Oracle® Fusion Middleware

Upgrade Guide for Oracle SOA Suite, WebCenter Portal, and ADF

11*g* Release 1 (11.1.1.6.0)

E10127-06

July 2012

ORACLE®

Oracle Fusion Middleware Upgrade Guide for Oracle SOA Suite, WebCenter Portal, and ADF, 11*g* Release 1 (11.1.1.6.0)

E10127-06

Primary Author: Showvik Roychowdhuri

Contributing Authors: Liz Lynch, Savita Thakur, Thomas Van Raalte

Contributors: Tal Broda, Paul Dickson, Robert Donat, Bruno Henriques, Jay Kasi, Yogeshwar Kuntawar, HT Ma, Sankar Mani, Michael Meiner, William Norcott, Sundari Revanur, Michael Rubino, Reza Shafii, Vishal Sharma, Sitaraman Swaminathan, Arun Thomas

Contents

4 Special Considerations for Oracle BPEL Long-Running Processes

Part II   Upgrading the Oracle SOA Suite, WebCenter, and ADF Environment

5 Upgrading an Oracle ADF 10*g* Environment

6 Upgrading an Oracle SOA Suite 10*g* Environment

7 Upgrading an Oracle WebCenter 10*g* Environment

Part III   Upgrading Oracle SOA Suite, Oracle WebCenter, and ADF Applications

8 Overview of Upgrading Oracle SOA Suite, WebCenter, and ADF Applications

9 Considerations When Upgrading All Oracle SOA Applications

10   Upgrading Oracle BPEL Process Manager Applications

11 Upgrading Oracle Business Activity Monitoring Applications

12 Upgrading Oracle B2B Applications

13 Upgrading Oracle Web Services Manager Policies and OC4J Security Enviroments

14 Upgrading Other Oracle SOA Suite Technologies

15 Considerations When Upgrading Oracle WebCenter Applications

16   Additional Oracle WebCenter Upgrade Details

A   Example: Upgrading and Redeploying SOA Order Booking

Preface

This preface contains the following sections:

- Audience
- Documentation Accessibility
- Related Documents
- Conventions

Audience

This guide is intended for Oracle Fusion Middleware system administrators who are responsible for installing, maintaining, and upgrading Oracle Fusion Middleware instances. It is assumed that readers of this manual have knowledge of the following:

- Oracle Fusion Middleware system administration and configuration
- The configuration and expected behavior of the system being upgraded

Documentation Accessibility

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

Access to Oracle Support

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

Related Documents

For more information, see the following related documentation available in the Oracle Fusion Middleware 11*g* documentation library:

- *Oracle Fusion Middleware Upgrade Planning Guide*
- *Oracle Fusion Middleware Installation Guide for Oracle SOA Suite and Oracle Business Process Management Suite*
- *Oracle Fusion Middleware Administrator's Guide*

> **Note:** For more detailed information about the Oracle Fusion
> Middleware upgrade documentation, see "Documentation Roadmap
> for Upgrading Your Middleware Environment" in the *Oracle Fusion
> Middleware Upgrade Planning Guide*.

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.

1

Summary of the Oracle SOA Suite, WebCenter, and ADF Upgrade Process

This chapter provides a high-level overview of the Oracle SOA Suite, WebCenter, and ADF upgrade process. Refer to the following sections for more information:

- Flow Chart of the Oracle SOA Suite, WebCenter, and ADF Upgrade Process
- Table Describing the Steps in the Oracle SOA Suite and WebCenter Upgrade Process

1.1 Flow Chart of the Oracle SOA Suite, WebCenter, and ADF Upgrade Process

Figure 1–1 provides a flow chart of the Oracle SOA, WebCenter, and ADF upgrade process. Review this chart to get familiar with the tasks you will be required to perform, based on your existing version of Oracle Application Server.

Figure 1–1   Flow Chart of the Oracle SOA Suite and WebCenter Upgrade Process
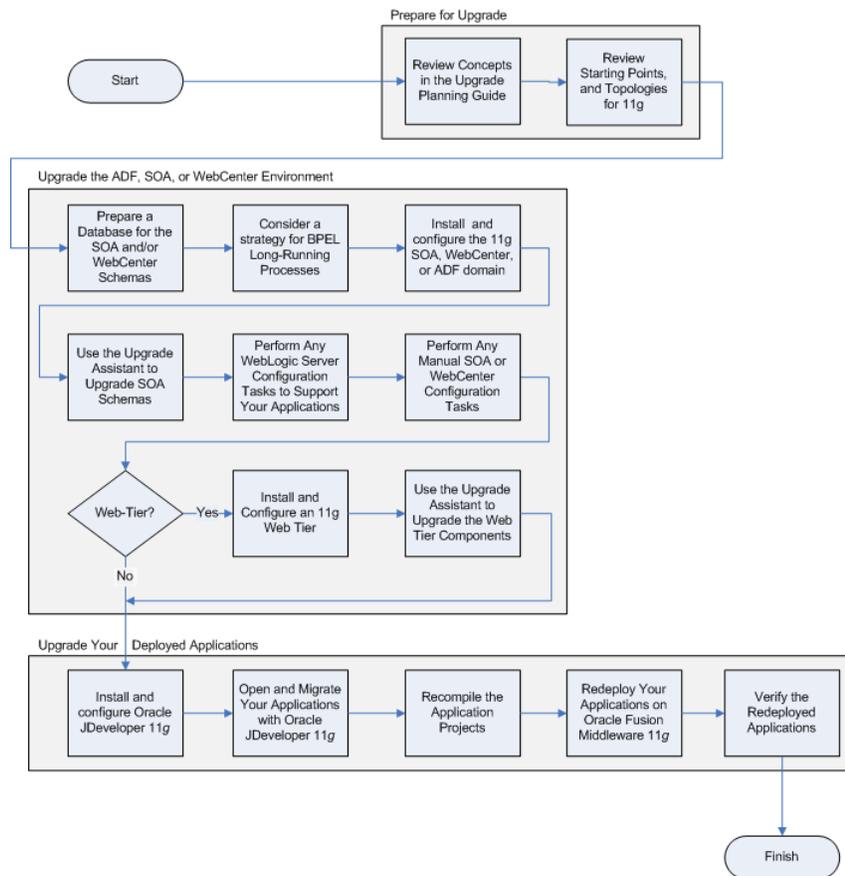
1.2 Table Describing the Steps in the Oracle SOA Suite and WebCenter Upgrade Process

Table 1–1 describes each of the steps in the upgrade process flow chart, which is shown in Figure 1–1. The table also provides information on where to get more information on each step in the process.

Table 1–1   Table Describing the Steps in the Oracle SOA Suite and WebCenter Upgrade Process

Step	Description	More Information
Review Concepts in the Oracle Fusion Middleware Upgrade Planning Guide	The *Oracle Fusion Middleware Upgrade Planning Guide* provides a high-level overview of how to upgrade your entire existing Oracle Application Server 10*g* environment to Oracle Fusion Middleware. It also includes compatibility information and instructions for upgrading any databases that support your middleware components.	*Oracle Fusion Middleware Upgrade Planning Guide*

Table 1–1   (Cont.) Table Describing the Steps in the Oracle SOA Suite and WebCenter Upgrade Process

Step	Description	More Information
Review the Starting Points and Topologies for 11*g*	Before starting your upgrade, you should be familiar with the new features, supported starting points, and recommended topologies for Oracle Fusion Middleware 11*g*. Oracle Fusion Middleware 11*g* introduces a new architecture, based on Oracle WebLogic Server. You should get familiar with the new 11*g* features before starting your upgrade.	Part I, "Preparing to Upgrade Oracle SOA Suite, WebCenter, and ADF" Chapter 3, "Oracle SOA Suite, WebCenter, and ADF Information for 10g Users"
Prepare a Database for the SOA and WebCenter Schemas	The Oracle SOA Suite and WebCenter components require schemas that must be installed in a database. As a result, before you begin the upgrade of a SOA or WebCenter environment, make sure your database is a supported version, and then use the Oracle Fusion Middleware Metadata Repository Creation Utility to install the new, 11*g* schemas.	SOA Environments: ■ Section 6.2, "Task 2: If Necessary, Upgrade the Oracle Database That Hosts the Oracle SOA Suite Schemas" ■ Section 6.3, "Task 3: Use the Repository Creation Utility to Install the 11g SOA Schemas" WebCenter Environments: ■ Section 7.2, "Task 2: If Necessary, Upgrade the Oracle Database That Hosts the WebCenter Schemas" ■ Section 7.3, "Task 3: Use the Repository Creation Utility to Install the Oracle WebCenter Portal 11g Schemas"
Consider a strategy for BPEL Long-Running Processes	If you are supporting any Oracle BPEL Process Manager long-running processes, consider the your options when upgrading your Oracle SOA Suite environment.	Chapter 4, "Special Considerations for Oracle BPEL Long-Running Processes"
Install the 11g SOA, WebCenter, or ADF, and create the Middleware home.	Use the appropriate software tools to: ■ Install the Oracle WebLogic Server and create the Middleware home. ■ Install Oracle Fusion Middleware software and create the Oracle home ■ Apply the latest Fusion Middleware Patch Set ■ Configure the domain ■ Verify the domain	SOA Environments: ■ Section 6.4, "Task 4: Install and Configure the Oracle SOA Suite 11g Middle Tiers" WebCenter environments: ■ Section 7.4, "Task 4: Install and Configure the Oracle WebCenter Portal 11g Middle Tiers" ADF environments: ■ Section 5.2, "Task 2: Install and Configure a Oracle WebLogic Server Domain For Your ADF Applications"
Use the Upgrade Assistant to Upgrade the SOA Schemas	If you are using Oracle Business Activity Monitoring or Oracle B2B, use the Upgrade Assistant to upgrade the BAM and B2B schemas.	Section 6.5, "Task 5: If Necessary, Upgrade the SOA Component Schemas"

Table 1–1 (Cont.) Table Describing the Steps in the Oracle SOA Suite and WebCenter Upgrade Process

Step	Description	More Information
Perform Any WebLogic Server Configuration Tasks to Support Your Applications	Manually modify any Oracle WebLogic Server properties or configuration settings; create data sources or other Java EE resources required by your applications.	*Oracle Fusion Middleware Upgrade Guide for Java EE*
Perform Any Manual SOA or WebCenter Configuration Tasks	Often, there are configuration tasks that you need to perform on the new Oracle WebLogic Server environment to support your Oracle SOA or WebCenter applications. In most cases, no manual configuration tasks are required to support ADF applications, as long as you have installed and configured your ADF environment as documented.	Chapter 9, "Considerations When Upgrading All Oracle SOA Applications" Chapter 15, "Considerations When Upgrading Oracle WebCenter Applications"
Install and Configure an 11*g* Web Tier	Use the appropriate software installer and configuration wizard to install and configure the Oracle HTTP Server and Oracle Web Cache Web Tier components.	"Task 1: Understand the Differences Between Using Oracle HTTP Server with OC4J and Oracle WebLogic Server" in the *Oracle Fusion Middleware Upgrade Guide for Java EE*
Use the Upgrade Assistant to Upgrade the Web Tier Components	The Oracle Fusion Middleware Upgrade Assistant copies configuration data from your existing Oracle Application Server Web tier components to the newly installed Oracle Fusion Middleware Web tier components.	"Upgrading a Web Server and Java EE Environment" in the *Oracle Fusion Middleware Upgrade Guide for Java EE*
Install and Configure Oracle JDeveloper 11*g*	You can download Oracle JDeveloper 11*g* Release 1 (11.1.1.6.0) from the Oracle Technology Network (OTN), or copy the ZIP file from the Oracle Fusion Middleware Companion CD–ROM.	Section 8.5, "Task 5: Install and Start Oracle JDeveloper 11g"
Open and Migrate Your Applications with Oracle JDeveloper Release 11*g*	Oracle JDeveloper 11 offers a migration feature that helps you convert your Oracle Application Server 10g applications so they can be recompiled for Oracle Fusion Middleware 11*g* Release 1 (11.1.1.6.0).	Section 8.6, "Task 6: Open and Upgrade the Application in Oracle JDeveloper"
Recompile the Application Projects	The application sources should be recompiled with JDeveloper 11*g*; check for any compilation errors.	Section 8.8, "Task 8: Review and Recompile the Application"
Implement any other code changes required for the application to run successfully on Oracle Fusion Middleware 11*g*	Identify any particular technologies or services required by your application and review the considerations	
Redeploy Your Applications on Oracle Fusion Middleware 11*g*	You can use JDeveloper to deploy your applications, or you can create an EAR file and deploy the application using the `asctl` command line or using Oracle Enterprise Manager Fusion Middleware Control.	Section 8.9, "Task 9: Deploy the Application"

Table 1–1 (Cont.) Table Describing the Steps in the Oracle SOA Suite and WebCenter Upgrade Process

Step	Description	More Information
Verify the Redeployed Applications	Make sure that the redeployed applications are working properly and no errors occurred during the redeployment.	N/A

Part I

Preparing to Upgrade Oracle SOA Suite, WebCenter, and ADF

Part I contains the following chapters:

- Chapter 2, "Supported Starting Points for Oracle SOA Suite, WebCenter, and ADF Upgrade"

- Chapter 3, "Oracle SOA Suite, WebCenter, and ADF Information for 10g Users"

- Chapter 4, "Special Considerations for Oracle BPEL Long-Running Processes"

2

Supported Starting Points for Oracle SOA Suite, WebCenter, and ADF Upgrade

Refer to the following sections for information about the supported starting points for upgrading to Oracle SOA Suite, WebCenter, and ADF 11*g*:

- Summary of Supported Upgrade Starting Points for Oracle SOA Suite, WebCenter, and ADF
- Information for Oracle SOA Suite 10g Release 2 (10.1.2) Users

2.1 Summary of Supported Upgrade Starting Points for Oracle SOA Suite, WebCenter, and ADF

This guide provides instructions for upgrading from the Oracle Application Server releases and installation types described in Table 2–1.

Table 2–1   Releases and Installation Types Supported By This Guide

Release	Installation Type	Description or Notes	Latest Patch Sets
10*g* Release 3 (10.1.3.1.0)	All Installation Types	This release included the components of the Oracle SOA Suite	10*g* Release 3 (10.1.3.5)
10*g* Release 3 (10.1.3.2.0)	All Installation Types	This release included the Oracle WebCenter components	10*g* Release 3 (10.1.3.5)

The patch sets listed in Table 2–1 were the latest patch sets available at the time this guide was published. For a list of the latest patch sets available for your installation, refer to My Oracle Support (formerly Oracle*MetaLink*):

`http://support.oracle.com/`

> **Note:**   If you are currently running Oracle Fusion Middleware 11*g*, refer to the *Oracle Fusion Middleware Patching Guide*, which provides information about applying the latest Oracle Fusion Middleware patches.
>
> This guide, as well as the other upgrade guides available in the Oracle Fusion Middleware 11*g* documentation library, provide instructions for upgrading from Oracle Application Server 10*g* to the latest Oracle Fusion Middleware 11*g* release.

2.2 Information for Oracle SOA Suite 10*g* Release 2 (10.1.2) Users

If you are using Oracle BPEL Process Manager or Oracle BAM 10g Release 2 (10.1.2), then you must first migrate your projects to 10*g* Release 3 (10.1.3) before upgrading to Oracle Fusion Middleware 11g. For more information on migrating from Oracle BPEL Process Manager 10*g* Release 2 (10.1.2) to 10*g* Release 3 (10.1.3), see:

- "Oracle BPEL Process Manager Migration" in the 10g Release 3 (10.1.3.1) *Oracle BPEL Process Manager Installation Guide*

- "Migrating Workflow Definitions from 10.1.2 to 10.1.3.1" in the 10g Release 3 (10.1.3.1) *Oracle BPEL Process Manager Developer's Guide*

3

Oracle SOA Suite, WebCenter, and ADF Information for 10*g* Users

This chapter provides an overview of the Oracle SOA Suite, WebCenter, and ADF software available in Oracle Fusion Middleware 11*g*. The following sections compare the architecture and directory structure of these technologies in 10*g* Release 3 (10.1.3) and in Oracle Fusion Middleware 11*g*:

- Information for Oracle ADF Users
- Information for Oracle SOA Suite Users
- Information for Oracle WebCenter Users

3.1 Information for Oracle ADF Users

In Oracle Application Server 10*g* Release 3 (10.1.3), Oracle provided the Oracle Application Development Framework (ADF), an end-to-end application framework that builds on Java EE standards and open-source technologies to simplify and accelerate implementing service-oriented applications.

The development and runtime environment required to deploy and manage ADF applications is similar in many ways to the environment required for other Java EE applications. For information about the types of Java EE environments available in 10*g* and instructions for upgrading those environments to Oracle Fusion Middleware 11*g*, refer to the *Oracle Fusion Middleware Upgrade Guide for Java EE*.

The difference between a typical Java EE environment and an environment that supports ADF applications is the availability of the ADF runtime libraries:

- In Oracle Application Server 10*g*, each instance of OC4J automatically provided the ADF runtime libraries required to support ADF applications.

- In Oracle Fusion Middleware 11*g*, an Oracle WebLogic Server domain, by default, does not contain the ADF runtime libraries. However, you can optionally configure or extend your domain to include the Java Runtime Files (JRF). The Oracle ADF runtime libraries are included as part of the JRF component.

Figure 3–1 compares how Oracle ADF applications are supported in OC4J and in Oracle WebLogic Server. Note that the Oracle WebLogic Server domain can be extended with the Java Runtime Files (JRF) domain template, which includes the required Oracle ADF libraries, as well as other important Oracle-specific technologies.

Figure 3–1   Support for Oracle ADF Applications in OC4J and Oracle WebLogic Server
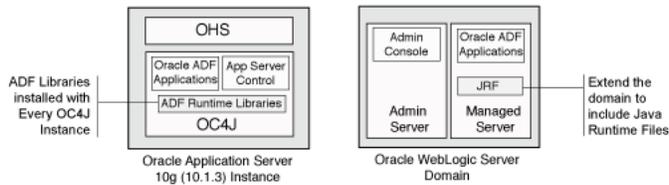
For more information about extending your domain to support Oracle ADF, see Chapter 17, "Scaling Your Environment," in the *Oracle Fusion Middleware Administrator's Guide*.

Note that unlike Oracle Application Server 10*g*, Oracle HTTP Server is not automatically installed and configured to route requests to Oracle WebLogic Server. Instead, you install Oracle HTTP Server and Oracle Web Cache in a separate Oracle home using the WebTier and Utilities CD-ROM. For more information, see "Upgrading a Web Server and Java EE Environment" in the *Oracle Fusion Middleware Upgrade Guide for Java EE*.

3.2 Information for Oracle SOA Suite Users

Oracle Application Server 10*g* Release 3 (10.1.3) provided installation options that allowed you to install and configure the Oracle Service Oriented Architecture (SOA) system components (except for Oracle BAM) as part of the Oracle Application Server installation. You could then use Oracle JDeveloper to develop and deploy SOA applications on the OC4J instances where the SOA components were installed.

Similarly, for Oracle Fusion Middleware, you can install and configure an Oracle WebLogic Server domain that supports the Oracle SOA infrastructure and system components, including Oracle BAM.

For more information, refer to the following sections:

■ Comparing Oracle SOA Suite on OC4J and Oracle WebLogic Server

■ Comparing the Oracle SOA Suite Directory Structure on OC4J and Oracle WebLogic Server

■ Upgrading an Oracle SOA Suite Enterprise Deployment

■ About Oracle BAM and Oracle B2B in Oracle Application Server 10g and Oracle Fusion Middleware 11g

3.2.1 Comparing Oracle SOA Suite on OC4J and Oracle WebLogic Server

Figure 3–2 shows how the SOA system components are installed as part of an OC4J 10*g* instance and compares that to how the SOA components are installed on a managed server when you install and configure the Oracle Fusion Middleware Oracle SOA Suite 11*g*.

Note that the SOA system components require a set of SOA schemas, which you install using the Repository Creation Utility (RCU). For more information about the schemas installed by RCU, refer to the *Oracle Fusion Middleware Repository Creation Utility User's Guide*.

Figure 3–2   Oracle SOA Suite Installed on an OC4J Instance and an Oracle WebLogic Server Domain
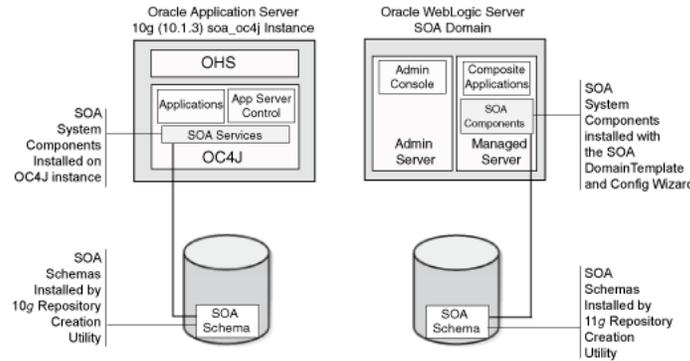
3.2.2 Comparing the Oracle SOA Suite Directory Structure on OC4J and Oracle WebLogic Server

When you install and configure the Oracle SOA Suite, you create a Oracle WebLogic Server domain that has been extended to support the SOA system components. The resulting files on disk consist of the SOA domain directory structure, as well as the SOA Oracle home, which contains the SOA component binary and support files, and the Oracle Common home, which contains Oracle Enterprise Manager Fusion Middleware Control and the Java Required Files (JRF).

The SOA Oracle home must be installed inside the Middleware home; the Oracle comon is installed automatically.

This is in contrast to the single Oracle home that was created during an Oracle Application Server 10g Release 3 (10.1.3) installation.

Figure 3–3 shows the directory structure that results from installing and configuring the Oracle SOA Suite domain.

Figure 3–3   The Directory Structure of an Oracle Fusion Middleware SOA Suite Installation
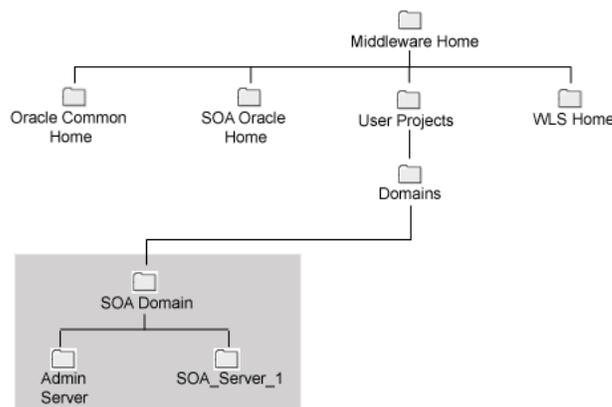
3.2.3 Upgrading an Oracle SOA Suite Enterprise Deployment

The instructions in this guide provide the steps for upgrading a basic Oracle SOA Suite environment. After reviewing the content of this guide, you should be able to transfer this knowledge to a more complex production environment.

For specific information about setting up a production environment that takes advantage of clustering and high availability, refer to the *Oracle Fusion Middleware Enterprise Deployment Guide for Oracle SOA Suite*.

For an overview of the high-level tasks you must perform when upgrading an enterprise deployment, see "Planning an Enterprise Deployment" in the *Oracle Fusion Middleware Upgrade Planning Guide*.

3.2.4 About Oracle BAM and Oracle B2B in Oracle Application Server 10*g* and Oracle Fusion Middleware 11*g*

Unlike the other Oracle SOA Suite components, Oracle BAM 10*g* and Oracle B2B were installed an configured separately.

Oracle BAM was supported on the Microsoft platform only, and Oracle B2B was installed on a separate server from the other Oracle SOA Suite 10*g* components.

For Oracle Fusion Middleware 11*g*, Oracle BAM 11*g* and Oracle B2B 11*g* are integrated with the other Oracle SOA Suite components. Oracle BAM can be deployed as part of the same Oracle WebLogic Server domain, and Oracle B2B is installed and configured automatically with the other Oracle SOA Suite components.

3.3 Information for Oracle WebCenter Users

Oracle Application Server 10*g* Release 3 (10.1.3) provided installation options that allowed you to install and configure the WebCenter system components as part of an Oracle Application Server install. You could then use Oracle JDeveloper to develop and deploy WebCenter applications on the OC4J instances where the WebCenter system components were installed.

Similarly, for Oracle Fusion Middleware, you can install and configure a Oracle WebLogic Server domain that supports the WebCenter system components.

Refer to the following sections for more information:

- New Product Names for Oracle WebCenter and Oracle Enterprise Content Management Suite

- Comparing Oracle WebCenter on OC4J and Oracle WebLogic Server

- Comparing the Oracle WebCenter Directory Structure on OC4J and Oracle WebLogic Server

3.3.1 New Product Names for Oracle WebCenter and Oracle Enterprise Content Management Suite

With Oracle Fusion Middleware 11*g* Release 1 (11.1.1.6.0), Oracle WebCenter and Oracle Universal Content Management have been rebranded. Table 3–1 provides a mapping of the old product names and terminology with the new ones.

> **Note:** This guide uses the old terminology unless a product name is qualified by the release number 11.1.1.6.0 or 11*g*, or a release 11.1.1.6.0-specific feature is discussed.

Table 3–1   Mapping Old Terminology with Release 11.1.1.6.0 Terminology

Old Name	New Name in Release 11.1.1.6.0
Product Name Changes	
Oracle WebCenter	Oracle WebCenter Portal
Oracle Universal Content Management	Oracle WebCenter Content
Oracle Content Server	Oracle WebCenter Content Server
Oracle WebCenter Discussions	Oracle WebCenter Portal's Discussion Server
Feature Name Changes in Documentation	
WebCenter application	WebCenter Portal: Framework application/Framework application
	WebCenter Portal application is the umbrella term for Framework applications, Portlet Producer applications, and the Spaces application
WebCenter domain	WebCenter Portal domain

3.3.2 Comparing Oracle WebCenter on OC4J and Oracle WebLogic Server

Figure 3–4 shows the how the Oracle WebCenter system components are installed as part of an OC4J 10*g* instance and compares that to how the Oracle WebCenter components are installed on a managed server when you install and configure the Oracle Fusion Middleware 11*g*.

Note that the WebCenter system components require a set of WebCenter schemas, which you install using the Repository Creation Utility (RCU). For more information about the schemas installed by RCU, refer to the *Oracle Fusion Middleware Repository Creation Utility User's Guide*.

Figure 3–4   Oracle WebCenter Installed on OC4J and on Oracle WebLogic Server
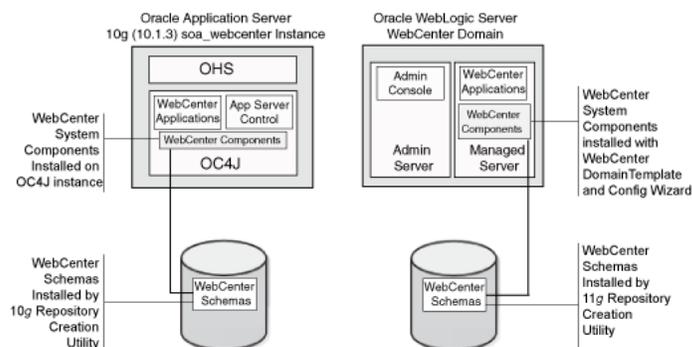
3.3.3 Comparing the Oracle WebCenter Directory Structure on OC4J and Oracle WebLogic Server

When you install and configure Oracle WebCenter, you create an Oracle WebLogic Server domain that has been extended to support the WebCenter system components. The resulting files on disk consist of the WebCenter domain directory structure, as well as the WebCenter Oracle home, which contains the WebCenter binary and support

files. The WebCenter Oracle home, like the SOA Oracle home, must reside inside the Middleware Oracle home.

This is contrast to the single Oracle home that was created during an Oracle Application Server 10*g* Release 3 (10.1.3) installation.

Figure 3–5 shows the directory structure that results from installing and configuring the Oracle SOA Suite domain.

Figure 3–5   The Directory Structure of an Oracle Fusion Middleware WebCenter Installation

4
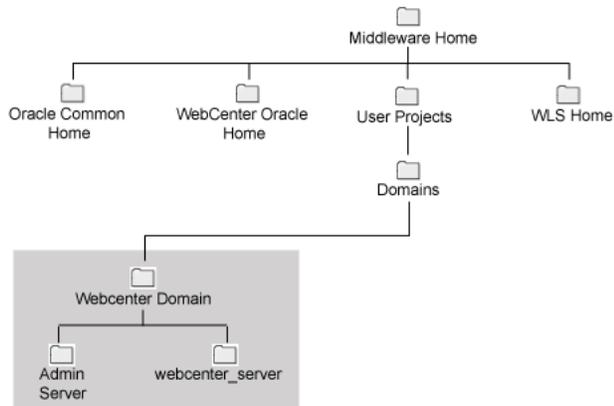
Special Considerations for Oracle BPEL Long-Running Processes

Before you begin the upgrade of your Oracle SOA Suite 11*g* environment, use the information in this chapter to determine whether or not you need to consider the impact of the upgrade on Oracle BPEL Process Manager long-running processes:

- What Is an Oracle BPEL Process Manager Long-Running Process?
- Two Approaches to Upgrading Long-Running Processes

4.1 What Is an Oracle BPEL Process Manager Long-Running Process?

Like any other BPEL process, a long-running process involves accessing a service and returning a value or result to Oracle BPEL Process Manager. However, the difference is that a long-running process may not return with a value for a relatively long period of time.

