Oracle WSM Gateway

The Oracle WSM Server Agent or the Oracle WSM Gateway strips off the `wsse:security` tag from the SOAP message before forwarding the request to the SPML Web Service. You must configure the Oracle WSM Server Agent or the Oracle WSM Gateway to use a custom policy step, which extracts the user credentials from the `wsse:security` tag and inserts them into a custom header tag of the SOAP header from where the SPML Web Service extracts the credentials.

The following sections describe the configuration steps that you must perform to secure the SPML Web Service by using the Oracle WSM Server Agent or the Oracle WSM Gateway.

- Configuring the Oracle WSM Server Agent
- Configuring the Oracle WSM Gateway

Note: The Oracle WSM Server Agent supports IBM WebSphere Application Server and Oracle Application Server. You can use the Oracle WSM Gateway for all the application servers.

12.4.1 Configuring the Oracle WSM Server Agent

The following steps are required to configure the Oracle WSM Server Agent for securing the SPML Web Service:

> **Note:** You configure the Oracle WSM Server Agent before deploying the SPML Web Service.

1. Adding a Server Agent
2. Defining a Policy for the Server Agent
3. Injecting the Server Agent
4. Deploying the SPML Web Service

12.4.1.1 Adding a Server Agent

To add a Server Agent:

> **See Also:** Chapter 6, "Installing Oracle WSM Agents" in *Oracle Web Services Manager Deployment Guide* for detailed information about adding a server agent

1. Use the Web Services Manager Control (for example, http://localhost:8888/ccore) to create a server agent , and select the following values from the list:
   - Component type: **Server Agent**
   - Container type: Select **OC4J** for Oracle Application Server. Select **OTHER** for IBM WebSphere Application Server.
2. Select **Register**. This generates a server agent component with a component ID.

12.4.1.2 Defining a Policy for the Server Agent

Use the Web Services Manager Control to define the policy that you want to associate with the server agent. The default implementation of the policy is provided at the following location:

OIM_HOME/SPMLWS/OWSMPolicy

> **See Also:** Chapter 5, "Oracle Web Services Manager Policy Management," in the *Oracle Web Services Manager Administrator's Guide* for detailed information about defining a policy for a server agent

You must associate a URL pattern with the policy. To do so, select **Policy Management**, **Manage Policies**, and then **Policies**. Then click the **Edit Mapping** button and enter the following as the URL pattern to associate with the policy:

```
/spmlws/HttpSoap11
```

To configure a custom policy, you must include the class file `com.oracle.xl.spmlws.ws.security.owsm.CustomPolicyStep` in *OIM_HOME*/SPMLWS/OWSMPolicy into the following file:

ORACLE_HOME/owsm/lib/extlib/coresvagent.jar

After creating the server agent component, you must add your custom step to that component. You can do this by clicking the **Steps** link for your registered component. Then, select the `CustomPolicyStep.xml` file from its location and click **Upload**. This XML file is located at *OIM_HOME*`/SPMLWS/OWSMPolicy/com/oracle/xl/spmlws/ws/security/owsm`.

At this stage, your custom policy step name is added to the list of available policies.

> **Note:** If you create a custom policy, then its class file must be included in the `coresvagent.jar` file that resides in the Web services EAR file.

12.4.1.3 Injecting the Server Agent

Injecting the server agent requires you to perform the following steps:

For Oracle Application Server

To inject the server agent:

1. Modify the attributes in the `ORACLE_HOME/owsm/bin/agent.properties` file with the following values:
   - agent.componentType: ServerAgent
   - agent.containerType: OC4J
   - agent.containerVersion: It must be "10.1.3" for Oracle Application Server.
   - agent.component.id: Enter the component ID that is generated when the agent is created and registered by using Web Services Manager Control.
2. Edit the following properties in the `agent.properties` file:
   - `webservice.application.input` - Enter the full path and name of the EAR file.
   - `webservice.application.webapp.name` - Uncomment and enter the WAR file name, `spmlws.war`.

     > **Note:** the WAR file is bundled in the `OIMSpmlWS.ear` file.

   - `webservice.application.contexturi` - Enter the context root, `/spmlws`.
3. Run the `wsmadmin installAgent` command.

For IBM WebSphere Application Server

To inject the server agent:

1. Because the Server Agent for SOA 10.1.3.1 release is supported only for Oracle Application Server, for WebSphere, you must first download the required ZIP file from the following location on Oracle Technology Network:

   For UNIX:

   `http://download.oracle.com/otn/linux/ias/101310/soa_linux_x86_ws_agent101310.zip`

For Microsoft Windows:

`http://download.oracle.com/otn/nt/ias/101310/soa_windows_x86_ws_agent101310.zip`

2. Extract the contents of the ZIP file to any location (for example `/owsm`).

   Tip: The Readme_Agentinstall.pdf file in the extracted ZIP file for more information about injecting the server agent

3. Browse to the `bin` directory, open the `agent.properties` file, and set the following properties in the file:
   - agent.componentType: serveragent
   - agent.containerType: For example, AXIS, WEBLOGIC, WEBSPHERE, TIBCO-BW, or OC4J
   - agent.containerVersion: The version of WebSphere on which you are deploying the SPML Web Service.
   - oc4j.home:`/owsm/oc4j` (assuming that `/owsm` is where you extracted the ZIP file)
   - oc4j.j2ee.home:`/owsm/oc4j/j2ee/home` (here `/owsm` is where you extracted the ZIP file)
   - webservice.application.input: Web application input file name with path location, that is, WAR or EAR file location. For example, `/owsm/wars/HelloWorldImpl.war`
   - webservice.application.webapp.name: not applicable if it is a WAR file
   - webservice.application.contexturi: not applicable if it is a WAR file
   - agent.component.id: Enter the component ID that is generated when the agent is created and registered by using Oracle Web Services Manager Control.
   - agent.policymanagerURL (for example, *`http://hostname:8888/policymanager`*. Provide a system name instead of localhost)
4. Open the `bin/coresv.properties` file and set the following properties:
   - coresv.home=`/owsm` (assuming `/owsm` is where you extracted the ZIP file)
   - ant.home: set home directory of the ANT installation
   - java.home: set home directory of the Java installation
   - lib.dir:`/owsm/lib` (assuming `/owsm` is where you extracted the ZIP file)
   - oc4j.j2ee.home:`/owsm/oc4j/j2ee/home` (optional if the properties are present)
   - external.oc4j.home:`/owsm/oc4j` (optional if the properties are present)
5. For configuring a custom policy, you must include the class file `com.oracle.xl.spmlws.ws.security.owsm.CustomPolicyStep` in *`OIM_HOME`*`/SPMLWS/OWSMPolicy` into the following file:

   `/owsm/lib/extlib/coresvagent.jar`

   (assuming `/owsm` is where you extracted the ZIP file)

6. In a command window, navigate to the `bin` directory. Run the `injectAgent` command. This command injects all the JAR files into the specified WAR or EAR file. (Before running this, set the path to the `ant bin` directory.)

12.4.1.4 Deploying the SPML Web Service

After you have installed the client agent, deploy the SPML Web Service. For information about how to deploy the SPML Web Service, refer to the "Deploying the SPML Web Service" section on page 12-6.

12.4.2 Configuring the Oracle WSM Gateway

The following steps are required to deploy the Oracle WSM Gateway for securing the SPML Web Service:

> **See Also:** *Oracle Web Services Manager Quick Start Guide* for detailed information about specific configurations of Oracle WSM

1. Registering the Oracle WSM Gateway
2. Registering the SPML Web Service with the Gateway
3. Adding a Custom Policy to the Gateway
4. Deploying the SPML Web Service
5. Viewing the WSDL File

12.4.2.1 Registering the Oracle WSM Gateway

To register the Gateway:

1. Click **Add New Component** in the Oracle Web Services Manager Control.
2. On the Add New Component page, enter the following values:
   - Component Name: for example, MyGateway
   - Component Type: Gateway (default value)
   - Container Type: Oracle Web Services Manager (default value)
   - Component URL: Enter the following: `http://fully_qualified_host_name:http_port/gateway` where fully_qualified_host_name is the URL for Oracle WSM, and http_port is the port on which Oracle WSM is hosted
   - Component Groups: accept the default values for the component groups
3. Click **Register**.
4. Click **OK**.