For example, a process designed to invoke an external task to process a customer order invokes an asynchronous backend application to start processing the order. The backend application interacts with a human for approval of the order. After the order is approved, the backend application performs a callback to the process, which performs a callback to the client application. This type of complex process is an example of a long-running process.

Long-running processes can affect your upgrade planning because you must consider the timing of shutting down your existing Oracle BPEL Process Manager 10*g* environment and starting up your Oracle BPEL Process Manager 11*g* environment. Ideally, you want all your long-running processes to finish successfully before you shut down and decommission your existing Oracle BPEL Process Manager 10*g* environment and start using your 11*g* environment.

4.2 Two Approaches to Upgrading Long-Running Processes

Oracle SOA Suite supports a wide variety of organizations and companies. As a result, it is difficult to determine the best upgrade strategy for all environments. However, in the area of long-running processes, there are two possible solutions you can consider. Note that both the solutions discussed here may not be suitable to all customers. If you are unclear about the best possible strategy for your particular organization, contact Oracle Support for more information about other possible alternatives to these possible solutions:

- Using a Side-by-Side Approach to Long-Running Processes During Upgrade
- Using a Grace Period and a Hard Stop of Long-Running Processes

4.2.1 Using a Side-by-Side Approach to Long-Running Processes During Upgrade

The side-by-side approach to long-running processes involves upgrading to an Oracle Fusion Middleware 11*g* environment while keeping the Oracle Application Server 10*g* environment up and running until the long-running processes finish up their tasks.

Table 4–1 shows a summary of the upgrade process you would use if you were considering the side-by-side approach to supporting Oracle BPEL Process Manager long running processes during upgrade.

Table 4–1   Steps for Using the Side-by-Side Approach to Long-Running Processes

Step Number	Task	More Information
1	Install and configure a new Oracle Fusion Middleware 11*g* environment.	Chapter 6, "Upgrading an Oracle SOA Suite 10g Environment"
2	Upgrade the Application in Oracle JDeveloper 11*g*.	Chapter 8, "Overview of Upgrading Oracle SOA Suite, WebCenter, and ADF Applications"
3	Deploy the application to Oracle Fusion Middleware 11*g*.	Section 8.9, "Task 9: Deploy the Application"
4	Make a copy of any client applications, upgrade them, and test them with the application you deployed on Oracle Fusion Middleware 11*g*.	Section 14.5, "Upgrading Oracle SOA Suite Client Applications"
5	Retire all processes on Oracle BPEL Process Manager 10*g*; this will stop any new processes from starting, but allow existing 10*g* processes to finish.	"How to Manage a Process Life Cycle" in the Oracle Application Server 10*g* Release 3 (10.1.3.1.0) *Oracle SOA Suite Developer's Guide*, available on the Oracle Technology Network (OTN): `http://www.oracle.com/technology/documentation`
6	With the 10*g* system still running, decommission the 10*g* client applications, and start the upgraded application clients, which are directing requests to the new 11*g* Oracle BPEL Process Manager.	"Starting and Stopping Oracle Fusion Middleware" in the *Oracle Fusion Middleware Administrator's Guide*
7	Monitor the 10*g* long-running processes until they all complete.	"Using the Instances Tab to Monitor Process Instances" in the Oracle Application Server 10*g* Release 3 (10.1.3.1.0) *Oracle SOA Suite Developer's Guide*, available on the Oracle Technology Network (OTN): `http://www.oracle.com/technology/documentation`
8	Shut down and decommission Oracle BPEL Process Manager 10*g*.	■ "Starting and Stopping Oracle BPEL Process Manager Components" in the Oracle Application Server 10*g* Release 3 (10.1.3.1.0) *Oracle BPEL Process Manager Quick Start Guide* ■ "Starting and Stopping Application Server Instances" in the in the Oracle Application Server 10*g* Release 3 (10.1.3.1.0) *Oracle Application Server Administrator's Guide*. The Oracle Application Server 10*g* Release 3 (10.1.3) documentation is available on the Oracle Technology Network (OTN): `http://www.oracle.com/technology/documentation`

4.2.2 Using a Grace Period and a Hard Stop of Long-Running Processes

As an alternative to the side-by-side approach to upgrading Oracle BPEL Process Manager long-running processes, you might consider the approach described in this section if you do not have many long-running processes, your long-running processes are less critical to your business model, or if you have more flexibility in terms of the timing or duration of the upgrade.

This alternative, grace-period approach can also be used if the long-running processes cannot complete until they receive client messages. For example, if the client sends a message to start the process and later sends another message in the middle of the process, then in the side-by-side approach, the process will never finish. This is because the processes rely on clients that have been upgraded to 11*g*. As a result, the 10g system will never receive the second client message.

Table 4–2 provides a summary of the upgrade process you would use if you were considering an alternative approach to the side-by-side approach to long-running processes.

Table 4–2   Steps for Using the Grace Period and a Hard Stop Approach to Long-Running Processes

Step Number	Task	More Information
1	Install and configure a new Oracle Fusion Middleware 11*g* environment.	Chapter 6, "Upgrading an Oracle SOA Suite 10g Environment"
2	Upgrade the Application in Oracle JDeveloper 11*g*.	Chapter 8, "Overview of Upgrading Oracle SOA Suite, WebCenter, and ADF Applications"
3	Deploy the application to Oracle Fusion Middleware 11*g*.	Section 8.9, "Task 9: Deploy the Application"
4	Make a copy of any client applications, upgrade them, and test them with the application you deployed on Oracle Fusion Middleware 11*g*.	Section 14.5, "Upgrading Oracle SOA Suite Client Applications"
5	Retire all processes on Oracle BPEL Process Manager 10*g*; this will stop any new processes from starting, but allow existing 10*g* processes to finish.	"How to Manage a Process Life Cycle" in the Oracle Application Server 10*g* Release 3 (10.1.3.1.0) *Oracle SOA Suite Developer's Guide*, available on the Oracle Technology Network (OTN): `http://www.oracle.com/technology/documentation`
6	Define a grace period and set a deadline for stopping the Oracle Application Server 10*g* environment and cutover to the Oracle Fusion Middleware 11*g* environment.	Use the Oracle Application Server 10*g* Oracle BPEL Control to monitor you long-running processes; use this information to define a reasonable time for the upgrade to Oracle Fusion Middleware 11*g*.

Table 4–2 (Cont.) Steps for Using the Grace Period and a Hard Stop Approach to Long-Running Processes

Step Number	Task	More Information
7	At the predefined deadline, abort all existing Oracle BPEL Process Manager process instances.	You can abort a process in one of two ways: ■ By deleting the instance from the BPEL Control. For more information, see "Using the Instances Tab to Monitor Process Instances" in the Oracle Application Server 10*g* Release 3 (10.1.3.1.0) *Oracle SOA Suite Developer's Guide,* available on the Oracle Technology Network (OTN): `http://www.oracle.com/technology/documentation` ■ By creating a Java program that locates and aborts all active instances. For this you can use the `abort()` method in the method in the interface `com.oracle.bpel.client.IInstanceHandle`. For more information, see the 10*g* Release 3 (10.1.3) *Oracle BPEL Process Manager Client Java API Reference.*
8	Shut down and decommission the Oracle Application Server 10*g* environment.	■ "Starting and Stopping Oracle BPEL Process Manager Components" in the Oracle Application Server 10*g* Release 3 (10.1.3.1.0) *Oracle BPEL Process Manager Quick Start Guide* ■ "Starting and Stopping Application Server Instances" in the in the Oracle Application Server 10*g* Release 3 (10.1.3.1.0) *Oracle Application Server Administrator's Guide.* The Oracle Application Server 10*g* Release 3 (10.1.3) documentation is available on the Oracle Technology Network (OTN): `http://www.oracle.com/technology/documentation`
9	Start and roll out the new upgraded client applications, which send requests to the new Oracle BPEL Process Manager 11*g*.	"Starting and Stopping Oracle Fusion Middleware" in the *Oracle Fusion Middleware Administrator's Guide*

Part II

Upgrading the Oracle SOA Suite, WebCenter, and ADF Environment

Part II contains the following chapters:

5

Upgrading an Oracle ADF 10*g* Environment

This chapter describes how to upgrade your Oracle Application Server 10*g* Oracle ADF environment to an equivalent environment in Oracle Fusion Middleware 11*g*.

Refer to the following sections for more information:

- Task 1: Decide Upon an Oracle Fusion Middleware 11g Topology for Your ADF Applications
- Task 2: Install and Configure a Oracle WebLogic Server Domain For Your ADF Applications

5.1 Task 1: Decide Upon an Oracle Fusion Middleware 11*g* Topology for Your ADF Applications

Before you begin the upgrade process, decide upon a topology for your new Oracle Fusion Middleware 11*g* environment.

For more information, see Section 3.1, "Information for Oracle ADF Users".

5.2 Task 2: Install and Configure a Oracle WebLogic Server Domain For Your ADF Applications

Before you upgrade to Oracle Fusion Middleware 11*g*, you must install an Oracle Fusion Middleware environment that is similar to the topology you set up for Oracle Application Server 10*g*.

Refer to the following sections for more information:

- Installing and Configuring a Development Domain with Oracle JDeveloper
- Installing and Configuring a Domain without JDeveloper
- Using the Java Required Files (JRF) Domain Template

5.2.1 Installing and Configuring a Development Domain with Oracle JDeveloper

Oracle WebLogic Server is available as part of Oracle JDeveloper Studio. As a result, if you are using Oracle JDeveloper Studio as your integrated development environment (IDE), you can install Oracle WebLogic Server as part of the Oracle JDeveloper installation.

For complete instructions for installing and configuring Oracle WebLogic Server with Oracle JDeveloper Studio, see "Installing the Oracle JDeveloper Studio Edition" in *Oracle Fusion Middleware Installation Guide for Oracle JDeveloper*.

5.2.2 Installing and Configuring a Domain without JDeveloper

If you are not using Oracle JDeveloper for developing your applications, then you can install the Oracle WebLogic Server software and the Oracle Fusion Middleware Application Developer software suite.

The Application Developer installer allows you to install the software you need to configure a domain with the Java Required Files (JRF) and Oracle Enterprise Manager Fusion Middleware Control domain templates. For information about the JRF template, see Section 5.2.3, "Using the Java Required Files (JRF) Domain Template".

For information about installing and configuring the Oracle Fusion Middleware Application Developer software, see the *Installation Guide for Application Developer*.

5.2.3 Using the Java Required Files (JRF) Domain Template

When you configure Oracle WebLogic Server, you configure each domain using domain templates. One of the domain templates available with Oracle Fusion Middleware 11*g* is the Java Required Files (JRF) template.

The JRF template provides important Oracle libraries and other capabilities that support new versions of APIs that many OC4J applications depend upon.

For information on the types of APIs in the JRF template that are important to upgraded OC4J applications, see "APIs Available With the Java Required Files (JRF) Domain Template" in the *Oracle Fusion Middleware Upgrade Guide for Java EE.*

To create or extend a domain using the JRF template, refer to the following:

- Creating a New Domain With the JRF Template

- Extending an Existing Domain With the JRF Template

- Upgrading Existing Oracle WebLogic Server Instances to the ADF 11g Runtime

5.2.3.1 Creating a New Domain With the JRF Template

There are multiple ways to create a new Oracle WebLogic Server domain using the JRF template:

- Install and configure a development domain using the Oracle JDeveloper 11*g* installer.

   The resulting domain is automatically created using the JRF template.

- Install and configure an Application Developer, Oracle SOA Suite, WebCenter, Oracle Identity Management, or Oracle Portal, Forms, Reports, and Discoverer domain.

   When you configure any Oracle Fusion Middleware software suite, you have the option of selecting the JRF template while running the configuration tool.

   For more information, refer to the appropriate Oracle Fusion Middleware installation guide.

5.2.3.2 Extending an Existing Domain With the JRF Template

To extend an existing domain with the JRF template, use one of the following methods:

- Use the Oracle JDeveloper 11*g* installer.

   In the Oracle JDeveloper installer, select a custom installation and select the ADF Runtime component. This step allows you to install the ADF runtime jar files and domain templates to the server environment.

For more information, see the *Oracle Fusion Middleware Installation Guide for Oracle JDeveloper*.

- Run the Fusion Middleware configuration wizard from an Oracle SOA Suite, WebCenter, Oracle Identity Management, or Oracle Portal, Forms, Reports, and Discoverer Oracle home.

  Select the options to extend the domain and select the JRF template when prompted with the list of available templates.

  For more information, refer to the appropriate Oracle Fusion Middleware installation guide.

- Use Oracle Enterprise Manager Fusion Middleware Control or the ApplyJRF WebLogic Scripting Tool (WLST) command to apply the JRF template to an existing WebLogic server instance.

  For more information, see "Applying Oracle JRF Template to a Managed Server or Cluster," in the *Oracle Fusion Middleware Administrator's Guide*.

5.2.3.3 Upgrading Existing Oracle WebLogic Server Instances to the ADF 11*g* Runtime

A version of Oracle JDeveloper 11*g* (11.1.1.0.x) was made available to Oracle customers before the release of Oracle Fusion Middleware 11*g*.

If you have already installed and configured an Oracle WebLogic Server domain using the Oracle JDeveloper 11*g* (11.1.1.0.x), then you should be aware that Oracle Fusion Middleware 11*g* and the new Oracle JDeveloper 11*g* (11.1.1.1.0) provides a newer version of the Oracle ADF runtime software.

To upgrade your existing Oracle WebLogic Server domain with the Oracle ADF 11*g* runtime software:

1. Open the uninstaller for of the Oracle WebLogic Server instance you want to upgrade.

   You can find the Oracle WebLogic Server uninstaller in the following location:

   On UNIX systems:

   ```
   WLS_HOME/wlserver_10.3/uninstall/uninstall.sh
   ```

   On Windows systems:

   ```
   WLS_HOME\wlserver_10.3\uninstall\uninstall.cmd
   ```

   Note that on a headless server (such as a Linux or UNIX-based system), you may have to set the DISPLAY variable to an X Server like VNC or Xming.

2. Select JDeveloper 11g (11.1.1.0.x) Studio Edition 11.1.1.0.0.

3. Deselect everything under WebLogic Server.

4. Click **Next** to complete the uninstall. If you're interested you can click on the Details button to see what was uninstalled.

5. Open the JDeveloper 11.1.1.1.0 installer.

6. Select the Oracle WebLogic Server instance where you want to install the new version of the ADF runtime.

7. Deselect everything, and then select only **Application Development Framework Runtime.**

Note that this will also automatically select the JDeveloper and ADF folders.

8. To avoid class loading issues, rebuild all your EAR files.

6

Upgrading an Oracle SOA Suite 10*g* Environment

This chapter describes how to upgrade your Oracle SOA Suite environment from Oracle Application Server 10*g* to Oracle Fusion Middleware 11*g*.

Upgrading this environment and your applications includes the following tasks:

- Task 1: Decide Upon an Oracle SOA Topology
- Task 2: If Necessary, Upgrade the Oracle Database That Hosts the Oracle SOA Suite Schemas
- Task 3: Use the Repository Creation Utility to Install the 11g SOA Schemas
- Task 4: Install and Configure the Oracle SOA Suite 11g Middle Tiers
- Task 5: If Necessary, Upgrade the SOA Component Schemas
- Task 6: Configure Any Required Java EE Features to Support Your SOA Applications
- Task 7: Perform Any Required Post-Upgrade Tasks for the SOA Components

6.1 Task 1: Decide Upon an Oracle SOA Topology

Before you begin the upgrade process, decide upon a topology for your new Oracle Fusion Middleware 11*g* environment.

For more information, see Section 3.2, "Information for Oracle SOA Suite Users".

Note that the instructions in this guide provide the steps for upgrading a basic Oracle SOA Suite environment. After reviewing the content of this guide, you should be able to transfer this knowledge to a more complex production environment.

For specific information about setting up a production environment that takes advantage of clustering and high availability, refer to the *Oracle Fusion Middleware Enterprise Deployment Guide for Oracle SOA Suite*.

For an overview of the high-level tasks you must perform when upgrading an enterprise deployment, see "Planning an Enterprise Deployment" in the *Oracle Fusion Middleware Upgrade Planning Guide*.

6.2 Task 2: If Necessary, Upgrade the Oracle Database That Hosts the Oracle SOA Suite Schemas

If you are upgrading an Oracle SOA Suite environment, then you must ensure that the version of the database that will contain the Oracle SOA Suite schemas is supported by Oracle Fusion Middleware 11*g*.

You can install a new database, or upgrade your existing database to a supported version.

Note that if your upgrade involves multiple databases, then you might have to upgrade both the source database and the target database before proceeding with the upgrade. For example, the Oracle BAM upgrade can involve separate source and target databases.

For instructions on verifying that your database meets the requirements of Oracle Fusion Middleware 11*g*, see "Upgrading and Preparing Your Databases" in the *Oracle Fusion Middleware Upgrade Planning Guide*.

6.3 Task 3: Use the Repository Creation Utility to Install the 11*g* SOA Schemas

If you are upgrading Oracle SOA Suite, use the Repository Creation Utility to install the schemas in the database.

Repository Creation Utility is a graphical and command-line tool used to create and manage Oracle Fusion Middleware database schemas in your database.

For more information, refer to the following sections:

- Running the Repository Creation Utility in Preparation for an Oracle SOA Suite Upgrade
- Selecting the Schemas Required for a Oracle SOA Suite Upgrade

6.3.1 Running the Repository Creation Utility in Preparation for an Oracle SOA Suite Upgrade

To start the Repository Creation Utility (RCU), you run the `rcu` command, which is in the root directory of the Repository Creation Utility CD-ROM or (if you have already installed the Oracle SOA Suite software on disk) in the bin directory of the Oracle SOA Suite Oracle home.

For more detailed instructions on how to obtain and run RCU, refer to the *Oracle Fusion Middleware Repository Creation Utility User's Guide*.

For more information about which schemas to select when running the Repository Creation Utility, refer to Section 6.3.2, "Selecting the Schemas Required for a Oracle SOA Suite Upgrade".

For more information about the Oracle SOA Suite schemas that are installed by the Repository Creation Utility, refer to "Metadata Repository Schemas" in the *Oracle Fusion Middleware Administrator's Guide*.

6.3.2 Selecting the Schemas Required for a Oracle SOA Suite Upgrade

You can use the Repository Creation Utility to install the schemas required for all of the Oracle Fusion Middleware software components that require a schema. However, there is no need to install all the schemas unless you plan to install a complete Oracle

Fusion Middleware environment and you plan to use the same database for all the Oracle Fusion Middleware component schemas.

For a Oracle SOA Suite upgrade, you must select the following schemas when you prompted by the Repository Creation Utility (as shown in Figure 6–1):

- Select the **Metadata Services** schema in the **AS Common Schemas** category.

  This schema supports Oracle Fusion Middleware Metadata Services (MDS), which is required by some Oracle Fusion Middleware components, as well as for custom applications that support MDS. For more information, see "Understanding the MDS Repository" in the *Oracle Fusion Middleware Administrator's Guide*.

- Select the entire **SOA Infrastructure** category of schemas.

  These schemas are required for the Oracle SOA Suite components.

Figure 6–1   Selecting the Required Schemas for a Oracle SOA Suite Upgrade
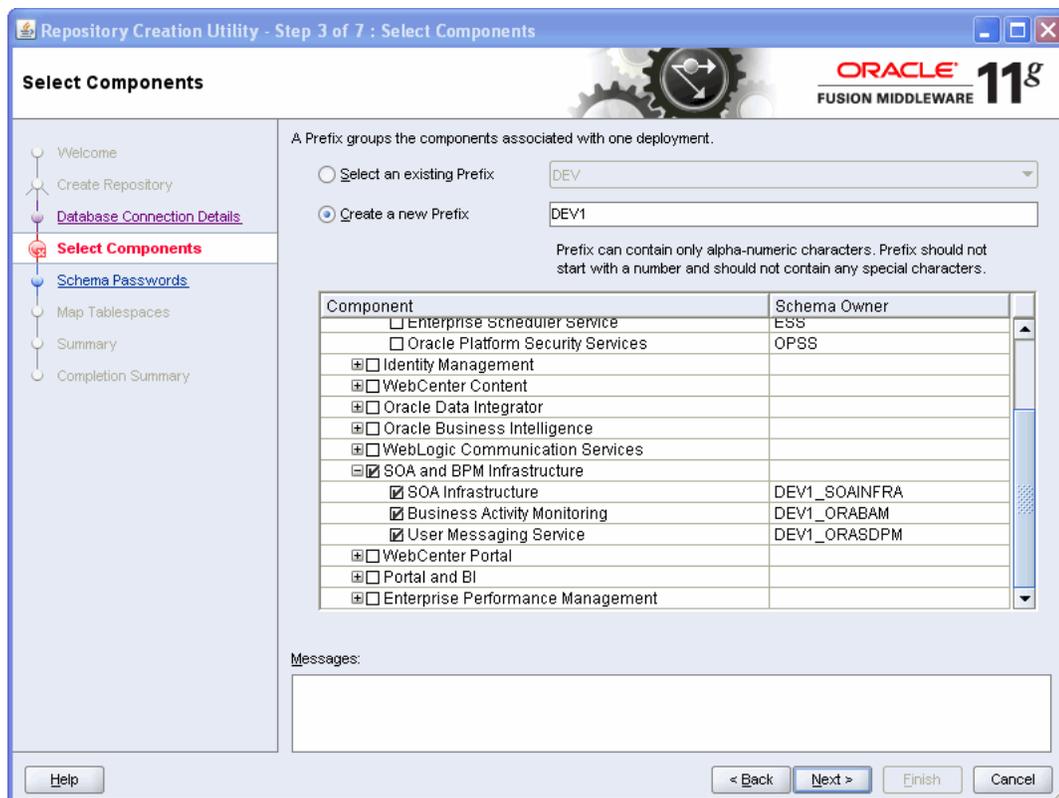
6.4 Task 4: Install and Configure the Oracle SOA Suite 11*g* Middle Tiers

Before you upgrade to Oracle Fusion Middleware 11*g*, you must install and configure an Oracle Fusion Middleware 11*g* environment that is similar to the topology you set up for Oracle Application Server 10*g*.

Refer to the following documentation resources for complete instructions on installing and configuring the Oracle SOA Suite middle tiers:

- The *Oracle Fusion Middleware Installation Planning Guide*

- The *Oracle Fusion Middleware Quick Installation Guide for Oracle SOA Suite and Oracle Business Process Management Suite*

- The *Oracle Fusion Middleware Installation Guide for Oracle SOA Suite and Oracle Business Process Management Suite*

Table 6–1 provides a high-level summary of the main steps you must perform when installing and configuring the Oracle SOA Suite middle tiers, as well as specific references to the procedures you must follow for each step in the *Oracle Fusion Middleware Quick Installation Guide for Oracle SOA Suite and Oracle Business Process Management Suite*.

Note that the procedures described in this section assume you have downloaded the latest version of Oracle WebLogic Server and Oracle Fusion Middleware 11*g*. For more information, refer to "Obtaining the Latest Oracle WebLogic Server and Oracle Fusion Middleware 11*g* Software" in the *Oracle Fusion Middleware Upgrade Planning Guide*.

Table 6–1   Main Steps Required to Install and Configure the Oracle SOA Suite *Middle Tiers*

Step #	Description	Section	Book
1	Install Oracle WebLogic Server and create the Middleware home	"Running the Installation Program in Graphical Mode"	*Oracle Fusion Middleware Installation Guide for Oracle WebLogic Server*
2	Install the Oracle SOA Suite software and create the Oracle SOA Suite Oracle home	"Installing Oracle SOA Suite and Oracle Business Process Management Suite"	*Oracle Fusion Middleware Installation Guide for Oracle SOA Suite and Oracle Business Process Management Suite*
3	Configure the Oracle SOA Suite domain	"Configuring Oracle SOA Suite"	*Oracle Fusion Middleware Installation Guide for Oracle SOA Suite and Oracle Business Process Management Suite*
4	Start the Oracle WebLogic Server domain	"Starting the Servers"	*Oracle Fusion Middleware Installation Guide for Oracle SOA Suite and Oracle Business Process Management Suite*
5	Verify the installation	"Verifying the Installation and Configuration"	*Oracle Fusion Middleware Installation Guide for Oracle SOA Suite and Oracle Business Process Management Suite*

6.5 Task 5: If Necessary, Upgrade the SOA Component Schemas

If you are upgrading the Oracle B2B or Oracle BAM components, then you must run the Oracle Fusion Middleware Upgrade Assistant to upgrade these component schemas to 11*g*.

The Oracle Fusion Middleware Upgrade Assistant automates the upgrade of many aspects of your Oracle Application Server 10*g* environment.

The Upgrade Assistant is installed automatically into the `bin` directory of your Oracle Fusion Middleware Oracle home. You run the Upgrade Assistant once for each 10*g* schema you are upgrading.

> **Note:**   If you are upgrading a 10*g* Release 3 (10.1.3.1.0) Oracle home that was installed using the Basic Installation option, then there is no need to run the Upgrade Assistant. When you select the 10*g* Release 3 (10.1.3.1.0) Basic Installation option, the installer does not install any components that can be upgraded by the Oracle Fusion Middleware Upgrade Assistant.

Refer to the following sections for more information:

- Task 5a: Start the Upgrade Assistant for an Oracle SOA Schema Upgrade
- Task 5b: If Necessary, Upgrade the B2B Schema
- Task 5c: If Necessary, Upgrade the BAM Schema

6.5.1 Task 5a: Start the Upgrade Assistant for an Oracle SOA Schema Upgrade

To start the Upgrade Assistant using the graphical user interface:

> **Note:** You can also use the Upgrade Assistant command-line interface to upgrade your Oracle Application Server 10*g* Oracle homes. For more information, see "Using the Upgrade Assistant Command-Line Interface" in the *Oracle Fusion Middleware Upgrade Planning Guide*.

1. Change directory the *ORACLE_HOME*/bin directory of the Oracle SOA Suite installation.

2. Enter the following command to start the Upgrade Assistant.

   On UNIX system:

   ```
   ./ua
   ```

   On Windows systems:

   ```
   ua.bat
   ```

   The Upgrade Assistant displays the Welcome screen as shown in Figure 6–2.

Figure 6–2   Upgrade Assistant Welcome Screen

3. Click **Next** to display the Specify Operation screen (Figure 6–3).

The options available in the Upgrade Assistant are specific to the Oracle home from which it started. When you start Upgrade Assistant from an Oracle SOA Suite and WebCenter Oracle home, the options shown on the Select Operation screen are the valid options for an Oracle SOA Suite and WebCenter Oracle home.

Figure 6–3   Upgrade Assistant Specify Operation Screen for a SOA Upgrade
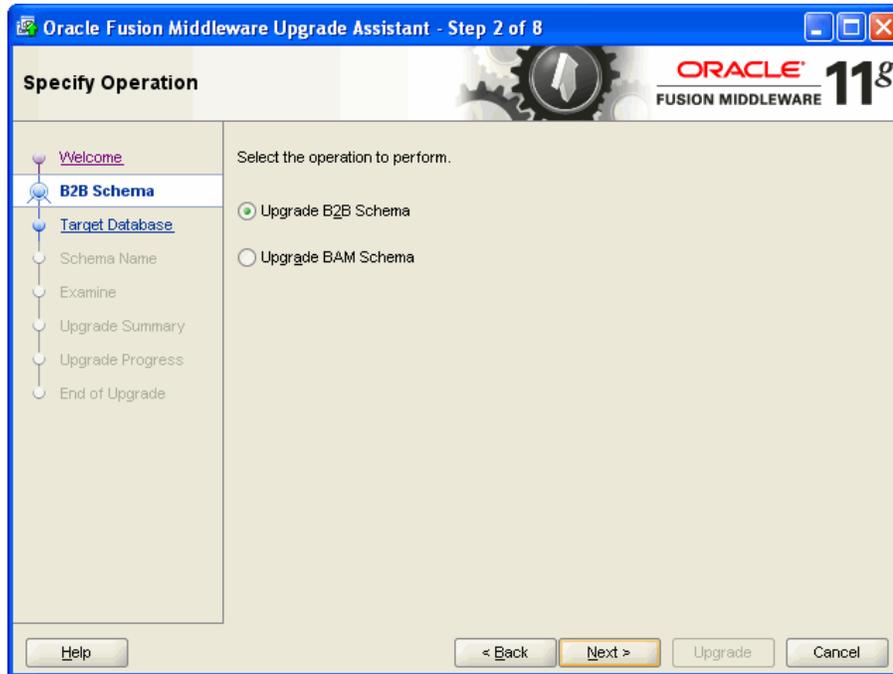
6.5.2 Task 5b: If Necessary, Upgrade the B2B Schema

Refer to the following sections for information about upgrading the Oracle B2B schema:

- General Information About the B2B Schema Upgrade

- Applying the Required Patch Set to the Oracle B2B 10g Middle Tier

- Setting the Maximum Message Size Attribute In the Oracle WebLogic Server Administration Console

- Performing the B2B Schema Upgrade

6.5.2.1 General Information About the B2B Schema Upgrade

If you are using Oracle Application Server 10*g* Release 2 (10.1.2) and you are upgrading Oracle Application Server Integration B2B, then you will need to upgrade the B2B schema in your database.