12.4.2.2 Registering the SPML Web Service with the Gateway

To register the SPML Web Service with the Gateway:

1. From the navigation pane of Oracle Web Services Manager Control, click **Policy Management**.
2. Click **Register Services**.
3. Click the **Services** link.

4. Click **Add New Service**. The Add New Service page is displayed. On this page, enter the following service details:
   - Service Name: SPMLService
   - Service Version: 1.0
   - Service Description: Processes SPML Requests
   - WSDL URL: The WSDL location, for example:

     `http://host:port/spmlws/.../HttpSoap11?wsdl`

5. Click **Next**. The Configure Messenger Step for New Service page is displayed. On this page, verify that the URL matches the URL you provided on the previous page. Click **Finish** to accept the default values for the remaining fields.
6. Click **Commit Policy**.

12.4.2.3 Adding a Custom Policy to the Gateway

Use the Oracle Web Services Manager Control to configure a policy that you want to associate with the Gateway. The default implementation of the policy is at the following location:

OIM_HOME/SPMLWS/OWSMPolicy

> **See Also:** Chapter 5, "Oracle Web Services Manager Policy Management," in *Oracle Web Services Manager Administrator's Guide* for more information about defining a policy for a Gateway

The policy extracts user credentials from the WSSE security tags and adds them to the SOAP header in the following custom tag.

```
<wsa1:OIMUser soap:mustUnderstand="0"
xmlns:wsa1="http://xmlns.oracle.com/OIM/provisioning">
<wsa1:OIMUserId
xlmns:wsa1="http://xmlns.oracle.com/OIM/provisioning">user1</wsa1:OIMUserId>
<wsa1:OIMUserPassword
xlmns:wsa1="http://xmlns.oracle.com/OIM/provisioning">password1</wsa1:OIMUserPassw
ord>
</wsa1:OIMUser>
```

The SPML Web Service interprets and processes this tag.

For configuring a custom policy, you must include the `com.oracle.xl.spmlws.ws.security.owsm.CustomPolicyStep` in *OIM_HOME*/SPMLWS/OWSMPolicy class file into the following file:

ORACLE_HOME/owsm/lib/coresv-4.0.jar

You must include the custom policy step file because the `coresv-4.0.jar` file contains all the policy-specific class files. Then, restart Oracle Application Server.

After you register the Gateway, you must add a custom step. To add a custom step:

1. Upload the `CustomPolicy.xml` file for the custom policy on the Add Step page. This XML file is located at the following path:

   OIM_HOME/SPMLWS/OWSMPolicy/com/oracle/xl/spmlws/ws/security/owsm

2. To open the Add Step page in the Oracle Web Services Manager Control, expand **Policy Management**, click **Manage Policies**, **Steps**, and then **Add Step**. Select the

`CustomPolicy.xml` file from its location and then click **Upload**. The name of the custom policy step is added to the list of available policies.

3. To add the custom policy to the pipeline in the Request block, browse to the Policy page. To open the Policy page, expand **Policy Management**, click **Manage Policies**, **Policies**, and finally click **Policy**.

   The Request block will enforce this configured custom policy for any valid incoming request sent to the SPML Web Service.

12.4.2.4 Deploying the SPML Web Service

After you have installed the client agent, deploy the SPML Web Service. For information about how to deploy the SPML Web Service, refer to the "Deploying the SPML Web Service" section on page 12-6.

12.4.2.5 Viewing the WSDL File

To view the Web Service Description Language (WSDL) file for the Web Service:

1. From the navigation pane of Web Services Manager Control, click **Policy Management**.
2. Click **Register Services**.
3. To access the Gateway (MyGateway), click **Services**.
4. From the list of services, click **Edit** for the required service.
5. In the Edit Service page, copy the URL displayed in the Service WSDL URL field.

You use this URL in the SPML client to access the SPML Web Service.

12.5 Postdeployment Tasks

If you are using JBoss Application Server, Oracle WebLogic Server, or IBM WebSphere Application Server, then there are no postdeployment steps to perform.

If you are using IBM WebSphere Application Server 6.1, extract the `xlDataObjectBeans.jar` file, and copy it into the `WEB-INF/lib` directory of the SPML Web Service WAR file. You must restart WebSphere after you copy this file.

> **See Also:** "Extracting xlDataObjectBeans.jar" in *Oracle Identity Manager Installation and Configuration Guide for IBM WebSphere Application Server*

12.6 Enabling SSL Communication

This section provides information about enabling Secure Sockets Layer (SSL) communication for the SPML Web Service. It is strongly recommended that you perform the instructions given in this section.

Note: Oracle recommends that you refer to application server-specific SSL configuration documentation for details. This section provides the minimum information required to enable SSL communication for the different application servers on which the SPML Web Service is supported.

Although this section provides information for specific releases of the application servers, if you are using a different release, then some steps of the procedure can vary.

12.6.1 JBoss Application Server

The following sections provide information required to enable SSL communication for the SPML Web Service installed on JBoss Application Server 4.2.3 GA.

12.6.1.1 Prerequisites

The following are the prerequisites for enabling SSL communication:

- JBoss Application Server is installed and Oracle Identity Manager and the SPML Web Service are deployed on it.
- The JBoss Application Server home directory is E:\jboss-4.2.3.GA.
- The identity store is `jbossserver.jks` and the password is `welcome`.
- Certificate request is made for localhost.
- The self-sign certificate is named `jbossserver.cert`.
- The private key alias is `serverjboss` and the password is `welcome`.

12.6.1.2 SSL Certificate Setup

This section discusses the following procedures for setting up SSL:

Tip: For enhanced protection, Oracle recommends that you create new certificates (either self-signed or CA certificates) and create a separate keystore and truststore for the client and the server with different passwords. Refer to the SSL configuration documentation of the application server for detail.

1. Generating Keys
2. Signing the Certificates
3. Exporting the Certificate
4. Configuring the server.xml File

Generating Keys

Generate keys by using the `keytool` command. The following keytool command generates an identity keystore `jbossserver.jks`:

```
keytool -genkey -alias serverjboss -keyalg RSA -keysize 1024 -dname
"CN=localhost,OU=Identity,O=Oracle,C=US" -keypass welcome -keystore
E:\jboss-4.2.3.GA\server\jbossserver.jks -storepass welcome -storetype jks
```

Signing the Certificates

Use the following `keytool` command to sign the certificates that were created:

```
keytool -selfcert -alias serverjboss -sigalg MD5withRSA -validity 2000 -keypass
welcome -keystore  E:\jboss-4.2.3.GA\server\jbossserver.jks -storepass welcome
```

Exporting the Certificate

Use the following `keytool` command to export the certificate from the identity keystore to a file (for example, `jbossserver.cert`):

```
keytool -export -alias serverjboss -file E:\jboss-4.2.3.GA\server\jbossserver.cert
-keypass welcome -keystore E:\jboss-4.2.3.GA\server\jbossserver.jks -storepass
welcome -storetype jks -provider sun.security.provider.Sun
```

Configuring the server.xml File

Make the following entry in the `server.xml` file:

```
<Connector port="8443" address="${jboss.bind.address}"
           maxThreads="100" strategy="ms" maxHttpHeaderSize="8192"
           emptySessionPath="true"
           scheme="https" secure="true" clientAuth="false"
              sslProtocol="TLS"
           keystoreFile="E:\jboss-4.2.3.GA\server\jbossserver.jks"
           keystorePass="welcome"
           truststoreFile="E:\jboss-4.2.3.GA\server\jbossserver.jks"
           truststorePass="welcome"/>
```

After you have performed the preceding steps, restart the server for the changes to take effect.

Note: You can use the certificate exported in the "Exporting the Certificate" step to import into the client-side truststore for SSL communication.

12.6.2 Oracle WebLogic Server

The following sections provide information required to enable SSL communication for the SPML Web Service installed on Oracle WebLogic Server.

12.6.2.1 Prerequisites

The following are the prerequisites for enabling SSL communication:

- Oracle WebLogic Server is installed.
- The WebLogic Domain directory is `C:\bea\user_projects\domains\oim`.
- The Oracle WebLogic Server home (*WL_HOME*) directory is `C:\bea\wlserver_10.3`.
- The identity store is `support.jks` and the password is `support`.
- The certificate request is made for xellerate.oracle.com host and for Oracle Identity Management Group.
- The self-sign certificate is named `supportcert.pem`.
- The private key alias is `support`, and the password is `weblogic`.

- The `setEnv.cmd` or `setEnv.sh` script is run to set up PATH, CLASSPATH, and other variables.

12.6.2.2 SSL Certificate Setup

This section discusses the following procedures for setting up SSL:

> **Tip:** For enhanced protection, Oracle recommends that you create new certificates (either self-signed or CA certificates) and create a separate keystore and truststore for the client and the server with different passwords. Refer to the SSL configuration documentation of the application server for detail.

1. Generating Keys
2. Signing the Certificates
3. Exporting the Certificate
4. Configuring the Oracle WebLogic Server

Generating Keys

Generate private/public certificate pairs by using the keytool command provided. The following command creates an identity keystore (`support.jks`). Change the parameter values passed to the keytool command according to your requirements. Ensure that there is no line break in the keytool argument.

```
keytool -genkey
        -alias support
        -keyalg RSA
        -keysize 1024
        -dname "CN=xellerate.oracle.com, OU=Identity, O=Oracle Corporation,
L=RedwoodShores, S=California, C=US"
        -keypass weblogic
        -keystore C:\bea\user_projects\domains\oim\support.jks
        -storepass support
```

> **Note:** Use the same host name that you would use in the `xlconfig.xml` file. For example, if you use `https://xellerate.oracle.com:7002` and `t3s://xellerate.oracle.com:7002` in the `xlconfig.xml` file, then the value of CN in the keytool command must be `xellerate.oracle.com`. Oracle recommends that you generate an SSL certificate by using the domain name (for example, `xellerate.oracle.com`) instead of the IP address.

Signing the Certificates

Use the following command to sign the certificates that you created.

```
keytool -selfcert -alias support
        -sigalg MD5withRSA
        -validity 2000
        -keypass weblogic
        -keystore C:\bea\user_projects\domains\oim\support.jks
        -storepass support
```

Note: Oracle recommends that you use trusted certificate authorities, for example, VeriSign or Thawte, for signing the certificates.

Exporting the Certificate

Use the following command to export the certificate from the identity keystore to a file, for example, `supportcert.pem`:

```
keytool -export -alias support
        -file C:\bea\user_projects\domains\oim\supportcert.pem
        -keypass weblogic
        -keystore C:\bea\user_projects\domains\oim\support.jks
        -storepass support
```

Configuring the Oracle WebLogic Server

To configure the Oracle WebLogic Server:

1. In the WebLogic Server Administration Console, click **Environment**, **Servers**, ***Server_Name***, **Configuration**, and then **General**.
2. Click **Lock & Edit**.
3. Select **SSL listen port enabled**. The default port is 7002.
4. Click the **Keystores** tab
5. From the **Keystore** list, select **Custom Identity and Java Standard Trust**.
6. In the **Custom Identity Keystore** field, specify `C:\bea\user_projects\domains\oim\support.jks` as the custom identity keystore file name.
7. Specify **JKS** as the custom identity keystore type.
8. Enter the password in the **Custom Identity Keystore Passphrase** and **Confirm Custom Identity Keystore Passphrase** fields.
9. Click **Save**.
10. Click the **SSL** tab.
11. Enter `support` as the private key alias.
12. Enter the password (for example, support) in the **Private Key Passphrase** and **Confirm Private Key Passphrase** fields.
13. Click **Save**.
14. Click **Activate changes**.
15. Restart the server for the changes to take effect.

Note: You can import the certificate exported in the "Exporting Certificate" step into the client-side truststore for SSL communication.

Import the certificate into the SPML client truststore by using the following keytool command:

```
keytool -import -alias serverwl -trustcacerts -file
D:\bea\user_projects\domains\ mydomain\wlservercert.pem -keystore
<client-trust store> -storepass <client-trust-store password>
```

12.6.3 IBM WebSphere Application Server

The following sections provide information required to enable SSL communication for the SPML Web Service installed on IBM WebSphere Application Server.

12.6.3.1 Prerequisites

The following are the prerequisites for enabling SSL communication:

- IBM WebSphere Application Server is installed and Oracle Identity Manager and the SPML Web Service are deployed on it.
- After configuring IBM WebSphere Application Server and deploying the SPML Web Service with Oracle Identity Manager, you can access the application by using SSL and non-SSL ports.
- To access the application securely by using SSL, you use port number 9443 or WC_defaulthost_secure. Consider the following example:

  ```
  https://localhost:9443/spmlws/HttpSoap11
  ```

- The default identity store is `key.p12`, and the password is `WebAS`.
- The default truststore is `trust.p12`, and the password is `WebAS`.

> **Note:** For SSL communication, export the default certificate from `key.p12`.

12.6.3.2 SSL Certificate Setup

The steps in this section enable you to do the following:

- Exporting Certificate to a File
- Importing the Certificate File

Exporting Certificate to a File

IBM WebSphere Application Server uses the IBM WebSphere default keystore (`key.p12`) and its default certificate. You must export this default certificate to a file. You can use the following keytool command to achieve this:

```
IBM_JDK_HOME/jre/bin/keytool -export -alias default -file <Exported Certificate
file> -keypass WebAS -keystore  FULL_PATH_OF_IBM_WEBSPHERE "key.p12" -storepass
WebAS -storetype pkcs12 -provider com.ibm.crypto.provider.IBMJCE
```

In the preceding command, replace the following to point to the appropriate location:

- The full path of IBM WebSphere Application Server `key.p12`, which is the default IBM keystore
- *IBM_JDK_HOME* to the IBM WebSphere Application Server Java folder
- Location for the exported certificate

Importing the Certificate File

Use the following keytool command to import the certificate file to the SPML Web Service client truststore:

```
keytool -import -alias serverws -trustcacerts -file <Exported Certificate file>
-keystore E:\SPMLTest\mykeystore -storepass mypass  -storetype jks
```

Tip: For enhanced protection, Oracle recommends that you create new certificates (either self-signed or CA certificates) and create a separate keystore and truststore for the client and the server with different passwords. Refer to the SSL configuration documentation of the application server for detail.

12.6.4 Oracle Application Server

The following sections provide information required to enable SSL communication for the SPML Web Service installed on Oracle Application Server.

12.6.4.1 Prerequisites

The following are the prerequisites for enabling SSL communication:

- Oracle Application Server is installed and Oracle Identity Manager and the SPML Web Service are deployed on it.
- Oracle Application Server 10.1.3 installation directory is `E:\product\10.1.3.1\OracleAS_1`.
- The identity store is `oc4jserver.jks`, and the password is `welcome`.
- Certificate request is made for localhost.
- The self-sign certificate is named `oc4jserver.cert`.
- The private key alias is `serveroc4j`, and the password is `welcome`.

12.6.4.2 SSL Certificate Setup

The steps in this section enable you to do the following:

Tip: For enhanced protection, Oracle recommends that you create new certificates (either self-signed or CA certificates) and create a separate keystore and truststore for the client and the server with different passwords. Refer to the SSL configuration documentation of the application server for details.