> **See Also:**   *Oracle Fusion Middleware User's Guide for Oracle B2B* in the Oracle Application Server 10*g* Release 2 (10.1.2) Documentation Library on the Oracle Technology Network:
>
> http://www.oracle.com/technology/documentation/

When you upgrade the B2B schema from 10*g* Release 2 (10.1.2), you must apply a software patch to your 10*g* Release 2 (10.1.2) installation, and then export the B2B 10*g* schema from your database. The exported data is saved to a ZIP file on disk. Then, you

can import the data into the database that contains your Oracle Fusion Middleware 11*g* component schemas.

6.5.2.2 Applying the Required Patch Set to the Oracle B2B 10*g* Middle Tier

Before you can perform the Oracle B2B schema upgrade, you must apply the a required patch set to the Oracle Application Server Integration B2B 10*g* Release 2 (10.1.2) middle tier.

This patch set is required in order to enable the Java command that you use to export the trading partner agreements, as described in the following step.

For specific information about the patch set requirements for upgrade and interoperability with previous Oracle Application Server releases, see the *Oracle Fusion Middleware Release Notes*.

6.5.2.3 Setting the Maximum Message Size Attribute In the Oracle WebLogic Server Administration Console

The exported customer data from your Oracle B2B 10*g* schema can result in a very large ZIP file on disk. To be sure that the large file can be imported successfully into the Oracle Fusion Middleware 11*g* metadata repository database, you should modify the default **Maximum Message Size** general protocol attribute for the WebLogic managed server where the SOA Suite components are deployed.

To perform this task:

1. Log in to the Oracle WebLogic Server Administration Console:

   ```
   http://localhost:7001/console
   ```

2. In the left pane of the Console, expand **Environment** and select **Servers**.

3. On the Summary of Servers page, select the server instance you want to configure. In most cases, the managed server that hosts the SOA components is called `soa_server1`.

4. Click **Protocols** to display the general connection settings for network protocols.

5. Scroll down to the **Maximum Message Size** property and set it to `200000000`.

6.5.2.4 Performing the B2B Schema Upgrade

> **Note:** This procedure assumes you have installed and configured your new Oracle Fusion Middleware 11*g* environment, as described in Section 6.4, "Task 4: Install and Configure the Oracle SOA Suite 11g Middle Tiers"

To upgrade the B2B schema:

1. Apply the required patch set to the Oracle Application Server Integration B2B 10*g* Release 2 (10.1.2) middle tier.

   For more information, see Section 6.5.2.2, "Applying the Required Patch Set to the Oracle B2B 10g Middle Tier".

2. Export the 10*g* Release 2 (10.1.2) trading partner agreements to an external ZIP file by using the following procedure:

a. Export the 10*g* Release 2 (10.1.2) trading partner agreements that you have deployed by running the following command in the Oracle Application Server Integration B2B Oracle home:

```
java -ms1024M -mx1024M oracle.tip.repos.export.Export 11g
```

This Java command creates the following directory, which contains the exported data:

On Windows systems:

```
%TEMP%/export.zip
```

On UNIX systems:

```
/tmp/export.zip
```

b. Create the following directory in the Oracle Fusion Middleware 11*g* Oracle home, which can be on the same host or on a different host from the 10*g* Oracle home:

On Windows systems:

```
ORACLE_HOME\b2b\upgrade
```

On UNIX systems:

```
ORACLE_HOME/b2b/upgrade
```

c. Copy the `export.zip` file to the directory you just created.

3. Use the Oracle WebLogic Server Administration Console to increase the **Maximum Message Size** property for the SOA component managed server (for example, `soa_server1`).

For more information, see Section 6.5.2.3, "Setting the Maximum Message Size Attribute In the Oracle WebLogic Server Administration Console".

4. Stop the administration server and the managed servers in the Oracle WebLogic Server domain that hosts the BAM application.

For more information, see "Starting and Stopping Oracle Fusion Middleware" in the *Oracle Fusion Middleware Administrator's Guide*.

5. Start the Upgrade Assistant as described in Task 5a: Start the Upgrade Assistant for an Oracle SOA Schema Upgrade.

6. Select **Upgrade B2B Schema** on the Specify Operation screen (Figure 6–3).

7. Refer to Table 6–2 for a description of the Upgrade Assistant screens that require input from you during a B2B Schema upgrade and the options on each screen.

Note that you are not prompted for the location of the ZIP file that contains the Oracle B2B 10*g* schema and customer data. Instead, the Upgrade Assistant automatically locates the ZIP file in the location specified in Step 2.

8. After the Specify Target Database screen, the Upgrade Assistant performs the following tasks and provides the progress on each task:

- Examines the components and schemas to be upgraded and verifies that they can be upgraded successfully.

- Provides a summary of the components to be upgraded so you can verify that Upgrade Assistant is upgrading the components and schemas you expect.

- Provides a progress screen so you can see the status of the upgrade as it proceeds.

- Alerts you of any errors or problems that occur during the upgrade.

    See Also: "Troubleshooting Your Upgrade" in the *Oracle Fusion Middleware Upgrade Planning Guide* for specific instructions for troubleshooting problems that occur while running the Upgrade Assistant

- Displays the End of Upgrade screen, which confirms that the upgrade was complete.

9. When the Oracle B2B schema upgrade is complete, start the administration server and Oracle SOA Suite managed servers in the Oracle WebLogic Server domain where Oracle B2B 11*g* is running.

Table 6–2   Upgrade Assistant Screens That Require Input During a B2B Schema Upgrade

Upgrade Assistant Screen	Description
Specify Target Database	Use this screen to enter the connection details for the database where you installed the Oracle B2B 11*g* schema with the Repository Creation Utility.
	■ **Target Database Type** - Select Oracle from this drop-down list to upgrade the Oracle B2B schema to an Oracle database.
	■ **Connect String** - Enter the connection string in the following format:
	`host:port:SID`
	For example:
	`host3:1521:soaDB3`
	■ **User Name** - Enter the database user name; if you enter `SYS`, be sure to enter the string, `SYS as SYSDBA`.
	■ **Password** - The password for the database user.
Specify Schema Name	Use this screen to select the Oracle B2B 11*g* schema name and to enter the password for the schema.
	This is the password you defined when you were installing the B2B schema with the 11*g* Repository Creation Utility.

6.5.3 Task 5c: If Necessary, Upgrade the BAM Schema

To upgrade the Oracle BAM schema, refer to the following sections:

- Understanding the Oracle BAM Schema Upgrade

- About Upgrading From Previous Versions of Oracle BAM

- Applying the Required Patch Set to the Oracle Database That Contains the Oracle BAM Schema

- Performing the Oracle BAM Schema Upgrade

- Recovering From and Retrying an Interrupted Oracle BAM Schema Upgrade

- Locating and Configuring the Oracle BAM Upgrade Log Files

6.5.3.1 Understanding the Oracle BAM Schema Upgrade

If you are upgrading Oracle Business Activity Monitoring, then you will need to upgrade the BAM schema in your database.

Note the following about the Oracle BAM schema upgrade:

- This procedure assumes you have already run the Repository Creation Utility and created the new Oracle BAM 11*g* schema in a supported database. When you create the Oracle BAM 11*g* schema it does not overwrite the existing Oracle BAM 10*g* schema.

- To upgrade the Oracle BAM schema, you must identify a database that contains the Oracle BAM 10*g* schema and a database that contains the Oracle BAM 11*g* schema.

- Note that the 10*g* schema and the 11*g* schema can reside in the same database or in different databases; in either case, you must identify the source and the target database.

6.5.3.2 About Upgrading From Previous Versions of Oracle BAM

If you are using Oracle Application Server 10*g* Release 2 (10.1.2) and you want to upgrade Oracle Business Activity Monitoring to Oracle Fusion Middleware 11*g*, then you must first upgrade to Oracle Application Server 10*g* Release 3 (10.1.3).

Refer to Section 3.5, "Upgrading an Installation" in the *Oracle Business Activity Monitoring Installation Guide* in the Oracle Application Server 10*g* Release 3 (10.1.3.1.0) documentation library on the Oracle Technology Network:

`http://www.oracle.com/technology/`

6.5.3.3 Applying the Required Patch Set to the Oracle Database That Contains the Oracle BAM Schema

Before you can upgrade the Oracle BAM schema, you must apply a patch set to the Oracle Database that contains the Oracle BAM 10*g* schemas.

For specific information about the patch set requirements for upgrade and interoperability with previous Oracle Application Server releases, see the *Oracle Fusion Middleware Release Notes*.

6.5.3.4 Performing the Oracle BAM Schema Upgrade

To upgrade the BAM schema:

1. Stop the administration server and the managed servers in the Oracle WebLogic Server domain that hosts the BAM application.

    For more information, see "Starting and Stopping Oracle Fusion Middleware" in the *Oracle Fusion Middleware Administrator's Guide*.

2. Start the Upgrade Assistant as described in Task 5a: Start the Upgrade Assistant for an Oracle SOA Schema Upgrade.

3. Select **BAM Schema** on the Specify Operation screen (Figure 6–3).

4. Refer to Table 6–3 for a description of the Upgrade Assistant screens that require input from you during a Oracle BAM schema upgrade and the options on each screen.

5. After the Specify Schema Name screen, the Upgrade Assistant performs the following tasks and provides the progress on each task:

- Examines the components and schemas to be upgraded and verifies that they can be upgraded successfully.

- Provides a summary of the components to be upgraded so you can verify that Upgrade Assistant is upgrading the components and schemas you expect.

- Provides a progress screen so you can see the status of the upgrade as it proceeds.

- Alerts you of any errors or problems that occur during the upgrade.

   See Also: "Troubleshooting Your Upgrade" in the *Oracle Fusion Middleware Upgrade Planning Guide* for specific instructions for troubleshooting problems that occur while running the Upgrade Assistant

- Displays the End of Upgrade screen, which confirms that the upgrade was complete.

Table 6–3   Upgrade Assistant Screens That Require Input During a BAM Schema Upgrade

Upgrade Assistant Screen	Description
Specify Source Database	Use this screen to enter the database connection details for the database that contains the Oracle BAM 10*g* schema.
Specify Target Database Details	Use this screen to enter the database connection details for the database that contains the Oracle BAM 11*g* schema that you created with the Oracle Fusion Middleware Repository Creation Utility.
Specify Schema Name	Use this screen to select the Oracle BAM 11*g* schema and enter the password for the BAM schema. This is the Oracle BAM schema password that you set in the Repository Creation Utility when you installed the schema in the database.
Examining Components	This page shows the status of examination phase, where the upgrade assistant verifies the source and target schema.

6.5.3.5 Recovering From and Retrying an Interrupted Oracle BAM Schema Upgrade

If the Oracle BAM schema upgrade is interrupted for any reason, you might have to perform some recovery steps before you can retry the upgrade procedure.

   Note: The procedure for upgrading the Oracle Business Activity Monitoring (BAM) schema is documented in the *Oracle Fusion Middleware Upgrade Guide for Oracle SOA Suite, WebCenter Portal, and ADF*.

During a Oracle BAM schema upgrade, the Upgrade Assistant automatically performs several tasks for you, including exporting the Oracle BAM schema from the source database and importing it into the database that hosts your Oracle Fusion Middleware 11*g* schemas. Upgrade Assistant uses Oracle Data Pump to perform the export and import operations.

If the Oracle BAM schema upgrade is interrupted (for example, if you exit the Upgrade Assistant during the upgrade, or if the database goes down), then you can recover from the upgrade interruption by using the following instructions:

1. Connect to the database that hosts the Oracle BAM 11*g* schemas.

Use the same database user and password that you used when connected to the database from the Upgrade Assistant.

For example:

```
connect sys/mypassword as sysdba
```

2. Enter the following SQL command to identify whether or not any Oracle Data Pump jobs are currently running:

```
SELECT * FROM DBA_DATAPUMP_JOBS;
```

3. If any Oracle Data Pump jobs are running, then wait until the job complete before proceeding to the next step.

4. Enter the following SQL command to drop a specific database table that was created during the export process.

```
DROP TABLE ORACLEBAM1013_EXPORT;
```

If the table exists, you cannot retry the upgrade, because the upgrade procedure will not run successfully.

5. Run the Upgrade Assistant again and retry the Oracle BAM upgrade.

6.5.3.6 Locating and Configuring the Oracle BAM Upgrade Log Files

When you use the Upgrade Assistant to upgrade Oracle BAM, the Upgrade Assistant uses the Morpheus command line utility to perform the 11*g* schema upgrade and report conversions, so the upgraded 10g schema can be used in Oracle BAM 11*g*.

The Morpheus command-line utility is located in following folder in the Oracle SOA Suite 11*g* Oracle home:

- On Windows systems:

  ORACLE_HOME\bam\bin\morpheus.bat

- On UNIX systems:

  ORACLE_HOME/bam/bin/morpheus

When the BAM upgrade runs the morpheus utility, it logs all the Morpheus-related log messages in an upgrade output log file identified by an `.out` file extension, located in the `ORACLE_HOME/upgrade/logs` folder.

The upgrade log level can be configured by modifying the following configuration file:

ORACLE_HOME/bam/bin/morpheus.logging.properties

6.6 Task 6: Configure Any Required Java EE Features to Support Your SOA Applications

Besides the SOA system components, which are installed during the Oracle WebLogic Server installation, you might have to configure additional Oracle WebLogic Server features to support the SOA applications you deploy.

For example, you may have to define application-required data sources, JMS resources, or other services. The resources and services provided by the Oracle WebLogic Server domain are common to all Java-based applications, including Oracle SOA Suite applications.

For more information, see the *Oracle Fusion Middleware Upgrade Guide for Java EE*.

6.7 Task 7: Perform Any Required Post-Upgrade Tasks for the SOA Components

Depending upon the components you are upgrading, the following post-upgrade tasks might apply:

- Configuring Oracle SOA Suite 11g to Use Identity Management Components
- About Case Sensitivity in Oracle Human Workflow

6.7.1 Configuring Oracle SOA Suite 11*g* to Use Identity Management Components

If your Oracle Application Server10*g* environment relied on any Oracle Identity Management components, such as Oracle Internet Directory, or if you were using any third-party LDAP servers, then you can associate those components with your new Oracle Fusion Middleware 11*g* environment.

Specifically, you can use Oracle Enterprise Manager Fusion Middleware Control to associate an LDAP directory with Oracle Fusion Middleware, so you can use the LDAP directory as the credential or policy store for the Oracle WebLogic Server domain.

For more information, see the following:

- "Reassociating Domain Stores with Oracle Enterprise Manager" in the *Oracle Fusion Middleware Application Security Guide*.
- "Configuring LDAP Authentication Providers" in *Oracle Fusion Middleware Securing Oracle WebLogic Server*.

6.7.2 About Case Sensitivity in Oracle Human Workflow

Oracle Human Workflow 10*g* and Oracle Human Workflow 11*g* support different default behavior in terms of the case sensitivity of user names, group names, and application role names:

- In Oracle Human Workflow 11*g*:
  - By default, all user names in the Human Workflow component are case insensitive, and all user names are stored in lower case.
  - By default, all group names and application role names are always case sensitive.
- In Oracle Workflow 10*g* Release 3 (10.1.3), the system was case sensitive to user names, by default, and all user names were stored in same case as seeded.

To keep both environments compatible, you should set the case sensitivity to be the same in 11*g* as it was in 10*g* Release 3 (10.1.3).

User name case insensitivity can be changed in `workflow-identity-config.xml`. The `caseSensitive` property identifies whether or not user names are case sensitive. This flag should be upgraded using the MBean browser.

For more information, see "Case Sensitivity" in the *Oracle Fusion Middleware Developer's Guide for Oracle SOA Suite*.

7

Upgrading an Oracle WebCenter 10*g* Environment

This chapter describes how to upgrade your Oracle WebCenter environment from Oracle Application Server 10*g* to Oracle Fusion Middleware *11g*.

Upgrading this environment and your applications includes the following tasks:

- Task 1: Decide Upon an Oracle WebCenter Portal 11g Topology
- Task 2: If Necessary, Upgrade the Oracle Database That Hosts the WebCenter Schemas
- Task 3: Use the Repository Creation Utility to Install the Oracle WebCenter Portal 11g Schemas
- Task 4: Install and Configure the Oracle WebCenter Portal 11g Middle Tiers
- Task 5: Configure Any Required Java EE Features to Support Your WebCenter Applications
- Task 6: Upgrade Back-End Components for WebCenter Services

Note: This chapter uses recently rebranded Oracle WebCenter Portal product names where appropriate. For more information, see Section 3.3.1, "New Product Names for Oracle WebCenter and Oracle Enterprise Content Management Suite."

7.1 Task 1: Decide Upon an Oracle WebCenter Portal 11*g* Topology

Before you begin the upgrade process, decide upon a topology for your new Oracle Fusion Middleware 11*g* environment.

For more information, see Section 3.3, "Information for Oracle WebCenter Users".

7.2 Task 2: If Necessary, Upgrade the Oracle Database That Hosts the WebCenter Schemas

If you are upgrading an Oracle WebCenter environment, you must ensure that the version of the database where you plan to install the schemas is supported by Oracle Fusion Middleware 11*g*.

You can install a new database, or upgrade your existing database to a supported version.

For instructions on verifying that your database meets the requirements of Oracle Fusion Middleware 11*g*, see "Upgrading and Preparing Your Databases" in the *Oracle Fusion Middleware Upgrade Planning Guide*.

7.3 Task 3: Use the Repository Creation Utility to Install the Oracle WebCenter Portal 11*g* Schemas

If you are upgrading Oracle WebCenter, use the Repository Creation Utility to install the schemas in the database you identified and prepared in Task 2: If Necessary, Upgrade the Oracle Database That Hosts the WebCenter Schemas.

For more information, refer to the following sections:

- Running the Repository Creation Utility in Preparation for an Oracle WebCenter Upgrade
- Selecting the Schemas Required for Oracle WebCenter Portal 11g

7.3.1 Running the Repository Creation Utility in Preparation for an Oracle WebCenter Upgrade

To start the Repository Creation Utility, you run the `rcu` command, which is in the root directory of the Repository Creation Utility CD-ROM or (if you have already installed the Oracle WebCenter Portal software on disk) in the bin directory of the Oracle WebCenter Portal Oracle home.

For more detailed instructions on how to obtain and run RCU, refer to *Oracle Fusion Middleware Repository Creation Utility User's Guide*.

For more information about which schemas to select when running the Repository Creation Utility, refer to Section 7.3.2, "Selecting the Schemas Required for Oracle WebCenter Portal 11g".

For more information about the Oracle WebCenter Portal schemas that are installed by the Repository Creation Utility, refer to "Metadata Repository Schemas" in *Oracle Fusion Middleware Administrator's Guide*.

7.3.2 Selecting the Schemas Required for Oracle WebCenter Portal 11*g*

You can use RCU to install the schemas required for all of the Oracle Fusion Middleware software components that require a schema. However, there is no need to install all the schemas unless you plan to install a complete Oracle Fusion Middleware environment and you plan to use the same database for all the Oracle Fusion Middleware component schemas.

For an Oracle WebCenter upgrade, you must select the following schemas when prompted by the Repository Creation Utility (as shown in Figure 7–1):

- Select the **Metadata Services** schema in the **AS Common Schemas** category.

  This schema supports Oracle Fusion Middleware Metadata Services (MDS), which is required by the Oracle WebCenter Spaces component.

- Select the entire **WebCenter Portal** category of schemas.

  These are the schemas supported by the various Oracle WebCenter Portal 11*g* products.

Figure 7–1   Selecting the Required Schemas for Oracle WebCenter Portal 11g
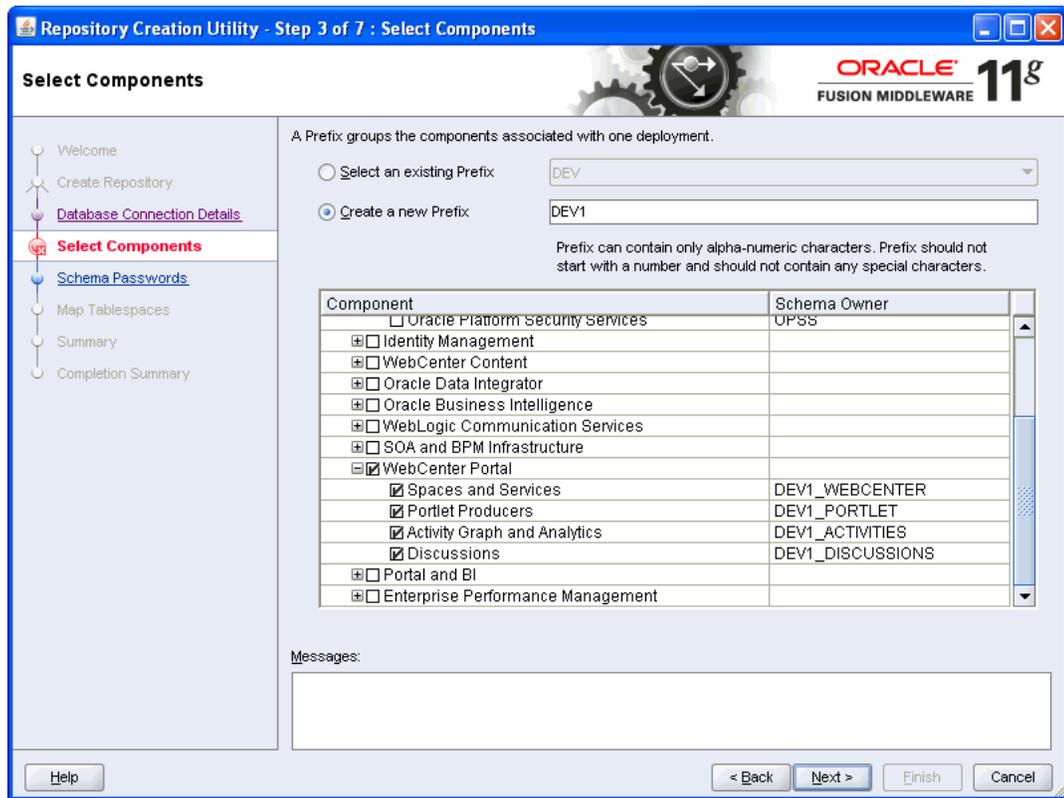
7.4 Task 4: Install and Configure the Oracle WebCenter Portal 11*g* Middle Tiers

Before you upgrade to Oracle Fusion Middleware 11*g*, you must install an Oracle Fusion Middleware environment that is similar to the topology you set up for Oracle Application Server 10*g*.

For complete instructions on installing Oracle Fusion Middleware, refer to the following documentation resources:

- *Oracle Fusion Middleware Installation Planning Guide*

- *Oracle Fusion Middleware Quick Installation Guide for Oracle WebCenter Portal*

- *Oracle Fusion Middleware Installation Guide for Oracle WebCenter Portal*

Table 7–1 provides a high-level summary of the main steps you must perform when installing and configuring the Oracle WebCenter Portal middle tiers, as well as specific references to the procedures you must follow for each step in *Oracle Fusion Middleware Installation Guide for Oracle WebCenter Portal*.

Note that the procedures described in this section assume you have downloaded the latest version of Oracle WebLogic Server and Oracle Fusion Middleware 11*g*. For more information, refer to "Obtaining the Latest Oracle WebLogic Server and Oracle Fusion Middleware 11*g* Software" in the *Oracle Fusion Middleware Upgrade Planning Guide*.

Table 7–1   Main Steps Required to Install and Configure Oracle WebCenter Portal Middle Tiers

Step #	Description	Section	Book
1	Install Oracle WebLogic Server and create the Middleware home	"Running the Installation Program in Graphical Mode"	*Oracle Fusion Middleware Installation Guide for Oracle WebLogic Server*
2	Install Oracle WebCenter Portal and create the WebCenter Portal Oracle home	"Installing Oracle WebCenter Portal"	*Oracle Fusion Middleware Installation Guide for Oracle WebCenter Portal*
3	Configure the Oracle WebCenter Portal domain	"Configuring Oracle WebCenter Portal"	*Oracle Fusion Middleware Installation Guide for Oracle WebCenter Portal*
4	Start the Oracle WebLogic Server domain	"Starting the Servers"	*Oracle Fusion Middleware Installation Guide for Oracle WebCenter Portal*
5	Verify the installation	"Verifying the Installation"	*Oracle Fusion Middleware Installation Guide for Oracle WebCenter Portal*

7.5 Task 5: Configure Any Required Java EE Features to Support Your WebCenter Applications

Besides the Oracle WebCenter Portal 11*g* system components, which are installed during the Oracle WebLogic Server installation, you might have to configure additional Oracle WebLogic Server features to support WebCenter applications you deploy.

For example, you may have to define application-required data sources, JMS resources, or other services. The resources and services provided by the Oracle WebLogic Server domain are common to all Java-based applications, including WebCenter applications.

For more information, see *Oracle Fusion Middleware Upgrade Guide for Java EE*.

7.6 Task 6: Upgrade Back-End Components for WebCenter Services

Oracle WebCenter provides a set of WebCenter services that expose social networking and personal productivity features for inclusion in WebCenter applications. Some of these services, such as Documents and Discussions, rely on back-end components. This section describes the back-end servers that you may need to prepare for working with Oracle WebCenter Portal 11*g*.

This section includes the following sections:

■   Upgrading Oracle WebCenter Discussions

■   Upgrading Oracle Content Server

■   Migrating Oracle WebCenter Wiki and Blog Server Data

7.6.1 Upgrading Oracle WebCenter Discussions

> **Note:** This section uses recently rebranded Oracle WebCenter Portal product names where appropriate. For more information, see Section 3.3.1, "New Product Names for Oracle WebCenter and Oracle Enterprise Content Management Suite."

If your WebCenter applications use the Discussions and Announcements services, you must install Oracle WebCenter Portal's Discussion Server 11*g*, and upgrade the database schema of your existing Oracle WebCenter Discussions 10.1.3.x server to Oracle WebCenter Portal's Discussion Server 11*g*.

Table 7–2 lists the tasks involved in upgrading Oracle WebCenter Discussions 10.1.3.x to Oracle WebCenter Portal's Discussion Server 11*g*.

Table 7–2   Tasks for Upgrading Oracle WebCenter Discussions

Step	Mandatory/Optional
Back up Oracle WebCenter Discussions database schema	Mandatory
Delete SSO-related system properties	Required only if SSO is configured
Install Oracle WebCenter Portal's Discussion Server 11*g*, if not already installed	Mandatory
Modify the database connection of Oracle WebCenter Portal's Discussion Server 11*g* to read data from the data store used by Oracle WebCenter Discussions 10.1.3.x	Mandatory
Use Jive Forums Upgrade Console to upgrade the discussions server	Mandatory
Configure WS-Security	Required only if SAML is enabled
Reconfigure SSO	Required only if SSO needs to be enabled

This section describes how to upgrade from Oracle WebCenter Discussions 10.1.3.x to Oracle WebCenter Portal's Discussion Server 11*g*.

To prepare your Oracle WebCenter Discussions 10.1.3.x server for upgrade:

1. Back up the database schema of your Oracle WebCenter Discussions 10*g*.

    > **Tip:** To find out the path to your database schema, log on to the Oracle WebCenter Discussions Admin Console as an administrator. From the **System** link at the top of the console, navigate to the **System Information** page. On this page, the **Datasource Info** section displays various details of your database schema, including the connection URL and the user name.

2. Remove the SSO-related system properties, if configured.

> **Note:** In Oracle WebCenter Portal's Discussion Server 11*g*, Single Sign-On (SSO) configuration has changed from Oracle WebCenter Discussions 10.1.3.x. If SSO is configured for Oracle WebCenter Discussions 10.1.3.x server and SSO-related system properties have been set to Oracle-specific SSO custom classes, then to avoid any SSO-related errors, you must disable the SSO configuration before you install Oracle WebCenter Portal's Discussion Server 11*g*. After upgrading your discussions server, you must reconfigure SSO if you require SSO authentication.

To remove the SSO-related system properties:

a. Log on to the Oracle WebCenter Discussions Admin Console as an administrator by using the following URL format:

```
http://host:port/owc_discussions/admin
```

Where *host* and *port* refer to the server and port number where your Oracle WebCenter Discussions 10*g* server is deployed.

b. On the **System** page of the Jive Forum Admin Console, in the navigation panel on the right, click **System Properties**.

c. On the **Jive Properties** page, under **All Properties**, delete the `AuthFactory.className` system property if it is set to the Oracle-specific custom class for SSO, `oracle.jive.sso.OracleSSOAuthFactory`.

d. Remove the `UserManager.className` system property if it is set to the Oracle-specific custom class for SSO, `oracle.jive.sso.OracleSSOUserManager`.

To upgrade Oracle WebCenter Discussions 10.1.3.x server to Oracle WebCenter Portal's Discussion Server 11*g*:

1. Ensure that Oracle WebCenter Portal's Discussion Server 11*g* is installed.

You can choose to install Oracle WebCenter Portal's Discussion Server 11*g* while installing WebCenter Portal 11*g* or later while extending your WebCenter Portal domain. For information, see *Oracle Fusion Middleware Installation Guide for Oracle WebCenter Portal*.

2. Modify the database connection of your Oracle WebCenter Portal's Discussion Server 11*g* to read data from the data source used by your Oracle WebCenter Discussions 10.1.3.x server. To edit the database connection, either use the Oracle WebLogic Server Administration Console or export/import your Oracle WebCenter Discussions 10.1.3.x server's database schema.

To use the database export and import commands to configure your database connection, export your Oracle WebCenter Discussions 10.1.3.x server's database schema to a dump file. Then, import the contents of the dump file into the database schema installed during the RCU installation for Oracle WebCenter Portal's Discussion Server 11*g*.

To edit the database connection by using the Oracle WebLogic Server Administration Console:

a. Log on to Oracle WebLogic Server Administration Console as an administrator. Use the following URL format:

```
http://host:port/console
```

Where *host* and *port* refer to the host name and port number where your Oracle WebLogic Server is installed. By default, the port number is 7001.

b. In Oracle WebLogic Server Administration Console, under **Domain Structure**, navigate to **Services** > **DataSources**.

c. On the **Summary of Data Sources** page, in the Data Sources table, click the **OWC_DiscussionsDS** connection link.

d. On the **Settings for DiscussionsDS** page, click the **Connection Pool** tab.

Modify the database connection of your Oracle WebCenter Portal's Discussion Server 11*g* to read data from your Oracle WebCenter Discussions 10.1.3.x server data store.

e. In the **URL** field on the **Connection Pool** tab, change the JDBC URL to point to the database used by Oracle WebCenter Discussions 10.1.3.x.

f. In the **Properties** field, change the user name to the one used for connecting to the database of Oracle WebCenter Discussions 10.1.3.x.

g. In the **Password** field, change the password to the one used for connecting to the database of Oracle WebCenter Discussions 10.1.3.x.

h. Click **Save**.

3. Restart the **WC_Collaboration** managed server where your Oracle WebCenter Portal's Discussion Server 11*g* is deployed.

4. Log on to Oracle WebCenter Portal's Discussion Server 11*g* as an administrator. Use the following URL format: http://*host*:*port*/owc_discussions.

5. In case the Jive Forums Upgrade Console displays, log on to the console as an administrator.

6. In Jive Forums Upgrade Console, select all the checkboxes to confirm that you have backed up your database and Jive Forum configuration files and to confirm the location of your jiveHome directory. (Figure 7–2)

Figure 7–2   Jive Forums Upgrade Console

7. Click **Start Upgrade** to upgrade the database schema for your discussions server.

   After all upgrade tasks have been completed, a confirmation message is displayed.

8. Restart the **WC_Collaboration** managed server where your Oracle WebCenter Portal's Discussion Server 11*g* is deployed.

9. Access your newly upgraded discussions server.

10. Configure Web Services Security (WS-Security) trust authentication on Oracle WebCenter Portal's Discussion Server and in your WebCenter applications if you plan to use this discussions server with WebCenter applications. WS-Security establishes a trust relationship between your WebCenter applications and the discussions server.

    For information about how to configure WS-Security, see the "Configuring WS-Security" chapter in *Oracle Fusion Middleware Administrator's Guide for Oracle WebCenter Portal*.

11. Reconfigure SSO for Oracle WebCenter Portal's Discussion Server, if required. For information, see the "Configuring Single Sign-On" chapter in *Oracle Fusion Middleware Administrator's Guide for Oracle WebCenter Portal*.

7.6.2 Upgrading Oracle Content Server

> **Note:**   This section uses recently rebranded Oracle WebCenter Portal product names where appropriate. For more information, see Section 3.3.1, "New Product Names for Oracle WebCenter and Oracle Enterprise Content Management Suite."

Oracle WebCenter Portal 11*g* supports Oracle Content Server 10.1.3.5.1 and Oracle WebCenter Content Server 11*g* as external content repositories. If you already have Oracle Content Server 10*g* installed, you must upgrade it to a supported version.

> **Note:** It is strongly recommended that you use Oracle Content Server 11*g* as the content repository for Oracle WebCenter Portal 11*g* to take advantage of the full set of features. If you use Oracle Content Server 10.1.3.5.1, you will have access only to a subset of features. For information, see section "Upgrading Oracle Universal Content Management 10*g*" in *Oracle Fusion Middleware Patching Guide*.

Refer to the following sections depending on the version of Oracle Content Server you wish to upgrade to:

- Upgrading to Oracle WebCenter Content Server 11g
- Upgrading to Oracle Content Server 10.1.3.5.1

7.6.2.1 Upgrading to Oracle WebCenter Content Server 11*g*

To use Oracle WebCenter Content Server 11*g*:

1. Upgrade your existing Oracle Content Server 10*g* installation to Content Server 11*g*. For information, see *Oracle Fusion Middleware Upgrade Guide for Oracle WebCenter Content*.

   If you do not already have Oracle Content Server installed, install Oracle WebCenter Content Server 11*g*. For information, refer to *Oracle WebCenter Content Installation Guide*.

2. Configure Content Server 11*g* for Oracle WebCenter Portal. For information, see the section "Prerequisites to Configuring Content Server" in *Oracle Fusion Middleware Administrator's Guide for Oracle WebCenter Portal*.

3. Disable `FrameworkFolders` and enable `folders_g`.

   When you upgrade Oracle Content Server 10*g* to Oracle WebCenter Content Server 11*g*, `FrameworkFolders` is enabled out-of-the-box. To configure Content Server 11*g* for WebCenter Portal 11*g*, you must disable `FrameworkFolders` and enable `folders_g`. For information, see the "Enabling Mandatory Components" section in *Oracle Fusion Middleware Administrator's Guide for Oracle WebCenter Portal*.

   > **Note:** For Content Server 11*g*, Oracle recommends Folders (the `FrameworkFolders` component) as the folder interface instead of Contribution Folders (the `Folders_g` component). However, WebCenter Portal still uses the `Folders_g` foldering capability. Therefore, to use Content Server 11*g* as a content repository for WebCenter Portal, you MUST have `Folders_g` enabled and `FrameworkFolders` disabled. You cannot have both enabled on a Content Server 11*g* instance.

7.6.2.2 Upgrading to Oracle Content Server 10.1.3.5.1

To patch your existing Oracle Content Server 10*g* installation to Oracle Content Server 10.1.3.5.1, you must install the latest `UpdateBundle` and the `Folders_g` component. For this, you must:

1. Download the latest `UpdateBundle`, `CS10gR35UpdateBundle` component from the following location:

   http://support.oracle.com

2. Install the latest `UpdateBundle`, and then `Folders_g`.

   The update bundle contains zip files for various other components, including `Folders_g`. When you install the update bundle, the zip files are extracted to the following directory:

   content_server_install_ home/custom/CS10gR35UpdateBundle/extras

   For installation information, see section "Uploading a Component" in the "Working with Content Server Components" guide available here:

   http://download.oracle.com/docs/cd/E10316_01/cs/cs_doc_ 10/documentation/developer/using_components_10en.pdf

3. Configure Oracle Content Server 10.1.3.5.1, if required. For information, see the section "Oracle Content Server 10.1.3.5.1 - Configuration" in *Oracle Fusion Middleware Administrator's Guide for Oracle WebCenter Portal*.

7.6.3 Migrating Oracle WebCenter Wiki and Blog Server Data

To support the wiki and blog functionality, Oracle WebCenter 10*g* relies on Oracle WebCenter Wiki and Blog Server, whereas WebCenter Portal 11*g* relies on Oracle WebCenter Content Server 11*g*. If your Oracle WebCenter 10*g* environment has Oracle WebCenter Wiki and Blog Server configured, you must migrate all your wiki and blog data to Content Server 11*g*. For information, see Section 15.5.1, "Migrating Data from Oracle WebCenter Wiki and Blog Server."

Part III

Upgrading Oracle SOA Suite, Oracle WebCenter, and ADF Applications

Part III contains the following chapters:

8

Overview of Upgrading Oracle SOA Suite, WebCenter, and ADF Applications

After you have installed and upgraded your Oracle Fusion Middleware environment, you can then upgrade and redeploy your Oracle Application Server 10*g* Release 2 (10.1.2) and 10*g* Release 3 (10.1.3) applications on your new Oracle Fusion Middleware topology.

This chapter describes the recommended process of migrating and redeploying your Oracle SOA Suite, WebCenter, and Java EE applications on Oracle Fusion Middleware 11*g*.

This chapter has the following sections:

- Task 1: Review Sample Application Upgrade Procedures
- Task 2: Verify that the Applications Are Up and Running Successfully on Oracle Application Server 10g
- Task 3: Review the Specific Considerations for the Type of Application You Are Upgrading
- Task 4: Verify That You Have Upgraded Your 10g Environment to 11g
- Task 5: Install and Start Oracle JDeveloper 11g
- Task 6: Open and Upgrade the Application in Oracle JDeveloper
- Task 7: Locate Specific Upgrade Information for the Components of Your Application
- Task 8: Review and Recompile the Application
- Task 9: Deploy the Application

8.1 Task 1: Review Sample Application Upgrade Procedures

Before you begin to upgrade your own custom Oracle SOA Suite, Oracle WebCenter, or Oracle ADF applications, consider reviewing the end-to-end procedures for upgrading some typical sample applications:

- If you want some experience upgrading a sample Oracle ADF application before tackling a larger scale upgrade, there is information available on Oracle Technology Network (OTN) that describes in detail how to migrate the SRDemo sample application, which is an ADF 10.1.3 JSF application using ADF Faces components, to Oracle JDeveloper 11g.

  For more information, see the article "Migrating the 10.1.3 SRDemo Sample to 11g", which is available on the Oracle Technology Network (OTN):

http://www.oracle.com/technology/products/jdev/tips/muench/1013srdemo/index.htm
l

- If you want to review the complete end-to-end steps for upgrading the an example Oracle SOA Suite 10*g* application, refer to Appendix A, "Example: Upgrading and Redeploying SOA Order Booking"

8.2 Task 2: Verify that the Applications Are Up and Running Successfully on Oracle Application Server 10*g*

Before you attempt to redeploy your Oracle Application Server 10*g* applications on Oracle Fusion Middleware 11*g*, you must be sure that the applications have been deployed successfully on Oracle Application Server 10*g* and that all components of the application are currently up and running.

In particular, be sure that any external references that your application relies upon are up and running and available. For example, if your application relies on external Web services, those Web services must be up and running when you start the 11*g* upgrade and redeployment process.

This step is necessary for two reasons:

- To verify that the application has no issues or problems in the 10*g* environment that might cause the upgrade and redeployment to fail on Oracle Fusion Middleware 11*g*.

- To reduce the amount of work required to redefine the external references in your application source code.

  For example, if your Oracle Enterprise Service Bus projects are calling external BPEL processes, it is important that the 10*g* Release 3 (10.1.3) BPEL processes are up and running and available when you upgrade the Oracle Enterprise Service Bus application.

  Otherwise, you must modify the Oracle Enterprise Service Bus project before the upgrade so the external references point to the new 11*g* BPEL processes. If the external references are not available, the upgrade to Oracle JDeveloper 11*g* will fail.

  Because of Oracle JDeveloper 11*g* development features, the process of redefining your external references is much easier and partially automated if you perform this task after the upgrade to Oracle JDeveloper 11*g*.

For an example of how to identify and verify the external partners for a specific, demonstration application (the SOA Order Booking demo application), see Section A.2.3, "Task 1c: Verify That All External Partners Are Available".

8.3 Task 3: Review the Specific Considerations for the Type of Application You Are Upgrading

Before you begin the upgrade process for each 10*g* application, make a list of the technologies and Oracle SOA Suite and WebCenter features you used when developing and deploying the application on Oracle Application Server 10*g*.

Then, review the information that pertains to your application in the remaining chapters of this guide. Those chapters contain information about upgrading specific Oracle SOA Suite and WebCenter features to Oracle Fusion Middleware 11*g*.

The chapters are organized by component and by the various techologies you might have used to develop and deploy your application. As a result, you can refer to the chapters and sections that apply to your specific types of applications.

8.4 Task 4: Verify That You Have Upgraded Your 10*g* Environment to 11*g*

Before you begin upgrading your applications to 11*g*, you must perform the upgrade tasks for your particular Oracle SOA Suite, WebCenter, and ADF or environment. For more information, refer to Part II, "Upgrading the Oracle SOA Suite, WebCenter, and ADF Environment".

In particular, as part of the environment upgrade, you should have:

- Verified that the new Oracle Fusion Middleware 11*g* environment is up and running.

- Applied any OC4J configuration settings to Oracle Fusion Middleware 11*g*; this includes any data sources, system properties, and other configuration settings that are required by the applications you are about to redeploy on Oracle Fusion Middleware 11*g*.

> **Note:** Before upgrading Oracle B2B-Related BPEL projects note, refer to Section 12.3.3, "Considerations When Upgrading B2B-Related Oracle BPEL Processes".

8.5 Task 5: Install and Start Oracle JDeveloper 11*g*

For complete instructions about downloading, installing, and starting Oracle JDeveloper 11*g*, refer to the *Oracle Fusion Middleware Installation Guide for Oracle JDeveloper*.

After you start Oracle JDeveloper 11*g*, you should verify that you have the proper Oracle JDeveloper extensions for the applications you will be developing. For more information, see:

- Section 9.2, "About Using the Oracle JDeveloper 11g Migration Wizard for Oracle SOA Suite Applications"

- Section 15.2, "Prerequisites for Upgrading WebCenter Applications"

> **Note:** If you intend to migrate application settings from JDeveloper 10.1.3.x, then close and remove your WebCenter applications from Oracle JDeveloper 10.1.3.x before you start Oracle JDeveloper 11*g* for the first time and before you install the WebCenter Oracle JDeveloper 11*g* extension.
>
> You can remove applications from Oracle JDeveloper by right-clicking an application name in the Application Navigator and selecting **Delete** from the shortcut menu. Removing an application removes it only from Oracle JDeveloper and not from the disk.

8.6 Task 6: Open and Upgrade the Application in Oracle JDeveloper

For each Oracle Application Server 10*g* Oracle SOA Suite, WebCenter, and ADF application, you should have an Oracle JDeveloper 10*g* workspace (.jws) file. When

you open a workspace from a previous version of Oracle JDeveloper, the Migration Wizard automatically upgrades the application to Oracle Fusion Middleware 11*g*.

This important step saves you time and effort in migrating to the newer versions of Java technologies supported by Oracle Fusion Middleware 11*g*.

Use the following steps to open an Oracle JDeveloper 10*g* workspace in Oracle JDeveloper 11*g*:

1. Save a copy of the project that you can manage using Oracle JDeveloper 10g Release 3 (10.1.3.1.0) in a separate folder or directory on disk.

   Oracle JDeveloper backs up all your project directories as they are imported and upgraded to Oracle JDeveloper 11*g*. However, it is still good development practice to back up your work before starting the upgrade process.

   Note that Oracle JDeveloper does not back up the workspace (.jws) file, only the project directories. Also, after you open and upgrade the workspace in Oracle JDeveloper 11*g*, you cannot open the upgraded copy of the workspace in a previous version of Oracle JDeveloper.

   As a result, if you need to restore your original application or projects, Oracle recommends that you restore your application from your own pre-upgrade backup of the entire application directory structure.

2. Select **Open** from the **File** menu and select the Oracle JDeveloper 10g `.jws` file.

   When selecting the workspace file, be sure the application source code is available in a directory location that will always be available. For example, do not attempt to upgrade a project that is stored on a network drive path (such as `\\110.34.40.43\share\Samples\mysample.jws`).

   Oracle JDeveloper displays Step 1, the Welcome page of the Migration Wizard.

3. Follow the instructions on the screen to migrate the project files and selected Java technologies to Oracle JDeveloper Release 11*g* and Oracle Fusion Middleware 11*g*.

4. Review the application migration log files, which are located in the following directory inside the Oracle JDeveloper home directory:

   On UNIX systems:

   `JDEV_HOME/JDeveloper/upgrade/logs/`

   On Windows systems:

   `JDEV_HOME\JDeveloper\upgrade\logs\`

   > **Note:** You can also upgrade individual Oracle JDeveloper 10*g* project (`.jpr`) files. However, Oracle recommends you take the following steps when upgrading `.jpr` files:
   >
   > 1. Start Oracle JDeveloper 11*g*.
   > 2. Create a new application in Oracle JDeveloper 11*g*.
   > 3. Open the .jpr file from within the application so it becomes a project within the newly created application.

8.7 Task 7: Locate Specific Upgrade Information for the Components of Your Application

This chapter provides an overview of the steps required to upgrade and redeploy your Oracle SOA Suite, WebCenter, and ADF applications on Oracle Fusion Middleware 11*g*.

However, there are important considerations and upgrade tasks you might need to perform in addition to the general tasks described here.

For more information, review the remaining chapters in this guide for any special considerations that affect your Oracle SOA Suite, WebCenter, and ADF applications.

The chapters are organized by component and by the various techologies you might have used to develop and deploy your application. As a result, you can refer to the chapters and sections that apply to your specific types of applications.

8.8 Task 8: Review and Recompile the Application

After you have opened the application in Oracle JDeveloper 11*g*, you should then perform the following tasks:

1. From the Oracle JDeveloper 11*g* **File** menu, select **Save All** to save all the projects in the selected workspace.

2. From the **Build** menu, select **Make All** to recompile all the projects in the workspace, or right-click one project at a time and select **Make *project_name***.

3. Review the output in the Compiler message panel at the bottom of the Oracle JDeveloper window.

4. For each compilation error, double-click the error icon and review the problem code.

   For more information, refer to the appropriate developer guide in the Oracle Fusion Middleware 11*g* documentation library. The Oracle Fusion Middleware 11*g* documentation library is available at the following location on the Oracle Technology Network (OTN):

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

8.9 Task 9: Deploy the Application

After you have reviewed the available resources and successfully recompiled the projects in the application, you can then deploy the application on Oracle Fusion Middleware 11*g*.

For more information about deploying applications, refer to the following resources:

- *Oracle Fusion Middleware Deploying Applications to Oracle WebLogic Server*, for information about deploying Java EE applications to Oracle WebLogic Server

- "Deploying Fusion Web Applications" in the *Oracle Fusion Middleware Fusion Developer's Guide for Oracle Application Development Framework*, for information about deploying Oracle ADF applications

- "Understanding the Deployment Process" in the *Oracle Fusion Middleware Administrator's Guide*, for general information about deploying applications in an Oracle Fusion Middleware environment.

9

Considerations When Upgrading All Oracle SOA Applications

This chapter provides important supplementary information upgrading Oracle SOA applications to Oracle Fusion Middleware 11*g*.

Use Chapter 8, "Overview of Upgrading Oracle SOA Suite, WebCenter, and ADF Applications" for the tasks required to upgrade any Oracle SOA Suite, WebCenter, and ADF application.

Use the following sections to understand tasks specific to upgrading Oracle SOA applications:

- Verify That All External Partners Are Available
- Section 9.2, "About Using the Oracle JDeveloper 11g Migration Wizard for Oracle SOA Suite Applications"
- Section 9.3, "Upgrade Tasks Associated with All Java Applications"
- Section 9.4, "Upgrade Tasks Associated with All Oracle SOA Suite Applications"

9.1 Verify That All External Partners Are Available

Before you upgrade the Oracle SOA Suite application to 11*g*, you must ensure that the external partners (in the form remote Web services) are up and running and available.

Oracle JDeveloper 11*g* must be able to access these external partners, as well as the Oracle BPEL Process Manager and Oracle Enterprise Service Bus services deployed on 10g; otherwise, it can not properly migrate the projects in your application to 11*g*.

To verify that the required Web services are available, you should review each of the projects in your Oracle SOA Suite applications before you migrate them to 11*g*. For example, you can review the Oracle BPEL Process Manager configuration files for each project and verify that the Web services URIs referenced in those configuration files are valid and working.

9.2 About Using the Oracle JDeveloper 11*g* Migration Wizard for Oracle SOA Suite Applications

When you open an Oracle Application Server 10*g* Oracle SOA Suite application in Oracle JDeveloper 11*g*, the Oracle JDeveloper Migration Wizard attempts to upgrade your application automatically to Oracle Fusion Middleware 11*g*.

However, there are some limitations to what the Oracle JDeveloper Migration Wizard can perform automatically. Refer to specific sections of this chapter for information

about the types of manual tasks you might have to perform on your Oracle SOA Suite applications before or after using the Migration Wizard.

Applying the Latest Patch Sets

For best results, Oracle recommends that you apply the most recent patch sets to your Oracle SOA Suite environment and that you use the latest 10*g* Release 3 (10.1.3) Oracle JDeveloper before upgrading to 11*g*.

Keeping Oracle JDeveloper and Oracle SOA Suite at the Same Version Level

As a general rule, you should always update your Oracle SOA Suite and Oracle JDeveloper installations at the same time and run the same version of both these Oracle products.

Verifying That You Have the Required SOA Composite Editor Oracle JDeveloper Extension

To upgrade your Oracle SOA Suite applications to 11*g*, you must have the Oracle SOA Composite Editor extension for Oracle JDeveloper 11*g*.

To verify that the extension is installed, select **About** from the Oracle JDeveloper **Help** menu, and click the **Version** tab. You should see an entry in the list of components called `SOA Composite Editor`.

If this component does not appear on the Version tab of the About dialog box, close the About dialog and select **Check for Updates** from the **Help** menu. Use the Check for Updates wizard to locate and install the latest version of the SOA Composite Editor extension.

9.3 Upgrade Tasks Associated with All Java Applications

Before you begin upgrading your Oracle SOA Suite applications, be sure to review the *Oracle Fusion Middleware Upgrade Guide for Java EE*, which contains information about upgrading standard Java EE applications to Oracle WebLogic Server.

If your applications contain any custom Java code, you should review the Java code against the procedures and recommendations available in the *Oracle Fusion Middleware Upgrade Guide for Java EE*.

9.4 Upgrade Tasks Associated with All Oracle SOA Suite Applications

The following information should reviewed when you are upgrading any Oracle SOA Suite application to Oracle Fusion Middleware 11*g*:

- Understanding Oracle SOA Suite API Changes for Oracle Fusion Middleware 11g
- Reviewing Your Projects for Dependent JAR Files
- Upgrading Applications That Require Proxy Settings for Web Services
- Recreating build.xml and build.properties Files Not Upgraded by the Migration Wizard
- Upgrading Projects That Use UDDI-Registered Resources
- Upgrading User-Defined (Custom) XPath Functions in Oracle SOA Suite
- Understanding File Naming Conventions in Oracle SOA Suite 11g
- Using the Oracle SOA Suite Command-Line Upgrade Tool
- Upgrading Oracle SOA Suite 10g Cross-References (XREF) Runtime Data to 11g

9.4.1 Understanding Oracle SOA Suite API Changes for Oracle Fusion Middleware 11*g*

Table 9–1 describes the APIs you can use in an Oracle SOA Suite application. For each Oracle Application Server 10*g* API, it provides a summary of the changes for Oracle Fusion Middleware 11*g* and where you can get more information about upgrading your applications that use the API.

Table 9–1   Summary of Oracle SOA Suite API Changes for Oracle Fusion Middleware 11g

10g API	Action for 11g	11g JavaDoc
Oracle Application Server Integration B2B Callouts Java API	Updated for Oracle Fusion Middleware 11*g*, but the new 11*g* version is backward compatible with 10*g*.	*Oracle Fusion Middleware B2B Callout Java API Reference*
Oracle BPEL Process Manager Workflow Services API	Updated for Oracle Fusion Middleware 11*g*, but the new 11*g* version is backward compatible with 10*g*.	*Oracle Fusion Middleware Workflow Services Java API Reference for Oracle SOA Suite*
Oracle Business Rules Java API	See Section 9.4.1.1, "Upgrading to the Oracle Fusion Middleware Java API for Oracle Business Rules"	*Oracle Fusion Middleware Java API Reference for Oracle Business Rules*
Oracle BPEL Process Manager Client Java API	See Section 9.4.1.2, "Upgrading to the Oracle Fusion Middleware Infrastructure Management Java API for Oracle SOA Suite"	See Section 9.4.1.2, "Upgrading to the Oracle Fusion Middleware Infrastructure Management Java API for Oracle SOA Suite"
Oracle BPEL Process Manager Sensor API	Updated for Oracle Fusion Middleware 11*g*, but the new 11*g* version is backward compatible with 10*g* and part of the new Infrastructure Management Java API.	"Using Oracle BPEL Process Manager Sensors" in the *Oracle Fusion Middleware Developer's Guide for Oracle SOA Suite*

9.4.1.1 Upgrading to the Oracle Fusion Middleware Java API for Oracle Business Rules

The following sections introduce the Oracle Business Rules 11*g* SDK and API and provide instructions for upgrading to the Oracle Business Rules API:

- Overview of the Oracle Business Rules SDK and API Changes for 11g

- Accessing a Dictionary in the Development Environment

- Accessing a Repository in a Production Environment

- Generating RL Code

- Upgrading WebDAV Oracle Business Rules Project

9.4.1.1.1   Overview of the Oracle Business Rules SDK and API Changes for 11*g* In Oracle Fusion Middleware11g, the Oracle Business Rules SDK and API has been significantly improved.

In Oracle Application Server 10*g*, developers were required to manually manage accessing the repository and creating and using `RuleSession` instances. In Oracle Fusion Middleware 11*g*, the Decision Point API and decision function features provide an interface and implementation that simplifies the definition and execution of rules.

When upgrading to Oracle Business Rules 11*g*, look first at how you can use these new features, as documented in the *Oracle Fusion Middleware User's Guide for Oracle Business Rules*.

However, if you want to continue to use the Oracle Business Rules SDK in the same way as you did for 11*g*, then the following sections describe how to directly translate the Oracle Business Rules 10*g* SDK to the 11*g* SDK.

All of the classes discussed in this section are in the `rulesdk2.jar` file and under the `oracle.rules.sdk2` package.

9.4.1.1.2 Accessing a Dictionary in the Development Environment In Oracle Business Rules 10*g*, it was common for an application developer to use a file-based repository while developing the application, and then switch to using a WebDAV repository for production.

In Oracle Business Rules 11*g*, you can instead access the rule dictionary file directly before it is packaged into a format which can be deployed to MDS.

In the following examples, compare the code required to access a dictionary in development mode in 10*g* (Example 9–1) with the code required in 11*g* (Example 9–2).

Note that in general, you should use the Decision Point API rather than continuing to access the RuleRepository directly.

Example 9–1 Accessing a Dictionary with Oracle Business Rules 10g in a Development Environment

```
String path; // the path to the file repository
Locale locale; // the desired Locale
// The following code assumes that the path and locale have been set appropriately
RepositoryType rt =
RepositoryManager.getRegisteredRepositoryType("oracle.rules.sdk.store.jar");
RuleRepository repos = RepositoryManager.createRuleRepositoryInstance(rt);
RepositoryContext rc = new RepositoryContext();
rc.setLocale(locale);
rc.setProperty("oracle.rules.sdk.store.jar.path", path);
repos.init(rc);
```

Example 9–2 Accessing a Dictionary with Oracle Business Rules 11g in a Development Environment

```
protected static final String DICT_LOCATION ="C:\\scratch\\CarRental.rules";
...
RuleDictionary dict = null;
Reader reader = null;
try {
  reader = new FileReader(new File(DICT_LOCATION));
  dict = RuleDictionary.readDictionary(reader, new
                 DecisionPointDictionaryFinder(null));
  List<SDKWarning> warnings = new ArrayList<SDKWarning>();
  dict.update(warnings);
  if (warnings.size() > 0 ) {
     System.err.println("Validation warnings: " + warnings);
     }
     } catch (SDKException e){
         System.err.println(e);
     } catch (FileNotFoundException e){
         System.err.println(e);
     } catch (IOException e){
         System.err.println(e);
     } finally {
         if (reader != null) {
             try { reader.close(); } catch (IOException ioe)
                 {ioe.printStackTrace();}
```

```
    }
}
```

9.4.1.1.3 Accessing a Repository in a Production Environment In Oracle Business Rules 10*g*, WebDAV was the recommended production repository.