1. Generating Keys
2. Signing the Certificates
3. Exporting the Certificate
4. Configuring Oracle Application Server

Generating Keys

Generate keys by using the keytool command provided. The following keytool command creates an identity keystore `oc4jserver.jks`:

```
keytool -genkey -alias serveroc4j -keyalg RSA -keysize 1024 -dname
"CN=localhost,OU=Identity,O=Oracle,C=US" -keypass welcome -keystore
E:\product\10.1.3.1\OracleAS_1\oc4jserver.jks -storepass welcome -storetype jks
```

Signing the Certificates

Use the following keytool command to sign the certificates you created:

```
keytool -selfcert -alias serveroc4j -sigalg MD5withRSA -validity 2000 -keypass
welcome -keystore  E:\product\10.1.3.1\OracleAS_1\oc4jserver.jks -storepass
welcome
```

Exporting the Certificate

Use the following keytool command to export the certificate from the identity keystore to a file:

```
keytool -export -alias serveroc4j -file
E:\product\10.1.3.1\OracleAS_1\oc4jserver.cert -keypass welcome -keystore
E:\product\10.1.3.1\OracleAS_1\oc4jserver.jks -storepass welcome -storetype jks
-provider sun.security.provider.Sun
```

Configuring Oracle Application Server

To configure Oracle Application Server:

1. Make a copy of the `E:\product\10.1.3.1\OracleAS_1\j2ee\home\config\default-web-site.xml` file at the same location and rename the copy to `secure-web-site.xml`.
2. In the `secure-web-site.xml` file, modify the following:
   - port attribute=`4443`
   - secure=`true`
   - protocol=`https`

   For example:

   ```
   web-site xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
   xsi:noNamespaceSchemaLocation="http://xmlns.oracle.com/oracleas/schema/web-site
   -10_0.xsd"  port="4443"  secure="true" protocol ="https" display-name="OC4J 10g
   (10.1.3) Default Web Site" schema-major-version="10" schema-minor-version="0">
   ```

3. In the same file, under the `web-site` node, add a new element `ssl-config` to point to the keystore as shown in the following example:

   ```
   <ssl-config keystore="E:/product/10.1.3.1/OracleAS_1/oc4jserver.jks"
   keystore-password="welcome" />
   ```

4. In the `E:\product\10.1.3.1\OracleAS_1\j2ee\home\config\server.xml` file, add the following entry:

   ```
   <web-site path="./secure-web-site.xml"/>
   ```

5. In the `E:\product\10.1.3.1\OracleAS_1\opmn\conf\opmn.xml` file, add the following:

   ```
   <port id="secure-web-site" range="4443" protocol="https"> under <ias-component
   id="default_group">
   ```

6. In the `opmn.xml` file, add the following in the `<data id="java-options" value="-Xrs" under <ias-component id="default_group">` tag:

   ```
   -Djavax.net.ssl.trustStore=E:\product\10.1.3.1\OracleAS_1\oc4jserver.jks
   -Djavax.net.ssl.trustStorePassword=welcome
   -Djavax.net.ssl.keyStore=E:\product\10.1.3.1\OracleAS_1\oc4jserver.jks
   -Djavax.net.ssl.keyStorePassword=welcome"
   ```

7. Restart the server for the changes to take effect.

12.6.5 Enabling SSL for HTTP Communication to Oracle HTTP Server

The following sections provide information about enabling SSL for HTTP communication to Oracle HTTP Server.

By default, the Oracle HTTP Server is configured with SSL and the SSL certificate store, which is located at *ORACLE_HOME*/Apache/Apache/conf/ssl.wlt/default/. The `listen` parameter in the *ORACLE_HOME*/Apache/Apache/conf/ssl.conf file points to the SSL port being used by the Oracle HTTP Server.

You do not make any configuration change to use the default certificate store that comes along with the installation.

> **Tip:** For enhanced protection, Oracle recommends that you create new certificates (either self-signed or CA certificates) and create a separate keystore and truststore for the client and the server with different passwords. Refer to the SSL configuration documentation of the application server for detail.

12.6.5.1 Exporting Certificate

You must export the certificate from the default Oracle wallet *ORACLE_HOME*/Apache/Apache/conf/ssl.wlt/default/ewallet.p12. This certificate is used for the Design Console to trust Oracle Application Server. To export the certificate:

1. Open the *ORACLE_HOME*/Apache/Apache/conf/ssl.wlt/default/ewallet.p12 file by using the Oracle Wallet Manager Console. To do so, click **Open Wallet** and browse to the location of the wallet.
2. When prompted, enter the store password as **welcome**.
3. Right click `Certificate (Ready)`, and click **Export User Certificate**.
4. Save the file as `server.cert`.

> **See Also:** The "Secure Sockets Layer" section in *Oracle Application Server Administrator's Guide* for more information about Oracle Wallet Manager

12.7 Developing the Client for the SPML Web Service

This section provides information and guidelines that you can apply while creating a client for the SPML Web Service.

> **Note:** In this chapter, the client for the SPML Web Service is referred to as the SPML client.

To develop an SPML client, you must refer to the WSDL file for each application server for which you develop the client. The WSDL files for each application server are in the *OIM_HOME*/SPMLWS/SampleHttpClient/wsdl directory.

The code files for a sample SPML client are available at the following location:

OIM_HOME/SPMLWS/SampleHttpClient

To keep sample code easy to understand, this SPML client uses an HTTP connection to the SPML Web Service instead of an HTTP/S connection.

This sample SPML client refers to XML files containing the SOAP requests and performs an HTTP post to the SPML Web Service.

A set of sample SOAP requests is shipped along with this release of Oracle Identity Manager. The following is the format of the path at which you can access the files for these sample SOAP requests:

`SampleHttpClient/sampleRequests/`*`Application Server`*

The `sampleRequests` directory contains separate SOAP requests for each of the supported application servers.

> **Note:** As mentioned earlier, the SPML Web Service supports requests in `UTF-8` encoding only. Therefore, Oracle recommends that you pass SPML requests over HTTP by specifying the charset as `UTF-8` in the content-type header in any client implementation. Refer to the implementation of the sendSOAPRequest function in the sample client provided in the *`OIM_HOME`*`/SPMLWS/SampleSPMLClient/src/testspml/HttpConnect.java` file. In addition, you must serialize the data into a `byte[]` before sending the request. The `main` function in the `SendSPMLRequest.java` file can copy files directly into a `byte[]` and then send it over the HTTP connection.

The following sections provide information that you can apply while developing the SPML client:

- Supported SPML Operations
- Authentication
- Fields Included in SPML Requests
- Structure of the SOAP Header
- Sample SOAP SPML Message

12.7.1 Supported SPML Operations

As mentioned earlier, the SPML Web Service supports the following operations:

- Add operations
- Modify operations

  You must ensure that the psoID is included in the SPML request for the modify operations.
- Delete operations

  You must ensure that the psoID is included in the SPML request for the delete operations.
- Add, Replace, or Delete references
- Lookup operations
- Search operations
- Password operations

You must ensure that the psoID is included in the SPML request for the password operations.

- Suspend, Resume, or Active User operations

  The suspend capability includes the `suspendRequest`, `resumeRequest`, and the `activeUser` operations. You must ensure to include psoID in the SPML request for the `resumeRequest` and the `activeUser` operations.

- ListTargets operations

Note: For more information about psoID, refer to the "Provisioning Operations Supported by the SPML Web Service" section on page 12-3.

12.7.2 Authentication

The SPML client must be authenticated for each SPML request sent. This is to ensure that unauthorized users are not allowed to use the SPML Web Service.

The SPML client can be authenticated in the following way. The SPML client sends the user credentials to the Web Service in the SOAP header. This can be done in the following ways:

- The credentials are provided as header information in a custom tag.
- Oracle WSM is configured to secure the SPML Web Service. In this case, the credentials are sent in standard WSSE tags. A default policy implementation processes these credentials. The default policy file, which is packaged along with the product, is located at the following path:

  OIM_HOME/SPMLWS/OWSMPolicy

Note: For details about configuring Oracle WSM with the SPML Web Service, refer to the "Enabling Security by Using Oracle Web Services Manager and Then Deploying the SPML Web Service" section on page 12-10.