In Oracle Business Rules 11*g*, WebDAV is no longer supported and Metadata Services (MDS) is the recommended repository. Also, the dictionary "name" and "version" have been replaced with a "package" and "name" (similarly to the Java class naming scheme).

In Oracle Business Rules 10*g*, the version did not provide true versioning. In Oracle Business Rules 11*g*, the equivalent to specifying a version is to simply change the name. For example, a 10*g* dictionary with the name `foo.bar.MyDict` and version 2 would in 11*g* be packaged as `foo.bar` and name `MyDict2`.

In the following examples, compare the code required to access a dictionary in production mode in 10*g* (Example 9–3) with the code required in 11*g* (Example 9–4).

Example 9–3   *Accessing a Dictionary with Oracle Business Rules 10g in a Production Environment*

```
String url; // the URL for the WebDAV repository
Locale locale; // the desired Locale
// The following code assumes that the url and locale have been set appropriately
RepositoryType rt =
    RepositoryManager.getRegisteredRepositoryType("oracle.rules.sdk.store.webdav");
RuleRepository repos = RepositoryManager.createRuleRepositoryInstance(rt);
RepositoryContext rc = new RepositoryContext();
rc.setLocale(locale);
rc.setProperty("oracle.rules.sdk.store.webdav.url", url);
repos.init(rc);
RuleDictionary dictionaryWithInitialVersion =
                repos.loadDictionary(dictionaryName);
RuleDictionary dictionarySpecificVersion = repos.loadDictionary(dictionaryName,
                dictionaryVersion);
```

Example 9–4   *Accessing a Dictionary with Oracle Business Rules 11g in a Production Environment*

```
import static
    oracle.rules.sdk2.repository.RepositoryManager.createRuleRepositoryInstance;
import static
    oracle.rules.sdk2.repository.RepositoryManager.getRegisteredRepositoryType;
import static oracle.rules.sdk2.store.mds.Keys.CONNECTION;
...
private static final String DICT_PKG = "oracle.middleware.rules.demo";
private static final String DICT_NAME = "CarRental";
private static final DictionaryFQN DICT_FQN =
new DictionaryFQN(DICT_PKG, DICT_NAME);
...
RuleRepository repo =
    createRuleRepositoryInstance(getRegisteredRepositoryType(CONNECTION));
repo.init(new RepositoryContext() {{
    setDictionaryFinder(new DecisionPointDictionaryFinder(null));
}});
RuleDictionary dict = repo.load(DICT_FQN);
```

9.4.1.1.4 Generating RL Code The following example shows the code required generate RL code in Oracle Business Rules 10*g*. In 11*g*, this code remains the same.

```
//init a rule session
String rsname = "vehicleRent";
String dmrl = dict.dataModelRL();
String rsrl = dict.ruleSetRL( rsname );
```

9.4.1.1.5 Upgrading WebDAV Oracle Business Rules Project If you are upgrading Oracle Business Rules 10*g* project that includes WebDAV, then Oracle recommends that you use Java JDK for upgrading to Oracle Business Rules 11*g*. If you are using Oracle JRockit as the JVM then the upgrade process will get stuck.

9.4.1.2 Upgrading to the Oracle Fusion Middleware Infrastructure Management Java API for Oracle SOA Suite

Oracle SOA Suite 11*g* introduces a new infrastructure management API that replaces several other APIs previously available in Oracle SOA Suite 10*g*.

For more information about the new API, refer to the *Oracle Fusion Middleware Release Notes*, which are available as part of the Oracle Fusion Middleware 11*g* documentation library.

9.4.2 Reviewing Your Projects for Dependent JAR Files

If you are upgrading an Oracle SOA Suite application that depends upon references to custom JAR file libraries. note that these references may not get upgraded automatically when you open and upgrade your application in Oracle JDeveloper 11*g*.

As a result, you should review your projects for these types of dependencies and, after the upgrade, make sure that you add any missing references, by selecting the **Libraries and Classpath** link in the Oracle JDeveloper 11*g* Project Properties dialog.

9.4.3 Upgrading Applications That Require Proxy Settings for Web Services

If you are upgrading an application that uses Web services resources outside your company firewall, you must modify a configuration file that will enable the upgrade to accesses proxy settings and adjust them accordingly during the upgrade of the application.

To configure Oracle JDeveloper 11*g* to use the proxies during the upgrade process:

1. Locate the following file in the *JDEV_HOME*/bin directory:

   ```
   ant-sca-upgrade.xml
   ```

2. Edit the file and modify the following settings to identify the proxy server and port required to resolve the Web services addresses for the applications you are upgrading.

   ```
   <jvmarg value="-Dhttp.proxySet=true"/>
   <jvmarg value="-Dhttp.proxyHost=PROXY_SERVER"/>
   <jvmarg value="-Dhttp.proxyPort=PROXY_PORT"/>
   ```

   Note that there are two locations in the file to modify these settings. One location is used to set the proxy when you are upgrading an ESB project to Oracle Mediator 11*g*; the other is used when you are upgrading Oracle BPEL Process Manager projects.

3. Stop and start Oracle JDeveloper 11*g* so your changes can take effect and then open and upgrade your application in Oracle JDeveloper 11*g*.

9.4.4 Recreating build.xml and build.properties Files Not Upgraded by the Migration Wizard

When you open and upgrade an application in Oracle JDeveloper 11*g*, the `build.xml` and `build.properties` files associated with your application projects are not upgraded.

Instead, you must recreate these files in Oracle JDeveloper 11*g* after the upgrade.

9.4.5 Upgrading Projects That Use UDDI-Registered Resources

The following information is important if any of your Oracle BPEL Process Manager or Oracle Enterprise Service Bus 10*g* projects use remote resources that are registered in a UDDI (Universal Description, Discover and Integration) registry, such as Oracle Service Registry (OSR).

Refer to the following for more information:

- Verifying that serviceKey Endpoints Are Available Before Upgrade
- Changing to the orauddi Protocol

9.4.5.1 Verifying that serviceKey Endpoints Are Available Before Upgrade

If you have a 10*g* Release 3 (10.1.3) project that references an endpoint URL that uses a serviceKey from OSR, then you must be sure that the endpoint URL is up and available when you upgrade the application. Otherwise, the upgrade of the application will fail. To prevent such a failure, verify that the endpoint URLs are available, and if necessary, modify the endpoint URLs in `bpel.xml` or the routing service to point to new URL which is accessible.

9.4.5.2 Changing to the orauddi Protocol

In Oracle Application Server 10*g* Release 3 (10.1.3), Oracle BPEL Process Manager and Oracle Enterprise Service Bus projects used the `uddi` protocol to obtain resource references from OSR. In Oracle Fusion Middleware 11*g*, Oracle BPEL Process Manager and Oracle Mediator projects use the `orauddi` protocol.

As a result, prior to upgrading your Oracle BPEL Process Manager or Oracle Enterprise Service Bus projects, you must do the following:

1. Use the service registry to modify the registered service so it uses the new bindings supported by Oracle Fusion Middleware 11*g*.

    For example, in OSR, do the following:

    a. Log in to your Oracle Service Registry.

    b. On the Search tab, click **Businesses**.

    c. Click **Add Name** to search for a business by name.

    d. From the search results, click the name of the business you want to modify.

    e. In the left pane of the View Business page, right-click the service you want to modify and select **Add Binding** from the context menu.

    f. From the **Type** drop-down menu on the Add Binding page, select **wsdlDeployment**.

    g. In the **Access Point** field, enter the URL.

    For example:

```
http://machine:port/Proj_ep?WSDL/
```

     h. Click **Add Binding**.

     i. Click **Save Changes**.

2. Open and upgrade the application in Oracle JDeveloper 11*g*.

3. In Oracle JDeveloper, edit the `composite.xml` file for the upgraded project and configure the endpoint URL using the **UDDI Registry** option on the Resource Palette.

For more information about the UDDI registry, visit the following URL:

```
http://www.oracle.com/technology/tech/soa/uddi/index.html
```

9.4.6 Upgrading User-Defined (Custom) XPath Functions in Oracle SOA Suite

When you open and upgrade your applications in Oracle JDeveloper 11*g*, any custom XPath functions in the application projects are not upgraded automatically.

As a result, after you upgrade your application, you must copy any XPath function classes into the server classpath and register the function in the server configuration file. This is a manual step because the Oracle JDeveloper Migration Wizard cannot assume the server information.

For more information, refer to "Creating User-Defined XPath Extension Functions" in the *Oracle Fusion Middleware Developer's Guide for Oracle SOA Suite*.

9.4.7 Understanding File Naming Conventions in Oracle SOA Suite 11*g*

Note that before you upgrade a SOA composite application, verify that the name of the application or project does not start with a number.

For example, if you attempt to upgrade a 10*g* sample application called `110.JMStoJMSHeaders`, Oracle JDeveloper displays a "grammar validation error" when you attempt to open the `composite.xml`.

To work around this problem, simply rename the application and remove the number before you upgrade the application.

9.4.8 Using the Oracle SOA Suite Command-Line Upgrade Tool

The following sections describe how to use the Oracle SOA Suite command-line upgrade tool:

- Benefits of Using the Oracle SOA Suite Command-Line Upgrade Tool
- Using the Oracle SOA Suite Command-Line Upgrade Tool with Oracle JDeveloper 11g
- Limitations When Upgrading Human Workflow Applications with the Oracle SOA Suite Command-Line Upgrade Tool
- Upgrading BPEL or ESB Projects with the Oracle SOA Suite Command-Line Upgrade Tool
- Combining Multiple BPEL Projects Into a Single Composite with the Oracle SOA Suite Command-Line Upgrade Tool
- Upgrading Oracle Enterprise Service Bus (ESB) Projects with the Oracle SOA Suite Command-Line Upgrade Tool

- Upgrading Domain Value Maps (DVMs) and Cross References (XREFs) in the ESB Repository

9.4.8.1 Benefits of Using the Oracle SOA Suite Command-Line Upgrade Tool

The Oracle SOA Suite Command-Line Upgrade Tool has the following benefits:

- You can use the command-line tool to automate your application upgrade work, using scripts or other command-line automation tools.

- The command-line tool upgrades both Oracle BPEL Process Manager projects, and upgrades Oracle Enterprise Service Bus 10*g* projects to Oracle Mediator 11*g*.

- With the command-line tool, you can merge BPEL projects together and create a single composite out of multiple BPEL projects. This is not possible when you use the Oracle JDeveloper Migration Wizard.

- The functionality is exactly the same as the JDeveloper mode when it comes to dealing with SOA project contents because the same codebase is used.

9.4.8.2 Using the Oracle SOA Suite Command-Line Upgrade Tool with Oracle JDeveloper 11*g*

The Oracle SOA Suite Command-Line Upgrade Tool is compatible (with restrictions) with the Oracle JDeveloper Migration Wizard. In other words, you can choose to remain in command-line mode all the way through the upgrade process (upgrade, compile, package, and deploy), or you can choose to move to Oracle JDeveloper, or you use both tools, with no functionality loss.

However, it is important to note that the command-line tool upgrades SOA project artifacts only. Other Oracle JDeveloper artifacts (for example, the `.jpr` and `.jws` files) are ignored.

To work around this restriction, note the following:

- The Oracle SOA Suite Command-Line Upgrade Tool copies files from the BPEL suitcase directory (the BPEL subdirectory or the directory hosting the BPEL files) to the specified target directory, specified on the command line.

- The above copying action does not copy the `.jpr` or `.jws` files. After the upgrade, the target directory contains only the upgraded SOA project contents.

- To remedy this problem in Oracle JDeveloper, you can create a new application or new project, and then define the project directory to be the newly upgraded composite directory.

9.4.8.3 Limitations When Upgrading Human Workflow Applications with the Oracle SOA Suite Command-Line Upgrade Tool

If you attempt to use the Oracle SOA Suite Command-Line Upgrade Tool to upgrade an application that contains Oracle Human Workflow projects, note that the tool will create a separate project for each upgraded task form.

This resulting project is an Oracle ADF project and Oracle ADF does not support command line deployment and Oracle JDeveloper deployment. As a result, after using the Oracle SOA Suite Command-Line Upgrade Tool, you must open the upgraded projects in Oracle JDeveloper 11*g* deploy it from JDeveloper.

For information about how to open the upgraded project in Oracle JDeveloper, see Chapter 9.4.8.2, "Using the Oracle SOA Suite Command-Line Upgrade Tool with Oracle JDeveloper 11g".

9.4.8.4 Upgrading BPEL or ESB Projects with the Oracle SOA Suite Command-Line Upgrade Tool

The files required to run the Oracle SOA Suite Command-Line Upgrade Tool are installed automatically when you install Oracle JDeveloper 11*g* and when you install Oracle SOA Suite 11*g*:

- When you install the Oracle SOA Suite 11*g*, the required XML files are installed in the bin directory of the Oracle SOA Suite Oracle home.

- When you install Oracle JDeveloper 11*g*, the required XML files are installed in the bin directory of the Oracle JDeveloper home.

You can use this files in this directory, together with Apache Ant, to migrate your 10*g* Release 3 (10.1.3) SOA projects to 11*g*.

> **Note:** For the purposes of this procedure, it is assumed you are running the Oracle SOA Suite Command-Line Upgrade Tool from the Oracle home of an Oracle SOA Suite installation.
>
> If you are running it from a Oracle JDeveloper home, replace *ORACLE_HOME* with *JDEV_HOME* in the following procedures.

To use the ANT project:

1. Set the ORACLE_HOME environment variable so it is defined as the path to the Oracle SOA Suite Oracle home.

2. Set the other environment variables that are required by the command-line tool, by running the following script:

   On UNIX systems:

   ```
   . ORACLE_HOME/bin/soaversion.sh
   ```

   On Windows systems:

   ```
   ORACLE_HOME\bin\soaversion.cmd
   ```

3. Run the Ant project, as follows:

   ```
   ant -f ORACLE_HOME/bin/ant-sca-upgrade.xml
           -Dsource sourceDir
           -Dtarget targetDir
           -DappName app_name
           -Drevision 1.0
           bpel_or_mediator_identifier
   ```

   For examples of using the Oracle SOA Suite Command-Line Upgrade Tool, see Example 9–5 and Example 9–6.

   For a description of each command-line argument, see Table 9–2.

4. After the upgrade is complete, run the SCA-Compiler (`ant-sca-compile.xml`) which will verify the migrated sources.

   Because the SCA-Upgrade will not generate complete artifacts in all cases, you will see some errors or warnings from SCA-Compiler with information on how to fix them. Check the SCA Compiler for reference.

5. After you get a clean pass from the SCA-Compiler, use the `ant-sca-package.xml` tool to package your application.

You can then deploy the application using `ant-sca-deploy.xml`. After deployment, your project will be available to the server for testing.

Note, however, that in most cases, you will likely want to open the upgraded project in Oracle JDeveloper 11*g*. From there, you can easily review, verify, and make any necessary updates to the application projects.

For more information, see Section 9.4.8.2, "Using the Oracle SOA Suite Command-Line Upgrade Tool with Oracle JDeveloper 11g".

Example 9–5   Command-Line Example on the UNIX Operating System

```
ant -f /Disk03/Oracle/Middleware/SOA_home/bin/ant-sca-upgrade.xml
    -Dsource /disk03/myProjects/my_bpel_app/
    -Dtarget /disk03/my11gProjects/my_bpel_app/
    -Drevision 1.0
    -DappName my_bpel_app
    bpel
```

Example 9–6   Command-Line Example on the Windows Operating System

```
ant -f C:\Oracle\Middleware\SOA_home\bin\ant-sca-upgrade.xml
    -Dsource C:\myProjects\my_bpel_app/
    -Dtarget C:\my11gProjects\my_bpel_app/
    -Drevision 1.0
    -DappName my_bpel_app
    bpel
```

Table 9–2   Summary of the Oracle SOA Suite Command-Line Upgrade Tool Arguments

Command-Line Argument	Description
-Dsource	The fully-qualified name and path for the directory where the Oracle SOA Suite 10*g* project files reside.
	For Oracle BPEL Process Manager projects, this can be the `bpel` folder, the parent directory that hosts the `bpel` folder.
	For Oracle Enterprise Service Bus projects, this can be the Oracle Enterprise Service Bus project folder.
-Dtarget	The fully-qualified name and path for the directory that will contain the upgraded project files.
	The Oracle SOA Suite Command-Line Upgrade Tool upgrades your Oracle BPEL Process Manager project files, and generates new, upgraded files in a subdirectory of the target directory specified in this argument.
-DappName	The Oracle SOA Suite Command-Line Upgrade Tool uses this value to create an application subdirectory inside the target directory. The tool then copies the upgraded project files to application subdirectory.
-Drevision	Need Info ------------
bpel_or_mediator_identifier	Can be one of two values:
	■ `bpel`, to identify a Oracle BPEL Process Manager project that you want to upgrade to Oracle BPEL Process Manager 11*g*
	■ `mediator`, to identify an Oracle Enterprise Service Bus project that you want to upgrade to Oracle Mediator 11*g*.
	If you do not specify a value, then the command-line tool assumes you are upgrading an Oracle BPEL Process Manager project.

When you run the Oracle SOA Suite Command-Line Upgrade Tool, note the following:

- If the *sourceDir* and the *targetDir* directories are the same directory, then the command-line upgrade tool will automatically create a backup of the directory before migration; if you must re-run the upgrade, you can restore the source files using the backup directory. You can identify the backup directory by the ".backup" suffix in its name.

  If the *sourceDir* and *targetDir* directories are different, a backup directory is not created and is not necessary, because the files in the *sourceDir* are not modified during the migration.

- After upgrade, the logs can be found in the following output directory:

  ```
  ORACLE_HOME/upgrade/logs
  ```

9.4.8.5 Combining Multiple BPEL Projects Into a Single Composite with the Oracle SOA Suite Command-Line Upgrade Tool

Using the Oracle SOA Suite Command-Line Upgrade Tool, you can merge BPEL projects together and create a single composite. This is not possible when you use the Migration Wizard in Oracle JDeveloper.

To combine multiple BPEL projects into a single composite, provide multiple source directories as part of the -Dsource property on the command line.

Path separators can be a colon (:) or a semicolon (;). Ant will convert the separator to platform's local conventions. As a guideline, also use double quotes to identity the multiple source directories to prevent Ant from parsing the input in an unexpected manner.

The first source directory specified will be considered as the root of the 11g project and will determine the composite name.

For example:

```
ant -f ORACLE_HOME/bin/ant-sca-upgrade.xml
    -Dsource "sourceDir1:sourceDir2"
    -Dtarget targetDir
    -DappName app_name
```

The first project in the source list is considered the root project and only those services are exposed as Composite Services. Anytime you use the merge feature, it is recommended that the projects be related.

Merging of projects is supported for BPEL projects only. ESB projects cannot be merged with other BPEL or other ESB projects.

9.4.8.6 Upgrading Oracle Enterprise Service Bus (ESB) Projects with the Oracle SOA Suite Command-Line Upgrade Tool

The Oracle SOA Suite Command-Line Upgrade Tool can also be used to upgrade ESB projects to Oracle Mediator 11*g*.

To upgrade an ESB project, use the instructions in Section 9.4.8.4, "Upgrading BPEL or ESB Projects with the Oracle SOA Suite Command-Line Upgrade Tool", but be sure to use the mediator argument, as follows:

```
ant -f ORACLE_HOME/bin/ant-sca-upgrade.xml
    -Dsource sourceDir
    -Dtarget targetDir
    -DappName app_name
```

```
mediator
```

9.4.8.7 Upgrading Domain Value Maps (DVMs) and Cross References (XREFs) in the ESB Repository

If you use domain value maps (DVMs) or Cross References in your Oracle BPEL Process Manager 10*g* or Oracle Enterprise Service Bus 10*g* projects, then note the following:

- The xPath functions you use to access the domain value maps or cross references are upgraded automatically to Oracle BPEL Process Manager and Oracle Mediator 10*g* when you open and upgrade your applications in Oracle JDeveloper 11*g*.

- However, you must perform a manual upgrade task to upgrade the domain value maps and cross references that are saved in the Oracle Enterprise Service Bus repository. The upgrade process moves the domain value maps from the ESB repository to the Oracle Fusion Middleware 11*g* Metadata Services (MDS) repository.

  For more information, see "Managing the MDS Repository" in the *Oracle Fusion Middleware Administrator's Guide*.

To upgrade your 10*g* domain value maps in the ESB repository perform the following tasks.

Task 1 Export the domain value map metadata to a ZIP file

1. Change directory to the Oracle Enterprise Service Bus Oracle home.

2. Use the export script to export the metadata to a ZIP file.

    For example, on UNIX systems:

    ```
    ORACLE_HOME/export.sh metadata10g.zip
    ```

Task 2 Convert the ZIP file to an Oracle SOA Suite archive file

Use Apache Ant and the `upgrade-xrefdvm` target in the `sca-upgrade.xml` file to use the metadata ZIP file to generate an Oracle SOA Suite archive JAR file:

1. Change directory to the Oracle SOA Suite 11*g* Oracle home.

2. Use the following command to generate a SOA archive file, which will automatically be called `sca_XrefDvmFiles10g_rev1.0.jar`:

    ```
    ant -f ant-sca-upgrade.xml upgrade-xrefdvm
        -Dsource=location_of_the_zip_file
        -Dtarget=location_of_the_soa_archive
    ```

    For example:

    ```
    ant -f ant-sca-upgrade.xml upgrade-xrefdvm
        -Dsource=$ORACLE_HOME/temp/upgrade/metadata10g.zip
        -Dtarget=$ORACLE_HOME/temp/upgrade
    ```

Task 3 Import the archive file into MDS repository

1. Start Oracle JDeveloper 11*g* and create a new application.

2. Import the Oracle SOA Suite archive into a new SOA project:

    a. From the Oracle JDeveloper 11*g* **File** menu, select **Import**, then **SOA Archive into SOA Project**.

    b. In the Create SOA Project from SOA Archive Dialog Box, select JAR Options in the navigation tree on the left, and then click **Browse** to locate the `sca_XrefDvmFiles10g_rev1.0.jar` file that you created previously in this procedure.

    c. Select **File Groups > Project Output** from the navigational tree on the left, and enter `XrefDvmFiles10g` in the **Target Directory in Archive** field.

    d. Click **OK** to create the new SOA project, which is called `XrefDvmFiles10g`.

    The new project consists of an empty composite, along with the upgraded XRef and DVM files.

3. Create a JAR file for the XRef and DVM metadata, and then deploy the JAR file to the Oracle SOA Infrastructure.

For more information, see "Deploying and Using Shared Metadata Across SOA Composite Applications," in the *Oracle Fusion Middleware Fusion Developer's Guide for Oracle Application Development Framework*.

After you upgrade the DVM and cross reference metadata to 11*g*, you can then use Oracle JDeveloper 11*g* to modify the entries in the `XrefDvmFiles10g` project as needed. Each time you make changes, you must then transfer them to the proper MDS repository using the same deployment process.

9.4.9 Upgrading Oracle SOA Suite 10*g* Cross-References (XREF) Runtime Data to 11*g*

If you have Oracle SOA Suite 11*g* cross reference (XREF) runtime data in an Oracle database and you want to use it with Oracle SOA Suite 11*g*, then you can use the following procedure to upgrade it for use with Oracle SOA Suite 11*g*.

This procedure and utility are necessary because the XREF database schema table (`XREF_DATA`) has changed for Oracle Fusion Middleware 11*g*. Perform the steps described in this section to modify the 10*g* XREF data so it adheres to the 11*g* schema format.

Note that this procedure assumes your 10*g* XREF data is in one Oracle database and your new Oracle Fusion Middleware 11*g* data is in another Oracle database. This scenario is known as an "out-of-place" data upgrade.

Refer to the following sections for more information:

- Procedure for Upgrading the XREF Runtime Data

- Additional Tasks and Considerations When Upgrading the XREF Runtime Data

9.4.9.1 Procedure for Upgrading the XREF Runtime Data

To upgrade the cross reference runtime data from Oracle SOA Suite 10g to Oracle SOA Suite 11g:

1. Locate the XREF 10*g* to 11*g* Data Upgrade Utility, and unzip it into a local directory.

This utility can be found in the following directory when you download and unpack the Oracle Fusion Middleware Repository Creation Utility (RCU) archive file:

`RCU_HOME/rcu/integration/soainfra/sql/xref/XREF10gto11gDataUpgradeUtility.zip`

2. Log in with SYS user privileges to the Oracle database where your Oracle SOA Suite 11*g* XREF runtime data is stored, and create a database link for accessing the second database where your XREF 10*g* data is stored.

For example:

```
CREATE PUBLIC DATABASE LINK dblink10g
    CONNECT TO 10g_db_user_name
    IDENTIFIED BY 10g_db_password
    USING '(description=(address=(protocol=TCP)(host=10g_db_hostname)
            (port=10g_db_listening_port))(connect_data=(sid=10g_db_SID)))'
```

In the previous example, replace the values shown in italics with the information required to connect to the database where your XREF data is stored.

3. Change directory to following subdirectory, which was created inside the local folder where you unzipped the XREF 10*g* to 11*g* Data Upgrade Utility:

```
XREF10gto11gDataUpgradeUtility/scripts
```

4. Use a text editor to open the following SQL file:

```
Upgrade10gXrefTo11gXref.sql
```

5. Make the following edits to contents the file:

   a. Locate the following entry in the `Upgrade10gXrefTo11gXref.sql` file and change the [*TO BE FILLED*] string in the file to an `oramds:` URL that points to the shared MDS data location:

   ```
   --specify the MDS_FOLDER_LOCATION where the shared xref artifacts should
   reside,
     -- 10g table name will get prefixed by this location.
     -- MDS_FOLDER_LOCATION  varchar2(1000) :=
   'oramds:/abc/ABCMetaData/xref/';
       MDS_FOLDER_LOCATION  varchar2(1000) := '[TO BE FILLED]';
   ```

   For more information about the `oramds:` URL protocol, see "How to Deploy Shared Metadata" in the *Oracle Fusion Middleware Developer's Guide for Oracle SOA Suite*.

   b. Optionally, change the default value of the `commitperbatch` parameter (the default is 2000).

   This parameter specifies the number of records which should be committed per batch during migration.

6. From the same directory, log in to Oracle database that hosts the Oracle SOA 11g XREF data as the XREF user, and enter the following:

```
set serverout on;
@Upgrade10gXrefTo11gXref.sql
```

These commands upgrade the data and show you the progress of the upgrade as server output log.

Alternatively, the logs can be spooled to a file:

```
spool on;
spool path_of_local_file;
@Upgrade10gXrefTo11gXref.sql
spool off;
```

9.4.9.2 Additional Tasks and Considerations When Upgrading the XREF Runtime Data

In cases where the 10*g* XREF data does not conform to the 11*g* schema, or if any errors or exceptions occur during the upgrade process, the utility copies any corrupted

records to the XREF_DATA_CORRUPTED table. This table has all the columns of XREF_DATA, with an additional column called EXCEPTION_CAUSE. This column stores the reason for the exception with the row number.

When you are finished upgrading the XREF data, remove the database link using the following SQL command as the SYS user:

```
DROP PUBLIC DATABASE LINK dblink10g;
```

If any records are not migrated successfully, the temporary tables XREF_DATA_ CORRUPTED and TEMP_UPGRADE_XREF_LOG are not removed by the utility and must be cleaned up manually.

10

Upgrading Oracle BPEL Process Manager Applications

This chapter provides important supplementary information upgrading Oracle BPEL Process Manager applications to Oracle Fusion Middleware 11*g*.

> **Note:** All 10*g* BPEL processes are upgraded to BPEL 1.1 only.

Use Chapter 8, "Overview of Upgrading Oracle SOA Suite, WebCenter, and ADF Applications" for the tasks required to upgrade any Oracle SOA Suite, WebCenter, and ADF application.

Use the following sections to learn about the upgrade tasks you will likely need to perform to upgrade your Oracle BPEL Process Manager applications and projects to Oracle Fusion Middleware 11*g*:

- Manual Upgrade Steps After Migrating Oracle BPEL Process Manager Applications
- Additional Considerations for Oracle BPEL Process Manager Applications

10.1 Manual Upgrade Steps After Migrating Oracle BPEL Process Manager Applications

The following sections describe some of the common upgrade tasks you will have to perform after you open your Oracle Application Server 10*g* Oracle SOA Suite application in Oracle JDeveloper 11*g*.

These tasks represent items that are not upgraded automatically by the Oracle JDeveloper Migration Wizard:

- Providing Missing Binding Information for External References
- Correcting Problems With Oracle BPEL Process Manager Test Suites
- Using Oracle BPEL Process Manager Deployment Plans After Upgrade
- Upgrading Fault Policies in an Oracle BPEL Process Manager Project
- Upgrading a 10g Project With No BPEL Folder
- Post-Upgrade Steps for Projects That Use WSIF Bindings to EJBs
- Upgrade Different Bindings

10.1.1 Providing Missing Binding Information for External References

This section describes how to fix compiler warning messages about missing bindings after you upgrade an Oracle BPEL Process Manager project in Oracle JDeveloper.

Why the Warning Messages Occur   When a source project in your application depends on Oracle Application Server 10*g* Release 3 (10.1.3) Oracle BPEL Process Manager or Oracle Enterprise Service Bus (ESB) services, then as a pre-cursor to migration, the Oracle JDeveloper Migration Wizard creates local copies of their abstract WSDLs.

The abstract WSDLs, by definition, lack service and port endpoint information. During upgrade, when the Migration Wizard creates external composite references for the service dependencies, the references are created as abstract references with no binding information.

These abstract references are flagged with WARNING messages in the upgrade log. After upgrade, when you attempt to compile the projects, Oracle JDeveloper will generate compilation errors alerting you to the presence of abstract references in the composite.

Using 10*g* Release 3 (10.1.3) or 11*g* Dependency Services   Based on the upgrade and deployment schedules of all the services in the dependency tree, you can choose to continue to use 10*g* Release 3 (10.1.3) dependency services or proceed with the upgrade and redeployment of the dependency service.

In general, Oracle recommends that users study their source project and their dependencies before any upgrade work. This analysis will result in a smoother upgrade experience.

How to Correct Binding Errors Discovered During the Recompile   To correct binding errors due to the abstract references, right-click on the reference node in the Oracle JDeveloper 11*g* composite editor and choose the correct concrete WSDL. If you want to continue using a 10*g* Release 3 (10.1.3) dependency service, you can do so. Some of the dependency service endpoint URLs that are captured in the upgrade log can be used for this step.

You can also redesign the dependency in Oracle JDeveloper 11*g*, using the WSIL Browser (Resource Palette), and discover the service from the UI in the composite modeler.

For more information, refer to the *Oracle Fusion Middleware Developer's Guide for Oracle SOA Suite*.

10.1.2 Correcting Problems With Oracle BPEL Process Manager Test Suites

During upgrade in Oracle JDeveloper, BPEL test suites are converted to composite test suites. However, only the "instance initiate" action is migrated. This results in a composite test which initiates only a test run. The rest of any BPEL-based test actions are not automatically migrated and must be manually set up after the upgrade.

To set up the tests in your 11*g* environment, refer to "Testing SOA Composite Applications" in the *Oracle Fusion Middleware Developer's Guide for Oracle SOA Suite*.

10.1.3 Using Oracle BPEL Process Manager Deployment Plans After Upgrade

Deployment and configuration plans that you set up in Oracle Application Server 10*g* for your Oracle BPEL Process Manager user projects are not upgraded in Oracle JDeveloper 11*g*.

In Oracle Fusion Middleware11*g*, a composite can be accompanied by deployment and configuration plan during deployment. However, you must create the 11*g* deployment plan manually, after you have migrated the application in Oracle JDeveloper.

For more information about creating configuration plans in 11*g*, see "Moving SOA Composite Applications to and from Development, Test, and Production Environments" in the Oracle Fusion Middleware Developer's Guide for Oracle SOA Suite.

10.1.4 Upgrading Fault Policies in an Oracle BPEL Process Manager Project

When you open your 10*g* application in Oracle JDeveloper and use the Oracle JDeveloper Migration Wizard to upgrade the application, any fault policies and bindings that you set up in a user project are not automatically upgraded to 11*g*.

The reasons are as follows:

- In Oracle BPEL Process Manager 10*g*, fault policies were stored with the server and bindings were either stored with the server or specified in `bpel.xml`. In Oracle BPEL Process Manager 11*g*, fault policies and fault bindings are stored in the Oracle JDeveloper 11*g* project.

- In general, fault policies and bindings in Oracle BPEL Process Manager 11*g* are different than those in previous releases. For example, the file name and syntax is different, fault policies apply both to Oracle Mediator and Oracle BPEL Process Manager, and bindings are at a reference, component, and composite level in 11*g*.

For these reasons, you must manually recreate the fault policies and bindings in the Oracle JDeveloper 11*g* project after you upgrade your applications.

Similarly, when you upgrade an Oracle Enterprise Service Bus 10*g* Release 3 (10.1.3) project to 11*g*, you must add the retry parameters defined in `esb_config.ini` file to the 11*g* `fault-policy.xml` file.

For more information, see:

- "Using Fault Handling in a BPEL Process" in the *Oracle Fusion Middleware Developer's Guide for Oracle SOA Suite*

- "Using Mediator Error Handling" in the *Oracle Fusion Middleware Developer's Guide for Oracle SOA Suite*.

10.1.5 Upgrading a 10*g* Project With No BPEL Folder

If the 10*g* Release 3 (10.1.3) application you are upgrading includes a project where all the project artifacts are in one folder and the BPEL artifacts are not separated into their own folder in the project, then the Oracle JDeveloper Migration Wizard will not be able to upgrade those artifacts.

To avoid this problem, do one of the following:

- Before upgrading the project, use Oracle JDeveloper 10*g* to create a BPEL folder and move all the artifacts (except the build.xml and build.properties files) into that folder. Then you can proceed with upgrading the project.

- Use the Oracle SOA Suite command-line upgrade tool. The command-line tool will upgrade the artifacts even if a BPEL folder does not exist in the project. For more information, see Section 9.4.8, "Using the Oracle SOA Suite Command-Line Upgrade Tool".

10.1.6 Post-Upgrade Steps for Projects That Use WSIF Bindings to EJBs

When you have Oracle SOA Suite 10*g* projects that use WSIF bindings to communicate with Enterprise Java Beans (EJBs), then you must make some modifications to those projects after they are upgraded to 11*g*.

For example, after you upgrade a project that uses WSIF bindings and EJBs, it will fail to run on due to WSIF binding errors.

To work around the these problems, you typically must perform the following steps after you have upgraded the project to Oracle JDeveloper 11*g*:

1. Replace the properties of the <binding.wsif> entry in the composite.xml file with the correct JNDI related properties for Oracle WebLogic Server.

   For example, the following are typical properties set for a Oracle WebLogic Server domain for an application called the `BankTransferDemo` application, which was deployed previously on Oracle Application Server 10*g* Release 3 (10.1.3):

```
<binding.wsif
   port="......."
   location="......">
   .
   <property name="jndiName">
       ejb/session/BankTransfer
   </property>
   .
   <property name="java.naming.factory.initial">
     weblogic.jndi.WLInitialContextFactory
   </property>
   .
   <property name="java.naming.provider.url">
     t3://[SERVER HOST NAME]:[SERVER PORT]
   </property>
   .
   <property name="java.naming.security.principal">
       [DOMAIN ADMIN USER NAME]
   </property>
   .
   <property name="java.naming.security.credentials">
       [DOMAIN ADMIN PASSWORD]
   </property>
   .
</binding.wsif>
```

2. Copy the EJB classes to the following directory in your newly upgraded 11*g* project:

   `SOA_JDEV_PROJECT_HOME/SCA-INF/classes/`

For more information about using Oracle JDeveloper to modify your Oracle SOA Suite applications, see the *Oracle Fusion Middleware Developer's Guide for Oracle SOA Suite*.

10.1.7 Upgrade Different Bindings

Oracle SOA Suite 10*g* supported a number of WSIF bindings. You can only upgrade EJB and Java bindings to Oracle SOA Suite 11*g*. To upgrade EJB bindings complete the steps described in Section 10.1.6, "Post-Upgrade Steps for Projects That Use WSIF Bindings to EJBs".

The HTTP WSIF binding is replaced in Oracle SOA Suite 11*g* with a variation of `binding.ws`. This is migrated using the upgrade tool in Oracle JDeveloper.

For all other WSIF binding, only the abstract reference is migrated. The binding has to be manually completed to a JCA binding (example JMS WSIF) or a web services binding (example JAXRPC or Axis WSIF).

For more information, see "Getting Started with Binding Components" in the *Oracle Fusion Middleware Developer's Guide for Oracle SOA Suite*.

10.2 Additional Considerations for Oracle BPEL Process Manager Applications

The following sections describe additional considerations you should review after upgrading Oracle BPEL Process Manager applications:

- Section 10.2.1, "Verifying New and Deprecated Properties in the bpel.xml Deployment Descriptor"
- Section 10.2.2, "Upgrading User-Defined (Custom) XPath Functions in an Oracle BPEL Process Manager Project"
- Section 10.2.3, "Change in Support for Multiple BPEL Implementations"
- Section 10.2.4, "Verifying the Properties File Name When Upgrading Projects That Use Correlation Sets in an Asynchronous Service."
- Section 10.2.5, "Upgrading Projects With the transaction=participate Property"
- Section 10.2.6, "Specifying Domain Descriptor Properties in Oracle BPEL Process Manager 11g"
- Section 10.2.7, "Upgrading Applications That Use Multiple Oracle BPEL Process Manager Domains"
- Section 10.2.8, "Using Encryption and Decryption"
- Section 10.2.9, "Verifying Process ID, Project Name, and BPEL File Name Conventions Before Upgrade"
- Section 10.2.10, "Verifying Oracle BPEL Process Manager 11g Namespace Prefixes in XPath Functions"
- Section 10.2.11, "Upgrading Oracle BPEL Process Manager Projects with Multiple Tasks"
- Section 10.2.12, "Upgrading Dynamic Partner Links"

10.2.1 Verifying New and Deprecated Properties in the bpel.xml Deployment Descriptor

If any of your Oracle Application Server 10*g* Release 3 (10.1.3) applications reference any properties that were stored in the `bpel.xml` deployment descriptor, note that these properties are now set in the `composite.xml` deployment descriptor file.

For more information about the properties available in the `composite.xml` deployment descriptor, as well as important information about properties that are no longer supported in Oracle Fusion Middleware 11*g*, refer to "Deployment Descriptor Properties" in the *Oracle Fusion Middleware Developer's Guide for Oracle SOA Suite*.

10.2.2 Upgrading User-Defined (Custom) XPath Functions in an Oracle BPEL Process Manager Project

When you open and upgrade your applications in Oracle JDeveloper 11*g*, any custom XPath functions in the application projects are not upgraded automatically.

As a result, after you upgrade your application, you must copy any XPath function classes into the server classpath and register the function in the server configuration file. This is a manual step because the Oracle JDeveloper Migration Wizard cannot assume the server information.

For more information, refer to "Creating User-Defined XPath Extension Functions" in the *Oracle Fusion Middleware Developer's Guide for Oracle SOA Suite*.

10.2.3 Change in Support for Multiple BPEL Implementations

You cannot upgrade a single BPEL file with multiple implementations in Oracle BPEL Process Manager 11*g*.

To migrate and merge multiple 10*g* Release 3 (10.1.3) projects with a single BPEL implementation file, you must rename the BPEL file in each project. In the bpel.xml file in each project, the bpel filename and deployment ID name must be renamed the same name, as well.

Before you upgrade the application, review the `bpel.xml` file and `*.bpel` file. Make sure that the BPEL file name and the process name are the same; then review the `bpel.xml` file and verify that the BPEL file name is the same as the deployment ID name.

10.2.4 Verifying the Properties File Name When Upgrading Projects That Use Correlation Sets in an Asynchronous Service.

If you created an Oracle BPEL Process Manager 10*g* project that used correlation sets in an asynchronous service, as described in the Oracle BPEL Process Manager 10*g* Release 3 (10.1.3) Developer's Guide, make sure that the correlation set properties file uses the following proper naming convention:

*BPEL_FILE_NAME_*Properties.wsdl

If this file naming convention is not followed in the Oracle BPEL Process Manager 10*g* project, the properties will not be upgraded correctly. If you need to change the name to meet the file naming conventions, then also update the `wsdl` file that imports the property file name, accordingly.

10.2.5 Upgrading Projects With the transaction=participate Property

If you open and upgrade an Oracle BPEL Process Manager 10*g* project that has a configuration property, `transaction=participate`, then that property will not be migrated to 11*g* when you open the application in Oracle JDeveloper 11*g*. This configuration property is no longer supported in Oracle Fusion Middleware 11*g*.

If you open and upgrade an Oracle BPEL Process Manager 10g project has a partnerLink binding property, `transaction=participate`, then that property will be dropped on the caller side. The callee project instead should have the property setting bpel.config.transaction=required. This caller to callee setting propagation is not performed by upgrade and must be performed manually after you upgrade the application and before you attempt to deploy it on Oracle Fusion Middleware 11g.

10.2.6 Specifying Domain Descriptor Properties in Oracle BPEL Process Manager 11*g*

In Oracle BPEL Process Manager 10*g*, you specify domain properties in the `domain.xml` descriptor file.

In Oracle BPEL Process Manager 11*g*, you specify Oracle BPEL Process Manager engine properties in `bpel-config.xml` and `soa-infra-config.xml` (both are accessible as MBeans).

Table 10–1 compares the domain descriptor properties available in Oracle BPEL Process Manager 10*g* and Oracle BPEL Process Manager 10*g*.

There is only one domain in Oracle BPEL Process Manager 11*g*, which means that these properties will affect the entire server.

Table 10–1   Domain Descriptor Properties in Oracle BPEL Process Manager 10g and 11g

Oracle BPEL Process Manager 10*g* in domain.xml file	Oracle BPEL Process Manager 11*g* MBean Property	Description
auditDetailThreshold	auditDetailThreshold	The maximum size (in bytes) of an audit trail details string before it is stored separately from the audit trail.
		If a details string is larger than this threshold, then it will not be immediately loaded when the audit trail is initially retrieved; a link will be displayed with the size of the details string.
		Typically, the details string will contain the contents of a BPEL variable. In cases where the variable is very large, performance may be severely impacted by logging it to the audit trail. The 10g default value is 50 kilobytes.
auditLevel	auditLevel	Controls the amount of audit events logged by a process.
		The supported logging levels are:
		■ **off** - absolutely no logging performed whatsoever; may result in a slight performance boost for processing instances.
		■ **minimal** - all events are logged; however, no audit details are logged.
		■ **production** - all events are logged. The audit details for *assign* activities are not logged; the details for all other nodes are logged.
		■ **development** - all events are logged; all audit details for all additives are logged. The 10g default value is "development".
bpelcClasspath	bpelcClasspath	Server-side BPEL process compiler classpath. Any user-specific classes or libraries used by a BPEL java exec node (that have not been packaged in the BPEL archive) need to be specified here so that the server-side BPEL process compiler can successfully compile the BPEL process.
datasourceJndi	datasourceJndi (Moved to soa-infra-config.xml)	The JNDI name for the domain data source. This data source may refer to any data source (JTA not required).

Table 10–1   (Cont.) Domain Descriptor Properties in Oracle BPEL Process Manager 10g and 11g

Oracle BPEL Process Manager 10*g* in domain.xml file	Oracle BPEL Process Manager 11*g* MBean Property	Description
deliveryPersistPolicy	deliveryPersistPolicy	Changes whether the delivery messages are persisted. Delivery messages include invoke message, callback message and subscription message. ■ on - delivery messages are persisted ■ off - delivery messages are kept only in memory. ■ off.immediate - applicable for asynchronous processes only; messages are delivered synchronously regardless of the interface. The 10g default value is "on".
dspEngineThreads	dspEngineThreads	The total number of threads that will be allocated to process engine dispatcher messages. Engine dispatch messages are generated whenever an activity must be processed asynchronously by the BPEL engine. If the majority of processed deployed on the BPEL server are durable with a large number of dehydration points (mid-process receive, onMessage, onAlarm, wait), greater performance may be achieved by increasing the number of engine threads. Note that higher threads counts may cause greater CPU utilization due to higher context switching costs. The 10g default value is 30 threads. Any value less than 1 thread will be changed to the default.
dspInvokeAllocFactor	N/A	The percentage of active threads that should be tasked to process incoming invocation messages. After a thread has finished processing a message it may be retasked to process an engine or invocation message, depending upon the current thread allocation situation. This property was replaced by `dspInvokeThreads` in 10*g* Release 3 (10.1.3.4). The 10.1.3.1.0 default value is 0.4 (40 percent).
dspInvokeThreads	dspInvokeThreads	The total number of threads allocated to process invocation dispatcher messages. Invocation dispatch messages are generated for each payload received by Oracle BPEL Server and are meant to instantiate a new instance.If the majority of requests processed by the engine are instance invocations (as opposed to instance callbacks), greater performance may be achieved by increasing the number of invocation threads. Higher thread counts may cause greater CPU utilization due to higher context switching costs. The default value is 40 threads. Any value less than 1 thread is changed to the default.
dspMaxThreads	N/A	The maximum number of active threads that will be processing messages during peak load times. This is the simplest way to throttle the performance of the domain. The maximum value is dependent upon the application server or OS configuration. This property was replaced by dsp(Engine|Invoke|System)Threads in 10.1.3.4.0. The 10.1.3.1.0 default value is 100 threads.

Table 10–1 (Cont.) Domain Descriptor Properties in Oracle BPEL Process Manager 10g and 11g

Oracle BPEL Process Manager 10*g* in domain.xml file	Oracle BPEL Process Manager 11*g* MBean Property	Description
dspMinThreads	N/A	The minimum number of active threads that will be processing messages during peak load times. If the current number of active threads is under this number the load factor is not taken into consideration when determining if a new thread should be allocated or not.
		This property was replaced by `dsp(Engine|Invoke|System)Threads` in 10.1.3.4.0. The 10.1.3.1.0 default value is 5 threads.
dspSystemThreads	dspSystemThreads	The total number of threads that will be allocated to process system dispatcher messages. System dispatch messages are general housecleaning tasks that are typically processed quickly by the server (for example, releasing stateful message beans back to the pool).
		Typically, only a small number of threads are required to handle the number of system dispatch messages generated during runtime. The 10g default value is 2 threads. Any value less than 1 thread will be changed to the default.
expirationMaxRetry	expirationMaxRetry	The maximum number of times a failed expiration call (wait/onAlarm) will be retried before failing. If the activity or instance the expiration call is targeting cannot be found, the call will be rescheduled again. The 10g default is 5.
expirationRetryDelay	expirationRetryDelay	The amount of time in seconds to wait before rescheduling a failed expiration call. The 10g default value is 120 seconds.
instanceKeyBlockSize	instanceKeyBlockSize	The size of the block of instance ids to allocate from the dehydration store during each fetch. Instance ids for instantiated instances are pre-allocated from the dehydration store and kept in-memory.
		Once all of the in-memory instance ids have been exhausted, the next request will retrieve the next block from the dehydration store. Generally, the higher the block size, the less impact fetching instance ids will have on the overall performance of the engine (as the cost of allocating the block is amortized over a larger number of instances).
		The only downside to a large block size is the gap between instances in a clustered installation (as each node will have its own in-memory block of instance ids) and wasting instance ids if the engine is shutdown and the block is not utilized fully. The 10g default value is 10000.
largeDocumentThreshold	largeDocumentThreshold	The maximum size (in bytes) a BPEL variable can be before it is stored in a separate location from the rest of the instance scope data.
		Large XML documents will severely impact the performance of the entire BPEL server if they are constantly read-in and written-out whenever processing on an instance must be performed.
		By writing the variable to separate storage the performance impact is limited to times when the variable is explicitly used in a BPEL activity. The 10g default value is 50 kilobytes.

Table 10–1   (Cont.) Domain Descriptor Properties in Oracle BPEL Process Manager 10g and 11g

Oracle BPEL Process Manager 10*g* in domain.xml file	Oracle BPEL Process Manager 11*g* MBean Property	Description
minBPELWait	minBPELWait	The minimum amount of time allowed for a BPEL wait duration. Wait times shorter than this duration will not be respected and will essentially be "skipped" altogether. The 10g default value is 2 seconds.
processCheckSecs	N/A	The number of seconds to wait since the last time the process container checked the bpel archive before checking it again. By "checking" we mean checking the last modified time stamp on the bpel archive for a particular process; if the specified number of seconds has passed and the bpel archive file has been modified since the last time we checked, the process will be refreshed from the new archive. If not enough time has passed since the last time the stale check was performed, the currently loaded process classes will be used. To disable process checking use a value of -1; in this case once a process has been loaded the container will never check if a newer version of the same process has been deployed. The 10g default value is 1 second. This property is not relevant in 11g anymore as the SOA infrastructure now handles deployment via HTTP only.
statsLastN	statsLastN	The size of the "most recently processed" request list. After each request is finished, statistics for the request are kept in a list. A value less than or equal to 0 will disable statistics gathering. NOTE: engine throughput may be affected slightly if this option is enabled. Please choose a reasonable value for n (1000 is the recommended value).
syncMaxWaitTime	syncMaxWaitTime	The maximum time the process result receiver will wait for a result before returning. Results from asynchronous BPEL processes are retrieved synchronously via a receiver that will wait for a result from the container. The 10g default value is 45 seconds.
txDatasourceJndi	txDatasourceJndi (moved to soa-infra-config.xml)	The JNDI name for the domain transactional data source. This data source must be configured for JTA support.
uddiLocation	oracle.soa.uddi.registry.inquiryUrl (moved to soa-infra-config.xml)	The inquiry URL of the Uddi v3 compliant registry
uddiPassword	oracle.soa.uddi.registry.password (moved to soa-infra-config.xml)	Set if UDDI is secured, password
uddiUsername	oracle.soa.uddi.registry.username (moved to soa-infra-config.xml)	Set if UDDI is secured, username
validateXML	validateXML	If set to "true" the engine will apply schema validation for incoming and outgoing XML documents. The 10g default value "false".

10.2.7 Upgrading Applications That Use Multiple Oracle BPEL Process Manager Domains

If you have developed and deployed your Oracle Application Server 10*g* BPEL projects using multiple Oracle BPEL Process Manager domains, then you should be aware that Oracle Fusion Middleware 11*g* does not support multiple BPEL domains.

If you are using multiple domains in Oracle Application Server 10*g*, then you have two options when you upgrade to Oracle Fusion Middleware 11*g*:

- You can install and configure multiple Oracle SOA Suite Oracle WebLogic Server domains. You can then map each of your Oracle BPEL Process Manager 10*g* domains to a separate Oracle WebLogic Server domain.

- You can deploy all your Oracle BPEL Process Manager projects to a single Oracle SOA Suite Oracle WebLogic Server domain.

For more information, refer to the *Oracle Fusion Middleware Installation Guide for Oracle SOA Suite and Oracle Business Process Management Suite*.

10.2.8 Using Encryption and Decryption

There are no equivalent properties for encryption and decryption in Oracle Fusion Middleware 11*g* that are similar to those provided in Oracle BPEL Process Manager 10.1.3.5. You can achieve encryption and decryption by attaching Oracle Web Services Manager policies. Oracle Web Services Manager is available in Oracle SOA Suite 11*g*.

For more information about policies, see the following documentation:

- "Enabling Security with Policies" in the *Oracle Fusion Middleware Developer's Guide for Oracle SOA Suite*.

- *Oracle Fusion Middleware Security and Administrator's Guide for Web Services*

- *Oracle Fusion Middleware Administrator's Guide for Oracle SOA Suite and Oracle Business Process Management Suite*

10.2.9 Verifying Process ID, Project Name, and BPEL File Name Conventions Before Upgrade

Before you upgrade your BPEL projects to Oracle SOA Suite 11*g*, verify that the process ID in the bpel.xml file, the name of the BPEL project, and the name of the BPEL file are all the same value. Otherwise, you might encounter errors during the upgrade of your project in Oracle JDeveloper 11*g* or some elements of your project may not upgrade correctly.

10.2.10 Verifying Oracle BPEL Process Manager 11*g* Namespace Prefixes in XPath Functions

In some cases, XPath function namespace prefixes used in Oracle SOA Suite 10*g* have changed in Oracle SOA Suite 11*g*. As a result, if you are using an XPath function in an upgraded application, you might encounter "function not found" errors.

For example, if your Oracle BPEL Process Manager application used the `ora:parseXML()` function in 10*g*, you might encounter the following error:

```
"Function of name: ora:parseXML not found. please check the spelling and try gain"
```

In this example, the namespace prefix for the `parseXML` function has changed to `oraext:parseXML` in Oracle SOA Suite 11*g*.

For more information, see "parseXML" in the "XPath Extension Functions" appendix of the *Oracle Fusion Middleware Developer's Guide for Oracle SOA Suite*.

10.2.11 Upgrading Oracle BPEL Process Manager Projects with Multiple Tasks

In Oracle SOA Suite 10*g*, Oracle BPEL Process Manager projects could contain multiple tasks that invoke the same partner link.

Example 10–1 shows an example of a 10*g* composite application that defines two tasks that invoke the same partner link. If you attempt to upgrade such an application to 11*g*, Oracle JDeveloper generates errors when you attempt to compile the project.

To address the problem, you must modify the project before or after you upgrade the application in Oracle JDeveloper 11*g* so that each task invokes a unique partner link.

For more information, see "Creating a Partner Link" in the *Oracle Fusion Middleware Developer's Guide for Oracle SOA Suite*.

Example 10–1   Oracle BPEL Process Manager 10g Project with Multiple Tasks That Use the Same Partner Link

```
<partnerLink myRole="TaskServiceCallbackListener" name="TaskService"
                partnerRole="TaskService"
 .
  .
   .
 <invoke name="initiateTask_MainTask_1" partnerLink="TaskService"
                   portType="taskservice:TaskService"
                   operation="initiateTask"
                   inputVariable="initiateTaskInput"
                   outputVariable="initiateTaskResponseMessage">
              <correlations>
                    <correlation initiate="yes" set="WorkflowTaskIdCor"
                          pattern="in"/>
              </correlations>
 </invoke>
 .
  .
   .
 <invoke name="initiateTask_SubTask_1"
                          partnerLink="TaskService"
                          portType="taskservice:TaskService"
                          operation="initiateTask"
                          inputVariable="initiateTaskInput"
                          outputVariable="initiateTaskResponseMessage">
 .
  .
   .
</invoke>
```

10.2.12 Upgrading Dynamic Partner Links

If you have Dynamic Partner Link defined in Oracle BPEL Process Manager 10*g* then after you upgrade to Oracle BPEL Process Manager 11*g* you must modify the Dynamic Partner Link as shown in Example 10–2:

Example 10–2   Oracle BPEL Process Manager 11g Dynamic Partner Link

```
<invoke name="Invoke1"
```

```
                      inputVariable="Invoke1_QueryAddressByAliasKey_InputVariable"
                      outputVariable="Invoke1_QueryAddressByAliasKey_OutputVariable"
                      partnerLink="Service1" portType="ns1:AddressPort"
                      operation="QueryAddressByAliasKey" bpelx:invokeAsDetail="no">
<bpelx:inputProperty name="wsa.to"
expression=""http://localhost:7001/Simulator/simulator""/>
<bpelx:inputProperty name="endpointURI"
expression=""http://localhost:7001/Simulator/simulator""/>
</invoke>
```

For more information, see "Using a Dynamic Partner Link at Runtime" in the *Oracle Fusion Middleware Developer's Guide for Oracle SOA Suite*.

11

Upgrading Oracle Business Activity Monitoring Applications

This chapter provides important supplementary information upgrading Oracle SOA applications to Oracle Fusion Middleware 11*g*.

Use Chapter 8, "Overview of Upgrading Oracle SOA Suite, WebCenter, and ADF Applications" for the tasks required to upgrade any Oracle SOA Suite, WebCenter, and ADF application.

Refer to the following sections for information on upgrade tasks associated with Oracle Business Activity Monitoring:

- Migrating Oracle BAM Users, Groups, and Other LDAP Information to 11g
- Verifying Oracle BAM Action Form Templates After Upgrade
- Upgrading Oracle BAM External Data Sources
- Upgrading Images in Oracle BAM Reports
- Using Oracle BAM Web Services in 11g
- Upgrading Oracle BAM Sensor Actions
- Upgrading Enterprise Link Plans to Oracle BAM 11g
- Additional Considerations When Upgrading to Oracle BAM 11g

11.1 Migrating Oracle BAM Users, Groups, and Other LDAP Information to 11*g*

In general, after upgrade of the Oracle BAM schema, any Oracle BAM 10*g* users will appear in the Oracle BAM Administrator as inactive if they don't exist in the Oracle Fusion Middleware 11*g* security provider. However, you can use Oracle BAM Administrator software to remap 10*g* users to new Oracle BAM 11*g* users.