12.7.3 Fields Included in SPML Requests

Table 12–1 lists the mandatory and nonmandatory fields that can be included in SPML requests.

Note: In the following table, certain rows list psoID in the Mandatory Fields column. These requests do not require any Oracle Identity Manager attributes.

Table 12–1 Mandatory and Nonmandatory Fields Included in SPML Requests

SPML Request	Mandatory Fields	Nonmandatory Fields
addRequest for User	Users.User ID Users.First Name Users.Last Name Organizations.Organization Name Users.Xellerate Type Users.Role Users.Password	The rest of the OIM User fields pertaining to the creation of users are nonmandatory.
addRequest for Group	Groups.Group Name	The rest of the OIM User fields pertaining to the creation of groups are nonmandatory.
addRequest for Organization	Organizations.Organization Name	Organizations.Type Organizations.Parent Name
deleteRequest for User	psoID ID="Users:7"	
deleteRequest for Group	psoID ID="Groups:7"	
deleteRequest for Organization	psoID ID="Organizations:9"	
modifyRequest for User	psoID ID="Users:5" One or more modification elements, each corresponding to an attribute to be modified	
modifyRequest for Group	psoID ID="Groups:5" One or more modification elements, each corresponding to an attribute to be modified	
modifyRequest for Organization	psoID ID="Organizations:3" One or more modification elements, each corresponding to an attribute to be modified	
lookupRequest for User	psoID ID="Users:7"	
lookupRequest for Group	psoID ID="Groups:7"	
lookupRequest for Organization	psoID ID="Organizations:9"	
suspendRequest for User	psoID ID="Users:7"	
resumeRequest for User	psoID ID="Users:7"	
activeRequest for User	psoID ID="Users:7"	
setPasswordRequest for User	psoID ID="Users:7"	
resetPasswordRequest for User	psoID ID="Users:7"	

Table 12–1 (Cont.) Mandatory and Nonmandatory Fields Included in SPML Requests

SPML Request	Mandatory Fields	Nonmandatory Fields
searchRequest for User	The `basePsoID` in which the search is performed must be `Organization` or an empty string. For example: `<basePsoID ID="Organization:7"/>, <basePsoID ID=""/>` A set of `dsml:filter` conditions specifying equality matches for the attributes based on which the search has to be performed. **Note**: The objectclass information is sent as a part of one of the filters to specify to the SPML Web Service the container on which the search must be performed. For example: `<dsml:filter><dsml:equalityMatch name="Object Class"><dsml:values>Users</dsml:values></dsml:equalityMatch></dsml:filter>`	

Table 12–1 (Cont.) Mandatory and Nonmandatory Fields Included in SPML Requests

SPML Request	Mandatory Fields	Nonmandatory Fields
searchRequest for Group	The `basePsoID` in which the search is performed must be an empty string. A set of `dsml:filter` conditions specifying equality matches for the attributes based on which the search has to be performed. **Note**: The objectclass information is sent as a part of one of the filters to specify to the SPML Web Service the container on which the search must be performed. For example: `<dsml:filter><dsml:equalityMatch name="Object Class"><dsml:values>Groups</dsml:values></dsml:equalityMatch></dsml:filter>`	
searchRequest for Organization	The `basePsoID` in which the search is performed must be `Organization` or an empty string. A set of `dsml:filter` conditions specifying equality matches for the attributes based on which the search has to be performed. **Note**: The objectclass information is sent as a part of one of the filters to specify to the SPML Web Service the container on which the search must be performed. For example: `<dsml:filter><dsml:equalityMatch name="Object Class"><dsml:values>Organizations</dsml:values></dsml:equalityMatch></dsml:filter>`	
listTargetRequest	None	

12.7.4 Structure of the SOAP Header

The SPML Web Service requires Oracle Identity Manager credentials, which must be provided in the SOAP header depending on whether Oracle WSM is used for securing SPML. This information is explained in the following sections.

Using Custom Security Tags

The custom security tags (`wsa1:OIMUser`) can be used in a SOAP header to embed Oracle Identity Manager credentials when the SOAP request is sent directly to the SPML Web Service. The SPML Web Service interprets these tags, and server-side handlers extract the credential information, as illustrated in the following sample SOAP header:

```
<soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/"
xmlns:xsd="http://www.w3.org/2001/XMLSchema"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
<soapenv:Header>
<wsa1:OIMUser soapenv:actor="http://schemas.xmlsoap.org/soap/actor/next"
```

```
soapenv:mustUnderstand="0" xmlns:wsa1="http://xmlns.oracle.com/OIM/provisioning">
<wsa1:OIMUserPassword>password1</wsa1:OIMUserPassword>
<wsa1:OIMUserId>user1</wsa1:OIMUserId>
</wsa1:OIMUser>
</soapenv:Header>
……..</soapenv:Envelope>
```

Using WSSE Security Tags

If the Oracle WSM Gateway or Agent is used for securing the SPML Web Service, then the SPML SOAP message is intercepted by that Gateway or Agent. In this case, Oracle Identity Manager credentials are provided in standard wsse security tags, as illustrated in the following sample:

```
soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/"
xmlns:soapenc="http://schemas.xmlsoap.org/soap/encoding/"
xmlns:xsd="http://www.w3.org/2001/XMLSchema"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
<soapenv:Header>
  <wsse:Security
xmlns:wsse="http://docs.oasis-open.org/wss/2004/01/oasis-200401-wss-wssecurity-sec
ext-1.0.xsd">
  <wsse:UsernameToken>
  <wsse:Username>user1</wsse:Username>
<wsse:Password
Type="http://docs.oasis-open.org/wss/2004/01/oasis-200401-wss-username-token-profi
le-1.0#PasswordText">password1</wsse:Password>
  </wsse:UsernameToken>
 </wsse:Security>
</soapenv:Header>
……..</soapenv:Envelope>
```

wsa1:lang tag

In addition to Oracle Identity Manager credential information, the SPML client can also send locale information to the SPML Web Service in the SOAP header by using the `wsa1:lang` tag. These tags are then processed by the SPML Web Service. If Oracle Web Services Manager is configured, then the SPML Web Service ignores this tag. In this situation, the header information is as follows:

For custom tags:

```
<soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/"
xmlns:xsd="http://www.w3.org/2001/XMLSchema"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
<soapenv:Header>
<wsa1:OIMUser soapenv:actor="http://schemas.xmlsoap.org/soap/actor/next"
soapenv:mustUnderstand="0" xmlns:wsa1="http://xmlns.oracle.com/OIM/provisioning">
<wsa1:OIMUserPassword>password1</wsa1:OIMUserPassword>
<wsa1:OIMUserId>user1</wsa1:OIMUserId>
</wsa1:OIMUser>
<wsa1:lang soapenv:actor="http://schemas.xmlsoap.org/soap/actor/next"
soapenv:mustUnderstand="0" xmlns:wsa1="http://xmlns.oracle.com/OIM/provisioning">
<wsa1:language>en</wsa1:language>
<wsa1:sublanguage>US</wsa1:sublanguage>
</wsa1:lang>
</soapenv:Header>
……..</soapenv:Envelope>
```

For WSSE tags:

```
<soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/"
xmlns:soapenc="http://schemas.xmlsoap.org/soap/encoding/"
xmlns:xsd="http://www.w3.org/2001/XMLSchema"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
<soapenv:Header>
<wsse:Security
xmlns:wsse="http://docs.oasis-open.org/wss/2004/01/oasis-200401-wss-wssecurity-sec
ext-1.0.xsd">
<wsse:UsernameToken>
<wsse:Username>user1</wsse:Username>
<wsse:Password
Type="http://docs.oasis-open.org/wss/2004/01/oasis-200401-wss-username-token-profi
le-1.0#PasswordText">password1</wsse:Password>
</wsse:UsernameToken>
</wsse:Security>
<wsa1:lang soapenv:actor="http://schemas.xmlsoap.org/soap/actor/next"
soapenv:mustUnderstand="0" xmlns:wsa1="http://xmlns.oracle.com/OIM/provisioning">
<wsa1:language>en</wsa1:language>
<wsa1:sublanguage>US</wsa1:sublanguage>
</wsa1:lang>

</soapenv:Header>
.....</soapenv:Envelope>
```