For more information, see "Managing Oracle BAM Object Ownership," in *Oracle Fusion Middleware Administrator's Guide for Oracle SOA Suite and Oracle Business Process Management Suite*.

After upgrade, you might need to address the following for users and groups:

- Reset any Shared Folder permissions
- Reset any Data Object and Data Object folder permissions
- Reset any Distribution Lists
- Reset Security Filters

■ Reset Alerts actions to users and groups

Further, if your Oracle BAM 10*g* environment takes advantage of an LDAP directory server to store your Oracle BAM user names or other Oracle BAM information, then you should make sure that the LDAP directory you were using in Oracle Application Server 10*g* is associated with your new Oracle Fusion Middleware 11*g* environment.

For more information, see Section 6.7.1, "Configuring Oracle SOA Suite 11g to Use Identity Management Components".

11.2 Verifying Oracle BAM Action Form Templates After Upgrade

After you upgrade from Oracle BAM 10*g* to Oracle Fusion Middleware 11*g*, some of the Action Form templates you created in Oracle BAM 10*g* may not function correctly.

As a result, after you perform the upgrade, you should review the templates and verify that they are functioning correctly.

Specifically, use the following instructions to search for typical issues that might need to be corrected after the upgrade to Oracle Fusion Middleware 11*g*:

> **See Also:** "Creating Action Form Templates" in the *Oracle Fusion Middleware User's Guide for Oracle Business Activity Monitoring*

1. Open the Oracle BAM Start Page.

> **See Also:** "Starting the Oracle BAM Web Applications," in the *Oracle Fusion Middleware User's Guide for Oracle Business Activity Monitoring*

2. Start Oracle BAM Architect.

3. Select **Data Objects** from the menu.

4. Expand the **System** folder and open the **Views** folder.

5. Select the **Action Form Templates** data object and click **Contents**.

6. Click **Edit Contents**.

7. Click **Edit** next to the first row.

8. Copy the contents of the **FormInput** column to a text editor and remove any white space following an element. (in other words replace "> " with ">").

9. Search for the mixed case "ReportServer" and change to lower case "reportserver" to fix the links to the style sheets.

10. Copy and paste the updated text back into the **FormInput** column.

11. Repeat these steps for each entry in the **FormInput** column that contains data.

11.3 Upgrading Oracle BAM External Data Sources

External data sources defined in your Oracle BAM 10*g* environment are not upgraded to Oracle BAM 11*g*.

As a result, if you use external data sources in your Oracle BAM 10*g* environment, you must manually recreate those data sources in Oracle BAM 11*g*. This is because Oracle Fusion Middleware 11*g* supports a new set of database drivers, which are used to connect to an external database.

To configure external data sources for Oracle BAM 11*g*, refer to "Creating External Data Sources" in the *Oracle Fusion Middleware Developer's Guide for Oracle SOA Suite*.

11.4 Upgrading Images in Oracle BAM Reports

Any images you used in Oracle BAM 10*g* are not upgraded automatically to Oracle BAM 11*g*. As a result, you must manually move any of your Oracle BAM images to the Oracle BAM 11*g* before the images can be used in your Oracle BAM 11*g* reports.

- If you uploaded background images or inserted images into columnar views, you have two options to upgrade the images to Oracle BAM 11*g*:

    - **Option 1:** Copy all image files from the Oracle BAM 10*g* location to the Oracle BAM 11*g* location.

      Oracle BAM 10*g* location:

      ```
      ORACLE_HOME\BAM\images
      ```

      Oracle BAM 11*g* location:

      ```
      BAM_DOMAIN_HOME/servers/BAM_Managed_Server/
          /tmp/_WL_user/oracle-bam-11.1.1/
          /tmpdir/war/buildnumber/Images
      ```

      Note that the `Images` directory in Oracle BAM 11*g* is case-sensitive and must begin with an uppercase "I".

    - **Option 2:** Edit the Oracle BAM 11*g* reports and re-upload the image files.

      For more information, see "Setting Backgrounds for Reports" in the *Oracle Fusion Middleware User's Guide for Oracle Business Activity Monitoring*.

- If you manually placed image files in the "images" directory in Oracle BAM 10*g* and you referenced them in your Oracle BAM 10*g* HTML calculations, then do the following:

    1. Copy the Oracle BAM 10*g* directory to the Oracle BAM 11*g* directory location as described in **Option 1** provided earlier in this section.

    2. Edit your Oracle BAM 11*g* HTML calculation expressions and change any folder reference `images` to `Images`.

- If you manually placed image files in a custom folder in Oracle BAM 10*g* and you referenced them in HTML calculations, then you must do the following:

    1. Create custom folder in 11g relative to:

      ```
      BAM_DOMAIN_HOME/servers/BAM_Managed_Server/
          /tmp/_WL_user/oracle-bam-11.1.1/
          /tmpdir/war/buildnumber/Images
      ```

    2. Copy the 10*g* images to this new location.

    3. Edit your Oracle BAM 11*g* HTML calculation expressions and change any folder reference `images` to `Images`.

11.5 Using Oracle BAM Web Services in 11*g*

The following sections describe how Oracle BAM Web services support in Oracle BAM 10*g* is affected by upgrade to Oracle Fusion Middleware 11*g*:

- Using Web Services Clients with Oracle BAM 11g

- Invoking Web Services from Oracle BAM Alerts
- Using the ManualRuleFire Web Service in Oracle BAM 11g
- New Support for Web Services in Oracle BAM 11g

11.5.1 Using Web Services Clients with Oracle BAM 11*g*

If you have developed client applications that communicate via Web services with Oracle BAM 10*g* servers, then you can continue to use those clients after you upgrade to Oracle BAM 11*g*, with no changes to the client code.

This is possible because the `DataObjectOperations` Web service available in Oracle BAM 10*g* is still available in Oracle BAM 11*g*. The DataObjectOperations Web service has been renamed to `DataObjectOperations10131`, but the WSDL end point URL remains the same, so client processes can continue working with Oracle BAM 11*g*.

However, to ensure that your clients continue to work successfully with Oracle BAM 11*g*, Oracle recommends that you apply the latest patch set to any Oracle BPEL Process Manager 10*g* clients that are using Oracle BAM 11*g* environment.

> **Note:** At the time of this document was published, the latest patch set available for Oracle SOA Suite 10*g* and Oracle JDeveloper was 10*g* Release 3 (10.1.3.5).
>
> However, for more information on the latest patch sets required for Oracle Fusion Middleware 11*g* interoperability and upgrade, refer to the Oracle Fusion Middleware 11*g* Release Notes available in the Oracle Fusion Middleware documentation library on the Oracle Technology Network (OTN):
>
> `http://www.oracle.com/technology/documentation/`

If you are using the latest version of Oracle JDeveloper 10*g* Release 3 (10.1.3), to create a connection to Oracle BAM 11*g*, the **Domain** field in the BAM connection wizard is completely ignored and can be left blank.

For more information about the DataObjectOperations10131 Web service in Oracle BAM 11*g*, refer to "Using Oracle BAM Web Services" in the *Oracle Fusion Middleware Developer's Guide for Oracle SOA Suite*.

11.5.2 Invoking Web Services from Oracle BAM Alerts

In Oracle BAM 10*g*, you could configure alerts to invoke a Web service by invoking an external action. This method of using alerts to invoke a Web service is no longer supported in Oracle BAM 11*g*.

Instead, in Oracle BAM 11*g*, you can create an alert rule that calls the "Call a Web Service" action.

For more information, see "Creating Oracle BAM Alerts" in the *Oracle Fusion Middleware Developer's Guide for Oracle SOA Suite*.

11.5.3 Using the ManualRuleFire Web Service in Oracle BAM 11*g*

If you used the ManualRuleFire Web service in Oracle BAM 10*g*, note that the end point URL for this Web service has changed in Oracle BAM 11*g*.

For more information, refer to "Using the ManualRuleFire Web Service" in the *Oracle Fusion Middleware Developer's Guide for Oracle SOA Suite*.

11.5.4 New Support for Web Services in Oracle BAM 11*g*

For complete information about the new Web services features in Oracle BAM 11*g*, refer to "Using Oracle BAM Web Services" in the *Oracle Fusion Middleware Developer's Guide for Oracle SOA Suite*.

11.6 Upgrading Oracle BAM Sensor Actions

When you open and upgrade a Oracle BPEL Process Manager 10*g* Release 3 (10.1.3) project in Oracle JDeveloper 11*g*, any Oracle BAM sensor actions are migrated automatically to 11*g*.

Unlike the behavior in 11*g*, Oracle BAM sensor actions in Oracle Fusion Middleware 11*g* use the Oracle BAM connection Factory JNDI (as shown in the BAM Sensor Action dialog box) to connect to Oracle BAM server. They do not use the bpel.xml file as they did in 10*g* Release 3 (10.1.3).

The migrated Oracle BAM 11*g* sensor actions can be identified by the `eis/bam/soap` connection Factory JNDI name. The Oracle BAM connection specified in the 10*g* `bpel.xml` file is migrated to an Application Resource BAM connection with the name of `Migrated1`.

All connection properties are preserved "as is" except for PORTS, which are set to 9001.

Note that Oracle JDeveloper 11*g* cannot connect to Oracle BAM 10*g* Release 3 (10.1.3). When you open an application in Oracle JDeveloper 11*g*, it is assumed that you have already installed and upgraded to Oracle BAM 11*g*.

Similarly, the Oracle BPEL Process Manager 11*g* runtime cannot connect to Oracle BAM 10*g* Release 3 (10.1.3).

11.7 Upgrading Enterprise Link Plans to Oracle BAM 11*g*

Enterprise Link for Oracle Business Activity Monitoring 10*g* is no longer available in Oracle BAM 11*g*. Instead, Oracle BAM 11*g* supports integration with Oracle Data Integrator (ODI), which replaces Enterprise Link.

When you upgrade to Oracle BAM 11*g*, you have the following options, depending on how you use Enterprise Link in Oracle BAM 10*g*:

- If used Enterprise Link primarily for integrating JMS messages into Oracle BAM, then you should review the Direct EMS functionality available in Oracle BAM 11*g*. Direct EMS allows JMS messages to be sent directly to Oracle BAM without the need for adapters or ODI.

  For more information, see "Creating Oracle BAM Enterprise Message Sources" in the *Oracle Fusion Middleware Developer's Guide for Oracle SOA Suite*.

- If your Oracle BAM 10*g* EnterpriseLink plans are more complex, then you must manually recreate them as ODI scenarios. Oracle BAM 11*g* supports all the Oracle BAM 10*g* transformations (for example, insert, update, delete, and so on) as ODI Knowledge Modules via the ODI integration with Oracle BAM 11*g*.

  For more information, see "Using Oracle Data Integrator With Oracle BAM" in the *Oracle Fusion Middleware Developer's Guide for Oracle SOA Suite*.

11.8 Additional Considerations When Upgrading to Oracle BAM 11*g*

The following sections describe additional information to consider when upgrading to Oracle BAM 11*g*:

- Using Oracle BAM Alert Actions After Upgrade
- Recreating Oracle BAM Enterprise Message Sources After Upgrade

11.8.1 Using Oracle BAM Alert Actions After Upgrade

Some Oracle BAM alert actions were removed in Oracle BAM 11g, and after upgrading, any alerts using them may become invalid.

Specifically, the "Run a Plan" action was removed because Oracle BAM replaced Enterprise Link with Oracle Data Integrator, which uses the "Run an ODI Scenario" alert action.

For more information, see Section 11.7, "Upgrading Enterprise Link Plans to Oracle BAM 11g".

11.8.2 Recreating Oracle BAM Enterprise Message Sources After Upgrade

Enterprise Message Sources have changed from Oracle BAM 10*g* to Oracle BAM 11*g*. As a result, you must recreate any Enterprise Message Sources after you upgrade to Oracle Fusion Middleware 11*g*.

For more information, see "Creating Oracle BAM Enterprise Message Sources" in the *Oracle Fusion Middleware Developer's Guide for Oracle SOA Suite*.

12

Upgrading Oracle B2B Applications

This chapter provides important supplementary information upgrading Oracle B2B applications to Oracle Fusion Middleware 11*g*.

Use Chapter 8, "Overview of Upgrading Oracle SOA Suite, WebCenter, and ADF Applications" for the tasks required to upgrade any Oracle SOA Suite, WebCenter, and ADF application.

The following sections provide the information you should know about the Oracle B2B upgrade, as well as some specific post-upgrade tasks you might have to perform after upgrading to Oracle B2B 11*g*.

- Features That are Not Upgraded to Oracle B2B 11g
- Features That are Changed After Upgrade to Oracle B2B 11g
- Oracle B2B 11g Post-Upgrade Tasks

12.1 Features That are Not Upgraded to Oracle B2B 11*g*

The following Oracle B2B features are not upgraded automatically when you upgrade to Oracle Fusion Middleware 11*g*:

- Host and Partner Security Certificates
- Callout Library JAR Files
- Exchange Protocol Parameter Identifiers
- The Receipt Recipient Property of the AS2-1.1 Channel
- Generic TCP Channels
- PositionalFlatFile Documents

12.1.1 Host and Partner Security Certificates

Oracle B2B host and partner security certificates are not upgraded into the expected format for Oracle B2B 11*g*.

Instead, you must copy the certificates from the root directory of the `export.zip` file (created in Section 6.5.2.4, "Performing the B2B Schema Upgrade") and import them into the keystore.

Note that Oracle B2B now saves certificates in the keystore, rather than an SSL wallet.

For more information, see the following:

- "Security Features of Oracle B2B" in the *Oracle Fusion Middleware User's Guide for Oracle B2B*

- "Configuring Identity and Trust" in the *Oracle Fusion Middleware Securing Oracle WebLogic Server*.

12.1.2 Callout Library JAR Files

If you created your own custom callout library jar files in Oracle B2B 10*g*, then those jar files are not upgraded to Oracle B2B 11*g*.

Instead, you must recreate your custom callout jar files using Oracle B2B 11*g*.

For more information, see "Managing Callouts" in the *Oracle Fusion Middleware User's Guide for Oracle B2B*.

12.1.3 Exchange Protocol Parameter Identifiers

In Oracle B2B 10*g*, you could specify identifiers for the exchange protocol parameter of a channel. Adding identifiers to the exchange protocol parameter is no longer supported in Oracle B2B 11*g*, and these identifiers are not upgraded to Oracle B2B 11*g*.

Instead, Oracle B2B 11*g*, you must recreate the required identifiers as part of a Trading Partner Profile.

For more information, see "Creating Trading Partner Profiles" in the *Oracle Fusion Middleware User's Guide for Oracle B2B*.

12.1.4 The Receipt Recipient Property of the AS2-1.1 Channel

If you set the **Receipt Delivery Option** property of the AS2-1.1 channel in your Oracle B2B 10*g* environment, then this property value is not upgraded to Oracle B2B 11*g*. After you upgrade, you must set this property accordingly.

For more information, see "Configuring Channels" in the *Oracle Fusion Middleware User's Guide for Oracle B2B*.

12.1.5 Generic TCP Channels

Generic TCP channels you created in Oracle B2B 10*g* are not upgraded automatically to Oracle B2B 11*g*. To create Generic TCP channels in Oracle B2B 11*g*, create an MLLP channel and specify TCP information. For more information, see the the section, "Generic Support for TCP" in "About MLLP" in the *Oracle Fusion Middleware User's Guide for Oracle B2B*.

12.1.6 PositionalFlatFile Documents

Any PositionalFlatFile documents (for example, SAP iDoc documents), created under the Custom Document protocol (using ECS files for translation and validation) in Oracle B2B 10*g* are not upgraded automatically to Oracle B2B 11g.

In Oracle B2B 11g, create these documents under a new document protocol called PositionalFlat file and specify the identification details and provide the ECS file in the document definition.

For more information, see the the section, "Using the Positional Flat File Document Protocol" in the *Oracle Fusion Middleware User's Guide for Oracle B2B*.

12.2 Features That are Changed After Upgrade to Oracle B2B 11*g*

The following Oracle B2B features behave differently or must be configured differently after you upgrade to Oracle B2B 11*g*:

- Document Protocol Revision and Document Type Fields

- Exchange Protocol Parameters for the MLLP Channel

- Channel Passwords

- Optional Fields in Oracle B2B 10g That are Mandatory in Oracle B2B 11g

- Inbound and Outbound Partner Agreements

- RosettaNet Outbound Messages

12.2.1 Document Protocol Revision and Document Type Fields

These two fields have been modified for Oracle B2B 11*g*. In Oracle B2B 10*g*, both these fields included a name and version. Oracle B2B uses only the Document Protocol Version and Document Type Name fields.

As a result, when you enqueue a document to the IP_OUT_QUEUE, note that you must provide a different set of attributes than those provided in Oracle B2B 10*g*.

For more information, see "MS Enqueue and Dequeue Utilities" in the *Oracle Fusion Middleware User's Guide for Oracle B2B*.

12.2.2 Exchange Protocol Parameters for the MLLP Channel

The following exchange protocol parameters in Oracle B2B 10*g* are now attributes of the MLLP channel:

- `ImmediateACK`

- `ImmediateACK-Custom-File`

- `ImmediateACK-mapAckControlID`

- `ImmediateACK-MapImmTriggerEvt`

- `DiscardHL7ACK`

If you set these parameters in Oracle B2B 10*g*, then you will need to set them as attributes of the MLLP channel.

For more information, see "Configuring Channels" in the *Oracle Fusion Middleware User's Guide for Oracle B2B*.

12.2.3 Channel Passwords

Oracle B2B 11*g* uses a new method for saving user credentials and password information. As a result, any passwords you entered for channels in 10*g* are not upgraded to Oracle B2B 11*g*.

Instead, after upgrade, you must enter the passwords as part of the channel parameters.

For more information, see "Configuring Channels" in the *Oracle Fusion Middleware User's Guide for Oracle B2B*.

12.2.4 Optional Fields in Oracle B2B 10*g* That are Mandatory in Oracle B2B 11*g*

Note that the following fields have changed from optional to required; as a result, after upgrade, you might have to provide values for these fields:

- If you were using EDI EDIFACT and EDI X12 documents in Oracle B2B 10*g*, then note that the **Group ID** identifier is now a required identifier.

- If you are using RosettaNet documents in Oracle B2B 10*g*, then note that the **Collaboration Code** and **Time to perform for Collaboration** fields are not required. In Oracle B2B 11*g*, they are required.

For more information, see "Creating Types" in the *Oracle Fusion Middleware User's Guide for Oracle B2B*.

12.2.5 Inbound and Outbound Partner Agreements

When you upgrade to Oracle B2B 11*g*, a partner agreement that exists in Oracle B2B 10g will be displayed in Oracle B2B 11*g* as two agreements: one agreement is for the inbound exchange; the other is for the outbound exchange.

For example, a 10g `Rosetta 3A4` agreement is split into the following:

- `Pip3A4PurchaseOrderConfirmation_Inbound`

- `Pip3A4PurchaseOrderRequest_Outbound`

For more information, see "Creating and Deploying Trading Partner Agreements" in the *Oracle Fusion Middleware User's Guide for Oracle B2B*.

12.2.6 RosettaNet Outbound Messages

The behavior of RosettaNet outbound messages has changed in Oracle B2B 11g.

When using RosettaNet with Oracle B2B 11g, you must now set an additional enqueue property (msgType=2) for any outbound message that needs to be treated as a response message (for example, a PurchaseOrderConfirmation message).

 Otherwise, the outbound message will be treated as a request message. In Oracle B2B 10g, this property was not necessary.

For more information, see "Utilities for Enqueuing and Dequeuing" in the *Oracle Fusion Middleware User's Guide for Oracle B2B*.

12.3 Oracle B2B 11*g* Post-Upgrade Tasks

The following tasks should be performed after you upgrade to Oracle B2B 11*g*:

- Configuring MLLP Channels for Inbound and Outbound Messages

- Migrating Oracle B2B Metadata

- Considerations When Upgrading B2B-Related Oracle BPEL Processes

12.3.1 Configuring MLLP Channels for Inbound and Outbound Messages

After upgrade, perform one of the following tasks, depending on how you configured the MLLP channel in Oracle B2B 10*g*:

- If you used the `singleDC` property in the `tip.properties` file to enable one MLLP channel to transfer both inbound and outbound messages, then you must create a new channel and then update the partner agreement to use the new channel.

- If you did not use the `singleDC` property, then set the connection mode of the host trading partner to **Server**.

For more information, see "About MLLP" in the *Oracle Fusion Middleware User's Guide for Oracle B2B*.

12.3.2 Migrating Oracle B2B Metadata

You can migrate your existing Oracle B2B 10*g* to Oracle B2B 11*g* using the command-line tool:

1. Export the metadata from the Oracle B2B repository using the `b2bexport` utility as described in "Exporting Data" in the *Oracle Fusion Middleware User's Guide for Oracle B2B*.

2. Import the B2B metadata ZIP file to the repository using the `b2bimport` utility as described in "Importing Data" in the *Oracle Fusion Middleware User's Guide for Oracle B2B*.

12.3.3 Considerations When Upgrading B2B-Related Oracle BPEL Processes

By default, Oracle SOA Suite 10*g* uses Oracle Streams Advanced Queuing (AQ) as the default connection between the Oracle BPEL Process Manager and Oracle B2B. The Oracle AQ connection relies on a data source being available on the OC4J server.

When you upgrade to Oracle SOA Suite 11*g*, the required data source used by Oracle AQ is not available by default. As a result, you should consider the following when upgrading to Oracle B2B 11*g* and Oracle SOA Suite 11*g*:

- Before you upgrade to Oracle B2B 11*g*, create a data source in Oracle WebLogic Server that references the Oracle B2B 10*g* database. Note the JNDI name you provide for the database.

  For more information, see "Create JDBC data sources" in the *Oracle Fusion Middleware Oracle WebLogic Server Administration Console Online Help*.

- After the upgrade to Oracle B2B 11*g*, open the Open AQ Adapter External Reference, and enter JNDI name for the new Oracle WebLogic Server data source on the Service Connection page of the Connection wizard.

Note that when you open your Oracle SOA Suite 10*g* application in Oracle JDeveloper 11*g*, the Oracle JDeveloper Migration Utilty upgrades your application and makes the following related changes to your projects. These changes allow you to continue using the Oracle AQ connection between the Oracle B2B and Oracle BPEL Process Manager processes:

- For Oracle BPEL Process Manager projects, the variables used by the Oracle AQ connection are upgraded automatically to properties within an Oracle BPEL Process Manager assign statement or transformation map within the BPEL process file.

- For Oracle Enterprise Service Bus projects, the variables used by the Oracle AQ connection in are removed automatically from the 10*g* XSLT map file and upgraded to properties within an assign values field in Oracle Mediator.

Alternatively, you can recreate your projects in Oracle JDeveloper 11*g*, using the newer Oracle Fusion Middleware 11*g* features for defining the connections between Oracle Mediator projects and Oracle BPEL Process Manager processes.

For more information, see the *Oracle Fusion Middleware Developer's Guide for Oracle SOA Suite*.

13

Upgrading Oracle Web Services Manager Policies and OC4J Security Enviroments

This chapter provides important supplementary information upgrading Oracle SOA applications to Oracle Fusion Middleware 11*g*.

Use Chapter 8, "Overview of Upgrading Oracle SOA Suite, WebCenter, and ADF Applications" for the tasks required to upgrade any Oracle SOA Suite, WebCenter, and ADF application.

Use the following sections to understand tasks specific to upgrading Oracle SOA applications:

- Section 13.1, "Upgrading Oracle Web Services Manager (WSM) Policies"
- Section 13.2, "Upgrading Oracle Containers for J2EE (OC4J) Security Environments"

13.1 Upgrading Oracle Web Services Manager (WSM) Policies

In Oracle WSM 10*g*, you specify *policy steps* at each policy enforcement point. Each policy step is a fine-grained operational task that addresses a specific security operation, such as authentication and authorization; encryption and decryption; security signature, token, or credential verification; and transformation. Each operational task is performed on either the Web service request or response.

For more details about the Oracle WSM 10*g* policy steps, see "Oracle Web Services Manager Policy Steps" in the *Oracle Web Services Manager Administrator's Guide* in the Oracle Application Server 10*g* Release 3 (10.1.3.1.0) documentation library at:

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

In Oracle WSM 11*g*, you attach *policies* to Web service endpoints. Each policy consists of one or more *assertions*, defined at the domain-level, that define the security requirements. A set of predefined policies and assertions are provided out-of-the-box. For more details about the predefined policies, see "Predefined Policy Reference" in *Security and Administrator's Guide for Oracle Web Services*.

13.1.1 Before You Upgrade

Before you upgrade Oracle WSM policies, you must perform the following tasks:

- Install Oracle WSM 11*g*. For more information, see the *Oracle Fusion Middleware Installation Guide for Oracle SOA Suite and Oracle Business Process Management Suite*.

- Upgrade your Oracle Containers for J2EE (OC4J) 10*g* Web services to Oracle WebLogic Server 11*g* Web services.

For more information, see "Task 6: Upgrade the Application Web Services" in the *Oracle Fusion Middleware Upgrade Guide for Java EE*.

13.1.1.1 A Note About Oracle WSM 10*g* Gateways

As described in "Examining the Rearchitecture of Oracle WSM in Oracle Fusion Middleware" in the *Oracle Fusion Middleware Security and Administrator's Guide for Web Services*, Oracle Fusion Middleware 11*g* Release 1 (11.1.1.6.0) does not include a Gateway component.

You can continue to use the Oracle WSM 10g Gateway components with Oracle WSM 10*g* policies in your applications. For information about Oracle WSM 10*g* interoperability, see the *Oracle Fusion Middleware Interoperability Guide for Oracle Web Services Manager*.

13.1.1.2 A Note About Third-party Software

As described in "Examining the Rearchitecture of Oracle WSM in Oracle Fusion Middleware" in the *Oracle Fusion Middleware Security and Administrator's Guide for Web Services*, Oracle WSM 10*g* supported policy enforcement for third-party application servers, such as IBM WebSphere and Red Hat JBoss. Oracle Fusion Middleware 11*g* Release 1 (11.1.1.6.0) only supports Oracle WebLogic Server.

You can continue to use the third-party application servers with Oracle WSM 10*g* policies. For information about Oracle WSM 10*g* interoperability, see the *Oracle Fusion Middleware Interoperability Guide for Oracle Web Services Manager*.

13.1.2 Upgrading Oracle WSM 10*g* Predefined Policies

Table 13–1 describes the most common Oracle WSM predefined policy upgrade scenarios based on the following security requirements: authentication and authorization, message protection, transport, and logging. A comparison of the steps required to implement each security requirement in both the Oracle WSM 10*g* and Oracle WSM 11*g* environments is provided.

For more information about:

- Attaching policies in Oracle Fusion Middleware 11*g*, see "Attaching Policies to Web Services" in *Oracle Fusion Middleware Security and Administrator's Guide for Web Services*.

- Oracle WSM 10*g* policy steps, see "Oracle Web Services Manager Policy Steps" in *Oracle Web Services Manager Administrator's Guide* in the Oracle Application Server 10*g* Release 3 (10.1.3.1.0) documentation library at:

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

Table 13–1   Upgrading Oracle WSM 10g Predefined Policies

Security Requirements	Oracle WSM 10*g*	Oracle WSM 11*g*
Anonymous authentication with message protection (WS-Security 1.0)	Attach policy steps as follows: ■ **Client**: Sign Message and Encrypt. ■ **Web service**: Decrypt and Verify Signature.	1. Attach policies as follows: **Client**: oracle/wss10_message_protection_client_policy. **Web service**: oracle/wss10_message_protection_service_policy. 2. Leave the default configuration set for message signing and encryption. 3. Disable the Include Timestamp configuration setting.
Anonymous authentication with message integrity (WS-Security 1.0)	Attach policy steps as follows: ■ **Client**: Sign Message. ■ **Web service**: Verify Signature.	1. Attach policies as follows: **Client**: oracle/wss10_message_protection_client_policy. **Web service**: oracle/wss10_message_protection_service_policy. 2. Configure the policy assertion for message signing only. 3. Disable the Include Timestamp configuration setting.
Anonymous authentication with message confidentiality (WS-Security 1.0)	Attach policy steps as follows: ■ **Client**: XML Encrypt. ■ **Web service**: XML Decrypt.	1. Attach policies as follows: **Client**: oracle/wss10_message_protection_client_policy. **Web service**: oracle/wss10_message_protection_service_policy. 2. Configure the policy assertion for message encryption only. 3. Disable the Include Timestamp configuration setting.
Username token with message protection (WS-Security 1.0)	Attach policy steps as follows: ■ **Client**: Insert WSBASIC Credentials and Sign Message and Encrypt. ■ **Web service**: Decrypt and Verify Signature, Extract Credentials (configured as WSBASIC), and FileAuthenticate. Note: You can substitute File Authenticate with LDAP Authenticate, Oracle Access Manager Authenticate, Active Directory Authenticate, or SetMinder Authenticate.	1. Attach policies as follows: **Client**: oracle/wss10_username_token_with_message_protection_client_policy. **Web service**: oracle/wss10_username_token_with_message_protection_service_policy. 2. Leave the default configuration set for message signing and encryption. 3. Disable the Include Timestamp configuration setting. 4. Configure the Authentication and Identity Assertion provider.