12.7.5 Sample SOAP SPML Message

The following sample SOAP SPML message is an Add Request operation for the SPML Web Service on Oracle Application Server:

With custom security tags:

```
<?xml version="1.0" encoding="utf-8"?>
<soap:Envelope xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/">
<soap:Header>
<wsa1:OIMUser soap:mustUnderstand="0"
xmlns:wsa1="http://xmlns.oracle.com/OIM/provisioning">
<wsa1:OIMUserId
xmlns:wsa1="http://xmlns.oracle.com/OIM/provisioning">user1</wsa1:OIMUserId>
<wsa1:OIMUserPassword
xmlns:wsa1="http://xmlns.oracle.com/OIM/provisioning">password1</wsa1:OIMUserPassw
ord>
</wsa1:OIMUser>
</soap:Header>
<soap:Body>
<SPMLv2Document xmlns="http://xmlns.oracle.com/OIM/provisioning">
<addRequest returnData="everything" xmlns="urn:oasis:names:tc:SPML:2:0"
xmlns:dsml="urn:oasis:names:tc:DSML:2:0:core">
<data>
  <dsml:attr name="objectclass">
    <dsml:value>Users</dsml:value>
  </dsml:attr>
  <dsml:attr name="Users.User ID">
    <dsml:value>John Doe</dsml:value>
  </dsml:attr>
  <dsml:attr name="Users.First Name">
    <dsml:value>John</dsml:value>
  </dsml:attr>
  <dsml:attr name="Users.Last Name">
    <dsml:value>Doe</dsml:value>
  </dsml:attr>
```

```
  <dsml:attr name="Organizations.Organization Name">
    <dsml:value>Xellerate Users</dsml:value>
  </dsml:attr>
  <dsml:attr name="Users.Xellerate Type">
    <dsml:value>End-User</dsml:value>
  </dsml:attr>
  <dsml:attr name="Users.Role">
    <dsml:value>Full-Time</dsml:value>
  </dsml:attr>
  <dsml:attr name="Users.Password">
    <dsml:value>welcome</dsml:value>
  </dsml:attr>
</data>
</addRequest>
</SPMLv2Document>
</soap:Body>
</soap:Envelope>
```

With WSSE security tags:

```
<?xml version="1.0" encoding="utf-8"?>
<soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/"
xmlns:soapenc="http://schemas.xmlsoap.org/soap/encoding/"
xmlns:xsd="http://www.w3.org/2001/XMLSchema"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
<soapenv:Header>
<wsse:Security
xmlns:wsse="http://docs.oasis-open.org/wss/2004/01/oasis-200401-wss-wssecurity-sec
ext-1.0.xsd">
<wsse:UsernameToken>
<wsse:Username>user1</wsse:Username>
<wsse:Password
Type="http://docs.oasis-open.org/wss/2004/01/oasis-200401-wss-username-token-profi
le-1.0#PasswordText">password1</wsse:Password>
</wsse:UsernameToken>
</wsse:Security>
</soapenv:Header>
<soapenv:Body>
<SPMLv2Document xmlns="http://xmlns.oracle.com/OIM/provisioning">
<addRequest returnData="everything" xmlns="urn:oasis:names:tc:SPML:2:0"
xmlns:dsml="urn:oasis:names:tc:DSML:2:0:core">
<data>
  <dsml:attr name="objectclass">
    <dsml:value>Users</dsml:value>
  </dsml:attr>
  <dsml:attr name="Users.User ID">
    <dsml:value>John Doe</dsml:value>
  </dsml:attr>
  <dsml:attr name="Users.First Name">
    <dsml:value>John</dsml:value>
  </dsml:attr>
  <dsml:attr name="Users.Last Name">
    <dsml:value>Doe</dsml:value>
  </dsml:attr>
  <dsml:attr name="Organizations.Organization Name">
    <dsml:value>Xellerate Users</dsml:value>
  </dsml:attr>
  <dsml:attr name="Users.Xellerate Type">
    <dsml:value>End-User</dsml:value>
  </dsml:attr>
  <dsml:attr name="Users.Role">
```

```
    <dsml:value>Full-Time</dsml:value>
  </dsml:attr>
  <dsml:attr name="Users.Password">
    <dsml:value>welcome</dsml:value>
  </dsml:attr>
</data>
</addRequest>
</SPMLv2Document>
</soapenv:Body>
</soapenv:Envelope>
```

A

Reference for Design Console Users

In the previous release, this appendix described the Design Console actions and adapter task mapping information. From the current release onward, you will find this information in the "Design Console Actions" and the "Adapter Mapping Information" sections in the *Oracle Identity Manager Reference*.

B

Sample SPML Messages

This appendix lists some sample SOAP SPML messages that are supported by the SPML Web service. These SPML messages are embedded in a SOAP request.

Add Request

The following sample contains an Add Request operation to create a user with the user ID `John Doe` and subscribe him to two groups, Groups 5 and Groups 6:

```
<addRequest returnData="everything" xmlns="urn:oasis:names:tc:SPML:2:0"
xmlns:dsml="urn:oasis:names:tc:DSML:2:0:core">
    <data>
           <dsml:attr name="objectclass">
                 <dsml:value>Users</dsml:value>
           </dsml:attr>
          <dsml:attr name="Users.User ID">
                 <dsml:value>John Doe</dsml:value>
           </dsml:attr>
            <dsml:attr name="Users.First Name">
                  <dsml:value>John</dsml:value>
            </dsml:attr>
            <dsml:attr name="Users.Last Name">
                   <dsml:value>Doe</dsml:value>
             </dsml:attr>
             <dsml:attr name="Organizations.Organization Name">
                    <dsml:value>Xellerate Users</dsml:value>
             </dsml:attr>
             <dsml:attr name="Users.Xellerate Type">
                     <dsml:value>End-User</dsml:value>
             </dsml:attr>
             <dsml:attr name="Users.Role">
                      <dsml:value>Full-Time</dsml:value>
              </dsml:attr>
              <dsml:attr name="Users.Password">
                      <dsml:value>welcome</dsml:value>
                </dsml:attr>
        </data>
   <capabilityData capabilityURI="urn:oasis:names:tc:SPML:2:0:reference"
mustUnderstand="true">
        <reference typeOfReference="memberOf"
xmlns="urn:oasis:names:tc:SPML:2:0:reference">
                  <toPsoID ID="Groups:5" />
         </reference>
         <reference typeOfReference="memberOf"
xmlns="urn:oasis:names:tc:SPML:2:0:reference">
                   <toPsoID ID="Groups:6" />
         </reference>
```

```
      </capabilityData>
</addRequest>
```

The following sample contains the success response for the preceding Add Request operation:

```
<addResponse status="success">
<pso>
<psoID ID="Users:7"/>
<data>
<attr name="Users.User ID">
<value>John Doe</value>
</attr>
<attr name="Users.Key">
<value>7</value>
</attr>
<attr name="Users.Last Name">
<value>Doe</value>
</attr>
<attr name="Users.First Name">
<value>John</value>
</attr>
<attr name="Users.Xellerate Type">
<value>End-User</value>
</attr>
<attr name="Users.Creation Date">
<value>2007-08-28 12:42:32.147</value>
</attr>
<attr name="Users.Updated By">
<value>1</value>
</attr>
<attr name="Users.Update Date">
<value>2007-08-28 12:42:36.38</value>
</attr>
<attr name="Users.Status">
<value>Active</value>
</attr>
<attr name="Users.Disable User">
<value>0</value>
</attr>
<attr name="Users.Lock User">
<value>0</value>
</attr>
<attr name="Organizations.Key">
<value>1</value>
</attr>
<attr name="Users.Role">
<value>Full-Time</value>
</attr>
<attr name="Organizations.Organization Name">
<value>Xellerate Users</value>
</attr>
<attr name="Users.Provisioned Date">
<value>2007-08-28 12:42:32.147</value>
</attr>
<attr name="Users.Change Password At Next Logon">
<value>1</value>
</attr>
<attr name="objectclass">
<value>Users</value>
</attr>
```

```
</data>
<capabilityData capabilityURI="urn:oasis:names:tc:SPML:2:0:reference"
mustUnderstand="true">
<reference typeOfReference="memberOf">
<toPsoID ID="Groups:5"/>
</reference>
<reference typeOfReference="memberOf">
<toPsoID ID="Groups:6"/>
</reference>
</capabilityData>
</pso>
</addResponse>
```

The following sample contains the failure response for the Add Request operation in case the user with the user ID already exists:

```
<addResponse status="failure" error="alreadyExists">
<errorMessage>
exception=tcDuplicateUserException;errorMessage=Duplicate_User
</errorMessage>
</addResponse>
```

The following sample contains an Add Request operation to create a group and subscribe it to two groups Groups:4 and Groups:5. The request is also to assign the group to two administrator groups Groups:7 and Groups:8.

```
<addRequest returnData="everything" xmlns="urn:oasis:names:tc:SPML:2:0"
xmlns:dsml="urn:oasis:names:tc:DSML:2:0:core">
<data>
<dsml:attr name="objectclass">
<dsml:value>Groups</dsml:value>
</dsml:attr>
<dsml:attr name="Groups.Group Name">
<dsml:value>Add Group40</dsml:value>
</dsml:attr>
</data>
<capabilityData capabilityURI="urn:oasis:names:tc:SPML:2:0:reference"
mustUnderstand="true">
<reference typeOfReference="memberOf"
xmlns="urn:oasis:names:tc:SPML:2:0:reference">
<toPsoID ID="Groups:4" />
                                                              </reference>
                                <reference typeOfReference="memberOf"
xmlns="urn:oasis:names:tc:SPML:2:0:reference">
                                          <toPsoID ID="Groups:5" />
                                </reference>
<reference typeOfReference="administrator"
xmlns="urn:oasis:names:tc:SPML:2:0:reference">
            <toPsoID ID="Groups:7" />
</reference>
<reference typeOfReference="administrator"
xmlns="urn:oasis:names:tc:SPML:2:0:reference">
            <toPsoID ID="Groups:8" />
</reference>
              </capabilityData>
</addRequest>
```

The following sample contains the success response for the preceding Add Request operation to create a group:

```
<addResponse status="success">
```

```
<pso>
<psoID ID="Groups:11"/>
<data>
<attr name="Groups.Key">
<value>11</value>
</attr>
<attr name="Groups.Group Name">
<value>Add Group40</value>
</attr>
<attr name="Groups.Updated By">
<value>1</value>
</attr>
<attr name="Groups.Update Date">
<value>2007-08-28 15:22:04.953</value>
</attr>
<attr name="Groups.Creation Date">
<value>2007-08-28 15:22:04.953</value>
</attr>
<attr name="Groups.Created By">
<value>1</value>
</attr>
<attr name="objectclass">
<value>Groups</value>
</attr>
</data>
<capabilityData capabilityURI="urn:oasis:names:tc:SPML:2:0:reference"
mustUnderstand="true">
<reference typeOfReference="memberOf">
<toPsoID ID="Groups:4"/>
</reference>
<reference typeOfReference="memberOf">
<toPsoID ID="Groups:5"/>
</reference>
<reference typeOfReference="administrator">
<toPsoID ID="Groups:1"/>
</reference>
<reference typeOfReference="administrator">
<toPsoID ID="Groups:7"/>
</reference>
<reference typeOfReference="administrator">
<toPsoID ID="Groups:8"/>
</reference>
</capabilityData>
</pso>
</addResponse>
```

The following sample contains the failure response for the preceding operation in case the group already exists:

```
<addResponse status="failure" error="alreadyExists">
<errorMessage>exception=Duplicate_Group</errorMessage>
</addResponse>
```

Modify Request

The following sample contains a Modify Request operation to unsubscribe a user from two groups and subscribe him to two new groups, Groups:4 and Groups:5:

```
<modifyRequest returnData="everything" xmlns="urn:oasis:names:tc:SPML:2:0"
xmlns:dsml="urn:oasis:names:tc:DSML:2:0:core">
<psoID ID="Users:31"></psoID>
```

```
    <modification modificationMode="add">
             <capabilityData capabilityURI="urn:oasis:names:tc:SPML:2:0:reference"
mustUnderstand="true">
                    <reference typeOfReference="memberOf"
xmlns="urn:oasis:names:tc:SPML:2:0:reference">
                                <toPsoID ID="Groups:4" />
                    </reference>
                     <reference typeOfReference="memberOf"
xmlns="urn:oasis:names:tc:SPML:2:0:reference">
                                <toPsoID ID="Groups:5"/>
                     </reference>
              </capabilityData>
</modification>
<modification modificationMode="delete">
            <capabilityData capabilityURI="urn:oasis:names:tc:SPML:2:0:reference"
mustUnderstand="true">
                      <reference typeOfReference="memberOf"
xmlns="urn:oasis:names:tc:SPML:2:0:reference">
                               <toPsoID ID="Groups:6"/>
                      </reference>
                     <reference typeOfReference="memberOf"
xmlns="urn:oasis:names:tc:SPML:2:0:reference">
                             <toPsoID ID="Groups:7"/>
                      </reference>
           </capabilityData>
   </modification>
</modifyRequest>
```

The following sample contains the success response for the preceding Modify Request operation:

```
<modifyResponse status="success">
<pso>
<psoID ID="Users:31"/>
<data>
<attr name="Users.User ID">
<value>John Doe</value>
</attr>
<attr name="Users.Key">
<value>7</value>
</attr>
<attr name="Users.Last Name">
<value>Doe</value>
</attr>
<attr name="Users.First Name">
<value>John</value>
</attr>
<attr name="Users.Xellerate Type">
<value>End-User</value>
</attr>
<attr name="Users.Creation Date">
<value>2007-08-28 12:42:32.147</value>
</attr>
<attr name="Users.Updated By">
<value>1</value>
</attr>
<attr name="Users.Update Date">
<value>2007-08-28 12:42:36.38</value>
</attr>
<attr name="Users.Status">
<value>Active</value>
```

```
</attr>
<attr name="Users.Disable User">
<value>0</value>
</attr>
<attr name="Users.Lock User">
<value>0</value>
</attr>
<attr name="Organizations.Key">
<value>1</value>
</attr>
<attr name="Users.Role">
<value>Full-Time</value>
</attr>
<attr name="Organizations.Organization Name">
<value>Xellerate Users</value>
</attr>
<attr name="Users.Provisioned Date">
<value>2007-08-28 12:42:32.147</value>
</attr>
<attr name="Users.Change Password At Next Logon">
<value>1</value>
</attr>
</data>
<capabilityData capabilityURI="urn:oasis:names:tc:SPML:2:0:reference"
mustUnderstand="true">
<reference typeOfReference="memberOf">
<toPsoID ID="Groups:3"/>
</reference>
<reference typeOfReference="memberOf">
<toPsoID ID="Groups:4"/>
</reference>
<reference typeOfReference="memberOf">
<toPsoID ID="Groups:5"/>
</reference>
</capabilityData>
</pso>
</modifyResponse>
```

The following sample contains the failure response for the preceding operation in case the user is not available:

```
<modifyResponse status="failure" error="noSuchIdentifier">
<errorMessage>
exception=OIMSpmlException;errorMessage=NO_USER_ID_DEFINED
</errorMessage>
</modifyResponse>
```

The following sample contains a Modify Request operation that modifies the group Groups:36 and its group references by adding two more group membership (Groups:7 and Groups:8) and a group administrator (Groups:9). It also deletes an existing group membership (Groups:10) and an administrator reference (Groups:11).

```
<modifyRequest returnData="everything" xmlns="urn:oasis:names:tc:SPML:2:0"
xmlns:dsml="urn:oasis:names:tc:DSML:2:0:core">
       <psoID ID="Groups:36"></psoID>
        <modification>
                   <dsml:modification name="Groups.Group Name" operation="add">
                           <dsml:value>Changed</dsml:value>
                   </dsml:modification>
          </modification>
          <modification modificationMode="add">
```

```
                  <capabilityData
capabilityURI="urn:oasis:names:tc:SPML:2:0:reference" mustUnderstand="true">
                           <reference typeOfReference="memberOf"
xmlns="urn:oasis:names:tc:SPML:2:0:reference">
                                        <toPsoID ID="Groups:7"/>
                            </reference>
                            <reference typeOfReference="memberOf"
xmlns="urn:oasis:names:tc:SPML:2:0:reference">
                                        <toPsoID ID="Groups:8"/>
                           </reference>
                          <reference typeOfReference="administrator"
xmlns="urn:oasis:names:tc:SPML:2:0:reference">
                                       <toPsoID ID="Groups:9"/>
                           </reference>
                 </capabilityData>
          </modification>
          <modification modificationMode="delete">
                <capabilityData
capabilityURI="urn:oasis:names:tc:SPML:2:0:reference" mustUnderstand="true">
                      <reference typeOfReference="administrator"
xmlns="urn:oasis:names:tc:SPML:2:0:reference">
                                  <toPsoID ID="Groups:10"/>
                      </reference>
                       <reference typeOfReference="memberOf"
xmlns="urn:oasis:names:tc:SPML:2:0:reference">
                               <toPsoID ID="Groups:11"/>
                       </reference>
                </capabilityData>
           </modification>
</modifyRequest>
```

The following sample contains the success response for the preceding Modify Request operation:

```
<modifyResponse status="success">
<pso>
<psoID ID="Groups:36"/>
<data>
<attr name="Groups.Key">
<value>36</value>
</attr>
<attr name="Groups.Group Name">
<value>Changed</value>
</attr>
<attr name="Groups.Updated By">
<value>1</value>
</attr>
<attr name="Groups.Update Date">
<value>2007-08-29 21:00:51.657</value>
</attr>
<attr name="Groups.Creation Date">
<value>2007-08-27 15:22:43.97</value>
</attr>
<attr name="Groups.Created By">
<value>1</value>
</attr>
</data>
<capabilityData capabilityURI="urn:oasis:names:tc:SPML:2:0:reference"
mustUnderstand="true">
<reference typeOfReference="memberOf">
```

```
<toPsoID ID="Groups:7"/>
</reference>
<reference typeOfReference="memberOf">
<toPsoID ID="Groups:8"/>
</reference>
<reference typeOfReference="administrator">
<toPsoID ID="Groups:1"/>
</reference>
<reference typeOfReference="administrator">
<toPsoID ID="Groups:9"/>
</reference>
</capabilityData>
</pso>
</modifyResponse>
```

The following sample contains the failure response for the preceding operation in case the group ID is missing:

```
<modifyResponse status="failure" error="noSuchIdentifier">
<errorMessage>
exception=OIMSpmlException;errorMessage=GROUP_NOT_FOUND
</errorMessage>
</modifyResponse>
```

Add Request With Date Format

The following sample Add Request operation adds a user by assigning some start date and end date attribute. The date format should assume the format- yyyy-mm-dd HH:MM:SS.sss.

```
<addRequest returnData="everything" xmlns="urn:oasis:names:tc:SPML:2:0"
xmlns:dsml="urn:oasis:names:tc:DSML:2:0:core">
    <data>
 <dsml:attr name="objectclass">
         <dsml:value>Users</dsml:value>
</dsml:attr>
<dsml:attr name="Users.User ID">
    <dsml:value>John Doe</dsml:value>
</dsml:attr>
<dsml:attr name="Users.First Name">
    <dsml:value>John</dsml:value>
</dsml:attr>
<dsml:attr name="Users.Last Name">
    <dsml:value>Doe</dsml:value>
</dsml:attr>
<dsml:attr name="Organizations.Organization Name">
    <dsml:value>Xellerate Users</dsml:value>
</dsml:attr>
<dsml:attr name="Users.Xellerate Type">
    <dsml:value>End-User</dsml:value>
</dsml:attr>
<dsml:attr name="Users.Manager Login">
    <dsml:value>Jane Doe</dsml:value>
</dsml:attr>
<dsml:attr name="Users.Role">
    <dsml:value>Full-Time</dsml:value>
</dsml:attr>
<dsml:attr name="Users.Password">
    <dsml:value>welcome</dsml:value>
</dsml:attr>
```

```
<dsml:attr name="Users.Start Date">
    <dsml:value>2007-06-18 00:00:00.000</dsml:value>
</dsml:attr>
<dsml:attr name="Users.End Date">
    <dsml:value>2017-06-18 00:00:00.000</dsml:value>
</dsml:attr>
  </data>
</addRequest>
```

The following sample contains the success response for the preceding Add Request operation:

```
<addResponse status="success">
<pso>
<psoID ID="Users:8"/>
<data>
<attr name="Users.User ID">
<value>John Doe</value>
</attr>
<attr name="Users.Key">
<value>8</value>
</attr>
<attr name="Users.Last Name">
<value>Doe</value>
</attr>
<attr name="Users.First Name">
<value>John</value>
</attr>
<attr name="Users.Manager Key">
<value>4</value>
</attr>
<attr name="Users.Manager Login">
<value>John</value>
</attr>
<attr name="Users.Manager First Name">
<value>John</value>
</attr>
<attr name="Users.Manager Last Name">
<value>Doe</value>
</attr>
<attr name="Users.Xellerate Type">
<value>End-User</value>
</attr>
<attr name="Users.Creation Date">
<value>2007-08-29 21:27:03.39</value>
</attr>
<attr name="Users.Updated By">
<value>1</value>
</attr>
<attr name="Users.Update Date">
<value>2007-08-29 21:27:06.937</value>
</attr>
<attr name="Users.Status">
<value>Active</value>
</attr>
<attr name="Users.Disable User">
<value>0</value>
</attr>
<attr name="Users.Lock User">
<value>0</value>
</attr>
```

```
<attr name="Organizations.Key">
<value>1</value>
</attr>
<attr name="Users.Role">
<value>Full-Time</value>
</attr>
<attr name="Organizations.Organization Name">
<value>Xellerate Users</value>
</attr>
<attr name="Users.Start Date">
<value>2007-06-18 00:00:00.0</value>
</attr>
<attr name="Users.End Date">
<value>2017-06-18 00:00:00.0</value>
</attr>
<attr name="Users.Provisioning Date">
<value>2007-06-18 00:00:00.0</value>
</attr>
<attr name="Users.Provisioned Date">
<value>2007-08-29 21:27:03.39</value>
</attr>
<attr name="Users.Change Password At Next Logon">
<value>1</value>
</attr>
<attr name="objectclass">
<value>Users</value>
</attr>
</data>
<capabilityData capabilityURI="urn:oasis:names:tc:SPML:2:0:reference"
mustUnderstand="true">
<reference typeOfReference="memberOf">
<toPsoID ID="Groups:3"/>
</reference>
</capabilityData>
</pso>
</addResponse>
```

The following sample contains the failure response for the preceding operation in case the user already exists:

```
<addResponse status="failure" error="alreadyExists">
<errorMessage>
exception=tcDuplicateUserException;errorMessage=Duplicate_User
</errorMessage>
</addResponse>
```

Index