Table 13–1   (Cont.) Upgrading Oracle WSM 10g Predefined Policies

Security Requirements	Oracle WSM 10*g*	Oracle WSM 11*g*
Username token with message protection (WS-Security 1.0) and file authorization	Attach policy steps as follows: ■ **Client**: Insert WSBASIC Credentials and Sign Message and Encrypt. ■ **Web service**: Decrypt and Verify Signature, Extract Credentials (configured as WSBASIC), FileAuthenticate and File Authorize. Note: You can substitute File Authenticate with LDAP Authenticate, Active Directory Authenticate, or SetMinder Authenticate. Similarly, you can substitute File Authorize with LDAP Authorize, Active Directory Authorize, or SetMinder Authorize.	1. Attach policies as follows: **Client**: oracle/wss10_username_token_with_message_protection_client_policy. **Web service**: oracle/wss10_username_token_with_message_protection_service_policy and oracle/binding_authorization. 2. Leave the default configuration set for message signing and encryption. 3. Disable the Include Timestamp configuration setting. 4. Configure the Authentication and Identity Assertion provider.
ID propagation with SAML token (sender vouches) with message protection (WS-Security 1.0)	Attach policy steps as follows: ■ **Client**: SAML—Insert WSS 1.0 Sender-Vouches Token and Sign and Encrypt. ■ **Web service**: XML Decrypt and SAML—Verify WSS 1.0 Token.	1. Attach policies as follows: **Client**: oracle/wss10_saml_token_with_message_protection_client_policy. **Web service**: oracle/wss10_saml_token_with_message_protection_service_policy. 2. Disable the Include Timestamp configuration setting. 3. Leave the default configuration set for message signing and encryption.
HTTP basic authentication	Attach policy steps as follows: ■ **Client**: N/A. ■ **Web service**: Extract Credentials (configure as HTTP).	Attach policies as follows: ■ **Client**: oracle/wss_http_token_client_policy. ■ **Web service**: oracle/wss_http_token_service_policy.
Oracle Access Manager security (WS-Security 1.0)	Attach policy steps as follows: ■ **Client**: Insert Oracle Access Manager Token. ■ **Web service**: Extract Credentials and Oracle Access Manager Authenticate Authorize.	Attach policies as follows: ■ **Client**: oracle/wss_oam_token_client_policy. ■ **Web service**: oracle/wss_oam_token_service_policy.
Mutual authentication with message protection (WS-Security 1.0)	Attach policy steps as follows: ■ **Client**: Insert WSBASIC Credentials and Sign Message and Encrypt. ■ **Web service**: Decrypt and Verify Signature, Extract Credentials (configured as WSBASIC), and FileAuthenticate. Note: You can substitute File Authenticate with LDAP Authenticate, Oracle Access Manager Authenticate, Active Directory Authenticate, or SetMinder Authenticate.	1. Attach policies as follows: **Client**: oracle/wss10_x509_token_with_message_protection_client_policy. **Web service**: oracle/wss10_x509_token_with_message_protection_service_policy. 2. Leave the default configuration set for message signing and encryption. 3. Disable the Include Timestamp configuration setting. 4. Configure the Authentication and Identity Assertion provider.

Table 13–1   (Cont.) Upgrading Oracle WSM 10g Predefined Policies

Security Requirements	Oracle WSM 10*g*	Oracle WSM 11*g*
Username token over SSL	1. Configure the application server for SSL. 2. Attach policy steps as follows: **Client**: Insert WSBASIC Credentials. **Web service**: Extract Credentials and File Authenticate.	1. Configure the application server for SSL. 2. Attach policies as follows: **Client**: oracle/wss_username_token_over_ssl_client_policy. **Web service**: oracle/wss_username_token_over_ssl_client_service_policy. 3. Disable the Include Timestamp configuration setting.
ID propagation with SAML token (sender vouches) over SSL (WS-Security 1.0)	1. Configure the application server for SSL. 2. Attach policy steps as follows: **Client**: SAML—Insert WSS 1.0 Sender-Vouches Token. **Web service**: SAML—Verify WSS 1.0 Token.	1. Configure the application server for SSL. 2. Attach policies as follows: **Client**: oracle/wss_saml_token_over_ssl_client_policy. **Web service**: oracle/wss_saml_token_over_ssl_client_service_policy. 3. Disable the Include Timestamp configuration setting.
Log information	Attach the following policy step to the client or Web service: Log	Attach the following policy to the client or Web service: oracle/log_policy

13.1.3 Upgrading Oracle WSM Custom Policies

In Oracle WSM 10*g*, you create, develop, and deploy custom *policy steps* using the procedures described in the *Oracle Web Services Manager Extensibility Guide* in the Oracle Application Server 10*g* Release 3 (10.1.3.1.0) documentation library at:

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

In Oracle WSM 11*g*, you create, develop, and deploy custom *policy assertions*. You will need to redefine your custom policy steps as custom policy assertions using the procedures described in "Creating Custom Assertions" in *Security and Administrator's Guide for Oracle Web Services*.

13.2 Upgrading Oracle Containers for J2EE (OC4J) Security Environments

In OC4J 10*g*, you configure your security environment by modifying the contents of the XML-based deployment descriptor files. For complete details about securing OC4J environments, see *Oracle Application Server Web Services Security Guide* at:

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

In Oracle WSM 11*g*, you attach *policies* to Web service endpoints. Each policy consists of one or more *assertions*, defined at the domain-level, that define the security requirements. A set of predefined policies and assertions are provided out-of-the-box. For more details about the predefined policies, see "Predefined Policy Reference" in *Security and Administrator's Guide for Oracle Web Services*.

The following sections describe the most common OC4J upgrade scenarios based on the following security requirements: authentication, message protection, transport,

and logging. A comparison of the steps required to implement each security requirement in both the OC4J 10*g* and Oracle WSM 11*g* environments is provided.

- Section 13.2.1, "Before You Upgrade"

- Section 13.2.2, "Anonymous Authentication with Message Protection (WS-Security 1.0)"

- Section 13.2.3, "Anonymous Authentication with Message Integrity (WS-Security 1.0)"

- Section 13.2.4, "Anonymous Authentication with Message Confidentiality (WS-Security 1.0)"

- Section 13.2.5, "Username Token with Message Protection (WS-Security 1.0)"

- Section 13.2.6, "ID Propagation Using SAML Token (Sender Vouches) with Message Protection (WS-Security 1.0)"

- Section 13.2.7, "ID Propagation Using SAML Token (Holder of Key) with Message Protection (WS-Security 1.0)"

- Section 13.2.8, "Mutual Authentication with Message Protection (WS-Security 1.0)"

- Section 13.2.9, "Username token over SSL"

- Section 13.2.10, "ID Propagation with SAML Token (Sender Vouches) over SSL (WS-Security 1.0)"

- Section 13.2.11, "Log Information"

> **Note:** For information about configuring attaching policies in Oracle Fusion Middleware 11*g*, see *Oracle Fusion Middleware Security and Administrator's Guide for Web Services*.

The next section describes the prerequisites required before you upgrade.

13.2.1 Before You Upgrade

Before you upgrade the OC4J security environment, you must perform the following tasks:

- Install Oracle WSM 11*g*. For more information, see the *Oracle Fusion Middleware Installation Guide for Oracle SOA Suite and Oracle Business Process Management Suite*.

- Review "Task 6: Upgrade the Application Web Services" in the *Oracle Fusion Middleware Upgrade Guide for Java EE*.

  This section provides general information about upgrading OC4J Web services to Oracle WebLogic Server.

13.2.2 Anonymous Authentication with Message Protection (WS-Security 1.0)

The following sections describe how to implement authentication with message protection that conforms to the WS-Security 1.0 standard, and compare the steps required for the OC4J 10*g* and Oracle WSM 11*g* environments.

13.2.2.1 OC4J 10*g*

Edit the deployment descriptors for the Web service and client, as described in the following sections.

For more information about the deployment descriptor elements, see "OracleAS Web Services Security Schema" in *Oracle Application Server Web Services Security Guide* in the Oracle Application Server 10*g* Release 3 (10.1.3.1.0) documentation library at:

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

Web Service Client (with sample data)

Define the <signature> and <encrypt> elements in the client deployment descriptor. For example:

```
<signature>
   <signature-method>RSA-SHA1</signature-method>
   <tbs-elements>
      <tbs-element local-part="Body" name-space=
      "http://schemas.xmlsoap.org/soap/envelope/" />
      <tbs-element name-space=
"http://docs.oasis-open.org/wss/2004/01/oasis-200401-wss-wssecurity-utility-1.0.xs
d" local-part="Timestamp" />
   </tbs-elements>
   <add-timestamp created="true" expiry="28800" />
</signature>
<encrypt>
   <recipient-key alias="orakey"/>
   <encryption-method>AES-128</encryption-method>
   <keytransport-method>RSA-OAEP-MGF1P</keytransport-method>
   <tbe-elements>
      <tbe-element local-part="Body" name-space=
      "http://schemas.xmlsoap.org/soap/envelope/" mode="CONTENT" />
   </tbe-elements>
</encrypt>
```

Web Service (with sample data)

Define the <verify-signature> and <decrypt> elements in the service deployment descriptor. For example:

```
<verify-signature>
   <tbs-elements>
      <tbs-element name-space="http://schemas.xmlsoap.org/soap/envelope/"
      local-part="Body" />
      <tbs-element name-space=
"http://docs.oasis-open.org/wss/2004/01/oasis-200401-wss-wssecurity-utility-1.0.xs
d"
      local-part="Timestamp" />
   </tbs-elements>
   <verify-timestamp expiry="28800" created="true" />
</verify-signature>
<decrypt>
   <tbe-elements>
      <tbe-element
      name-space="http://schemas.xmlsoap.org/soap/envelope/"
      local-part="Body" mode="CONTENT" />
   </tbe-elements>
</decrypt>
```

13.2.2.2 Oracle WSM 11*g*

Perform the following steps:

1. Attach policies as follows:

   Client: oracle/wss10_message_protection_client_policy.

   Web service: oracle/wss10_message_protection_service_policy.

   For more information about attaching policies in Oracle Fusion Middleware 11*g*, see "Attaching Policies to Web Services" in *Oracle Fusion Middleware Security and Administrator's Guide for Web Services*.

2. Leave the configuration set for message body signing and encryption.

13.2.3 Anonymous Authentication with Message Integrity (WS-Security 1.0)

The following sections describe how to implement authentication with message integrity that conforms to the WS-Security 1.0 standard, and compare the steps required for the OC4J 10*g* and Oracle WSM 11*g* environments.

13.2.3.1 OC4J 10*g*

Edit the deployment descriptors for the Web service and client, as described in the following sections.

For more information about the deployment descriptor elements, see "OracleAS Web Services Security Schema" in *Oracle Application Server Web Services Security Guide* in the Oracle Application Server 10*g* Release 3 (10.1.3.1.0) documentation library at:

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

Web Service Client (with sample data)

Define the <signature> element in the client deployment descriptor. For example:

```
<signature>
   <signature-method>RSA-SHA1</signature-method>
   <tbs-elements>
      <tbs-element local-part="Body" name-space=
       "http://schemas.xmlsoap.org/soap/envelope/" />
      <tbs-element name-space=
"http://docs.oasis-open.org/wss/2004/01/oasis-200401-wss-wssecurity-utility-1.0.xs
d" local-part="Timestamp" />
   </tbs-elements>
   <add-timestamp created="true" expiry="28800" />
</signature>
```

Web Service (with sample data)

Define the <verify-signature> element in the service deployment descriptor. For example:

```
<verify-signature>
   <tbs-elements>
      <tbs-element name-space="http://schemas.xmlsoap.org/soap/envelope/"
       local-part="Body" />
      <tbs-element name-space=
"http://docs.oasis-open.org/wss/2004/01/oasis-200401-wss-wssecurity-utility-1.0.xs
d"
       local-part="Timestamp" />
   </tbs-elements>
   <verify-timestamp expiry="28800" created="true" />
</verify-signature>
```

13.2.3.2 Oracle WSM 11*g*

Perform the following steps:

1. Attach policies as follows:

   Client: oracle/wss10_message_protection_client_policy.

   Web service: oracle/wss10_message_protection_service_policy.

   For more information about attaching policies in Oracle Fusion Middleware 11*g*, see "Attaching Policies to Web Services" in *Oracle Fusion Middleware Security and Administrator's Guide for Web Services*.

2. Configure the policy assertion for message body signing only.

13.2.4 Anonymous Authentication with Message Confidentiality (WS-Security 1.0)

The following sections describe how to implement authentication with message confidentiality that conforms to the WS-Security 1.0 standard, and compare the steps required for the OC4J 10*g* and Oracle WSM 11*g* environments.

For more information about the deployment descriptor elements, see "OracleAS Web Services Security Schema" in *Oracle Application Server Web Services Security Guide* in the 10*g* Release 3 (10.1.3.1.0) documentation library at:

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

13.2.4.1 OC4J 10*g*

Edit the deployment descriptors for the Web service and client, as described in the following sections.

Web Service Client (with sample data)

Define the <encrypt> element in the client deployment descriptor. For example:

```
<encrypt>
   <recipient-key alias="orakey"/>
   <encryption-method>AES-128</encryption-method>
   <keytransport-method>RSA-OAEP-MGF1P</keytransport-method>
   <tbe-elements>
      <tbe-element local-part="Body" name-space=
       "http://schemas.xmlsoap.org/soap/envelope/" mode="CONTENT" />
   </tbe-elements>
</encrypt>
```

Web Service (with sample data)

Define the <decrypt> element in the service deployment descriptor. For example:

```
<decrypt>
   <tbe-elements>
      <tbe-element
       name-space="http://schemas.xmlsoap.org/soap/envelope/"
       local-part="Body" mode="CONTENT" />
   </tbe-elements>
</decrypt>
```

13.2.4.2 Oracle WSM 11*g*

Perform the following steps:

1. Attach policies as follows:

Client: oracle/wss10_message_protection_client_policy.

Web service: oracle/wss10_message_protection_service_policy.

For more information about attaching policies in Oracle Fusion Middleware 11*g*, see "Attaching Policies to Web Services" in *Oracle Fusion Middleware Security and Administrator's Guide for Web Services*.

2. Configure the policy assertion for message body encryption only.

13.2.5 Username Token with Message Protection (WS-Security 1.0)

The following sections describe how to implement username token with message protection that conforms to the WS-Security 1.0 standard, and compare the steps required in the OC4J 10*g* and Oracle WSM 11*g* environments.

13.2.5.1 OC4J 10*g*

Edit the deployment descriptors for the Web service and client, as described in the following sections.

For more information about the deployment descriptor elements, see "OracleAS Web Services Security Schema" in *Oracle Application Server Web Services Security Guide* in the 10*g* Release 3 (10.1.3.1.0) documentation library at:

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

Web Service Client (with sample data)

Define the <username-token>, <signature>, and <encrypt> elements in the client deployment descriptor. For example:

```
<username-token password-type="PLAINTEXT" add-nonce="false"
 add-created="false" />
<signature>
   <signature-method>RSA-SHA1</signature-method>
   <tbs-elements>
      <tbs-element local-part="Body" name-space=
       "http://schemas.xmlsoap.org/soap/envelope/"/>
      <tbs-element name-space=
"http://docs.oasis-open.org/wss/2004/01/oasis-200401-wss-wssecurity-utility-1.0.xs
d"
      local-part="Timestamp" />
      <tbs-element name-space=
"http://docs.oasis-open.org/wss/2004/01/oasis-200401-wss-wssecurity-secext-1.0.xsd
"
      local-part="UsernameToken" />
   </tbs-elements>
   <add-timestamp created="true" expiry="28800" />
</signature>
<encrypt>
   <recipient-key alias="orakey" />
   <encryption-method>AES-128</encryption-method>
   <keytransport-method>RSA-OAEP-MGF1P</keytransport-method>
   <tbe-elements>
      <tbe-element local-part="Body" name-space=
       "http://schemas.xmlsoap.org/soap/envelope/" mode="CONTENT" />
      <tbe-element name-space=
"http://docs.oasis-open.org/wss/2004/01/oasis-200401-wss-wssecurity-secext-1.0.xsd
"
      local-part="UsernameToken" />
   </tbe-elements>
```

```
    </encrypt>
```

Web Service (with sample data)

Define the <verify-username-token>, <verify-signature>, and <decrypt> elements in the service deployment descriptor. For example:

```
<verify-username-token password-type="PLAINTEXT" require-nonce="false"
 require-created="false" />
<verify-signature>
   <tbs-elements>
      <tbs-element name-space="http://schemas.xmlsoap.org/soap/envelope/"
       local-part="Body" />
      <tbs-element name-space=
"http://docs.oasis-open.org/wss/2004/01/oasis-200401-wss-wssecurity-utility-1.0.xs
d"
       local-part="Timestamp" />
   </tbs-elements>
   <verify-timestamp expiry="28800" created="true" />
</verify-signature>
<decrypt>
   <tbe-elements>
      <tbe-element name-space="http://schemas.xmlsoap.org/soap/envelope/"
       local-part="Body" mode="CONTENT" />
   </tbe-elements>
</decrypt>
```

13.2.5.2 Oracle WSM 11*g*

Perform the following steps:

1. Attach policies as follows:

    Client: oracle/wss10_username_token_with_message_protection_client_policy.

    Web service: oracle/wss10_username_token_with_message_protection_service_ policy.

    For more information about attaching policies in Oracle Fusion Middleware 11*g*, see "Attaching Policies to Web Services" in *Oracle Fusion Middleware Security and Administrator's Guide for Web Services*.

2. Leave the configuration set for message body signing and encryption.

3. Configure the Authentication and Identity Assertion provider.

13.2.6 ID Propagation Using SAML Token (Sender Vouches) with Message Protection (WS-Security 1.0)

The following sections describe how to implement ID propagation using SAML token sender vouches with message protection that conforms to the WS-Security 1.0 standard, and compare the steps required in the OC4J 10*g* and Oracle WSM 11*g* environments.

For more information about the deployment descriptor elements, see "OracleAS Web Services Security Schema" in *Oracle Application Server Web Services Security Guide 10g (10.1.3.1.0)* at:

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

13.2.6.1 OC4J 10*g*

Edit the deployment descriptors for the Web service and client, as described in the following sections.

Web Service Client (with sample data)

Define the <saml-token>, <signature>, and <encrypt> elements in the client deployment descriptor. For example:

```
<saml-token issuer-name="www.oracle.com" name="weblogic"
 name-format="UNSPECIFIED">
   <subject-confirmation-method>
      <confirmation-method>SENDER-VOUCHES</confirmation-method>
   </subject-confirmation-method>
</saml-token>
<signature>
   <signature-method>RSA-SHA1</signature-method>
   <tbs-elements>
      <tbs-element local-part="Body"
       name-space="http://schemas.xmlsoap.org/soap/envelope/" />
      <tbs-element name-space=
"http://docs.oasis-open.org/wss/2004/01/oasis-200401-wss-wssecurity-utility-1.0.xs
d"
       local-part="Timestamp" />
   </tbs-elements>
   <add-timestamp created="true" expiry="28800" />
</signature>
<encrypt>
   <recipient-key alias="orakey" />
   <encryption-method>AES-128</encryption-method>
   <keytransport-method>RSA-OAEP-MGF1P</keytransport-method>
   <tbe-elements>
      <tbe-element local-part="Body" name-space=
       "http://schemas.xmlsoap.org/soap/envelope/" mode="CONTENT" />
   </tbe-elements>
</encrypt>
```

Web Service (with sample data)

Define the <verify-saml-token>, <verify-signature>, and <decrypt> elements in the service deployment descriptor. For example:

```
<verify-saml-token>
   <subject-confirmation-methods>
      <confirmation-method>SENDER-VOUCHES</confirmation-method>
   </subject-confirmation-methods>
</verify-saml-token>
<verify-signature>
   <tbs-elements>
      <tbs-element name-space="http://schemas.xmlsoap.org/soap/envelope/"
       local-part="Body" />
      <tbs-element name-space=
"http://docs.oasis-open.org/wss/2004/01/oasis-200401-wss-wssecurity-utility-1.0.xs
d"
       local-part="Timestamp" />
   </tbs-elements>
   <verify-timestamp expiry="28800" created="true" />
</verify-signature>
<decrypt>
   <tbe-elements>
```

```
    <tbe-element name-space="http://schemas.xmlsoap.org/soap/envelope/"
     local-part="Body" mode="CONTENT" />
  </tbe-elements>
</decrypt>
```

13.2.6.2 Oracle WSM 11*g*

Attach policies as follows:

- **Client**: oracle/wss10_saml_token_with_message_protection_client_policy.

- **Web service**: oracle/wss10_saml_token_with_message_protection_service_policy.

For more information about attaching policies in Oracle Fusion Middleware 11*g*, see "Attaching Policies to Web Services" in *Oracle Fusion Middleware Security and Administrator's Guide for Web Services*.

13.2.7 ID Propagation Using SAML Token (Holder of Key) with Message Protection (WS-Security 1.0)

The following sections describe how to implement ID propagation using SAML token holder of key with message protection that conforms to the WS-Security 1.0 standard, and compare the steps required in the OC4J 10*g* and Oracle WSM 11*g* environments.

For more information about the deployment descriptor elements, see "OracleAS Web Services Security Schema" in *Oracle Application Server Web Services Security Guide* in the 10*g* Release 3 (10.1.3.1.0) documentation library at:

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

13.2.7.1 OC4J 10*g*

Edit the deployment descriptors for the Web service and client, as described in the following sections.

Web Service Client (with sample data)

Define the <saml-token>, <signature>, and <encrypt> elements in the client deployment descriptor. For example:

```
<saml-token issuer-name="www.oracle.com" name="weblogic"
 name-format="UNSPECIFIED">
   <subject-confirmation-method>
      <confirmation-method>HOLDER-OF-KEY</confirmation-method>
   </subject-confirmation-method>
</saml-token>
<signature>
   <signature-method>RSA-SHA1</signature-method>
   <tbs-elements>
      <tbs-element local-part="Body"
       name-space="http://schemas.xmlsoap.org/soap/envelope/" />
      <tbs-element name-space=
"http://docs.oasis-open.org/wss/2004/01/oasis-200401-wss-wssecurity-utility-1.0.xs
d"
       local-part="Timestamp" />
   </tbs-elements>
   <add-timestamp created="true" expiry="28800" />
</signature>
<encrypt>
   <recipient-key alias="orakey" />
   <encryption-method>AES-128</encryption-method>
```

```
      <keytransport-method>RSA-OAEP-MGF1P</keytransport-method>
      <tbe-elements>
         <tbe-element local-part="Body" name-space=
          "http://schemas.xmlsoap.org/soap/envelope/" mode="CONTENT" />
      </tbe-elements>
   </encrypt>
```

Web Service (with sample data)

Define the <verify-saml-token>, <verify-signature>, and <decrypt> elements in the service deployment descriptor. For example:

```
<verify-saml-token>
   <subject-confirmation-methods>
      <confirmation-method>HOLDER-OF-KEY</confirmation-method>
   </subject-confirmation-methods>
</verify-saml-token>
<verify-signature>
   <tbs-elements>
      <tbs-element name-space="http://schemas.xmlsoap.org/soap/envelope/"
       local-part="Body" />
      <tbs-element name-space=
"http://docs.oasis-open.org/wss/2004/01/oasis-200401-wss-wssecurity-utility-1.0.xs
d"
       local-part="Timestamp" />
   </tbs-elements>
   <verify-timestamp expiry="28800" created="true" />
</verify-signature>
<decrypt>
   <tbe-elements>
     <tbe-element name-space="http://schemas.xmlsoap.org/soap/envelope/"
      local-part="Body" mode="CONTENT" />
   </tbe-elements>
</decrypt>
```

13.2.7.2 Oracle WSM 11*g*

Attach policies as follows:

- **Client**: oracle/wss10_saml_hok_with_message_protection_client_policy.

- **Web service**: oracle/wss10_saml_hok_with_message_protection_service_policy.

For more information about attaching policies in Oracle Fusion Middleware 11*g*, see "Attaching Policies to Web Services" in *Oracle Fusion Middleware Security and Administrator's Guide for Web Services*.

13.2.8 Mutual Authentication with Message Protection (WS-Security 1.0)

The following sections describe how to implement mutual authentication with message protection that conforms to the WS-Security 1.0 standard, and compare the steps required in the OC4J 10*g* and Oracle WSM 11*g* environments.

For more information about the deployment descriptor elements, see "OracleAS Web Services Security Schema" in *Oracle Application Server Web Services Security Guide* in the 10*g* Release 3 (10.1.3.1.0) documentation library at:

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

13.2.8.1 OC4J 10*g*

Edit the deployment descriptors for the Web service and client, as described in the following sections.

Web Service Client (with sample data)

Define the <x509-token>, <signature>, and <encrypt> elements in the client deployment descriptor. For example:

```
<x509-token />
<signature>
   <signature-method>RSA-SHA1</signature-method>
   <tbs-elements>
      <tbs-element local-part="Body"
       name-space="http://schemas.xmlsoap.org/soap/envelope/" />
      <tbs-element name-space=
"http://docs.oasis-open.org/wss/2004/01/oasis-200401-wss-wssecurity-utility-1.0.xs
d"
       local-part="Timestamp" />
   </tbs-elements>
   <add-timestamp created="true" expiry="28800" />
</signature>
<encrypt>
   <recipient-key alias="orakey" />
   <encryption-method>AES-128</encryption-method>
   <keytransport-method>RSA-OAEP-MGF1P</keytransport-method>
   <tbe-elements>
      <tbe-element local-part="Body"
       name-space="http://schemas.xmlsoap.org/soap/envelope/" mode="CONTENT" />
   </tbe-elements>
</encrypt>
```

Web Service (with sample data)

Define the <verify-x509-token>, <verify-signature>, and <decrypt> elements in the service deployment descriptor. For example:

```
<verify-x509-token />
<verify-signature>
   <tbs-elements>
      <tbs-element name-space="http://schemas.xmlsoap.org/soap/envelope/"
       local-part="Body" />
      <tbs-element name-space=
"http://docs.oasis-open.org/wss/2004/01/oasis-200401-wss-wssecurity-utility-1.0.xs
d"
       local-part="Timestamp" />
   </tbs-elements>
   <verify-timestamp expiry="28800" created="true" />
</verify-signature>
<decrypt>
   <tbe-elements>
      <tbe-element name-space="http://schemas.xmlsoap.org/soap/envelope/"
       local-part="Body" mode="CONTENT" />
   </tbe-elements>
</decrypt>
```

13.2.8.2 Oracle WSM 11*g*

Perform the following steps:

1. Attach policies as follows:

    Client: oracle/wss10_x509_token_with_message_protection_client_policy.

    Web service: oracle/wss10_x509_token_with_message_protection_service_policy.

    For more information about attaching policies in Oracle Fusion Middleware 11*g*, see "Attaching Policies to Web Services" in *Oracle Fusion Middleware Security and Administrator's Guide for Web Services*.

2. Leave the configuration set for message body signing and encryption.

3. Configure the Authentication and Identity Assertion provider.

13.2.9 Username token over SSL

The following sections describe how to implement username token over SSL, and compare the steps required in the OC4J 10*g* and Oracle WSM 11*g* environments.

For more information about the deployment descriptor elements, see "OracleAS Web Services Security Schema" in *Oracle Application Server Web Services Security Guide* in the 10*g* Release 3 (10.1.3.1.0) documentation library at:

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

13.2.9.1 OC4J 10*g*

Configure the application server for SSL and edit the deployment descriptors for the Web service and client, as described in the following sections.

Web Service Client (with sample data)

Define the <username-token> and <signature> elements in the client deployment descriptor. For example:

```
<username-token password-type="PLAINTEXT" add-nonce="true"
 add-created="true" />
<signature>
   <add-timestamp created="true" expiry="28800" />
</signature>
```

Web Service (with sample data)

Define the <verify-username> element in the service deployment descriptor. For example:

```
<verify-username-token password-type="PLAINTEXT" require-nonce="false"
require-created="false" />
<signature>
   <verify-timestamp expiry="28800" created="true" />
</signature>
```

13.2.9.2 Oracle WSM 11*g*

Perform the following step:

1. Configure the application server for SSL.

2. Attach policies as follows:

    Client: oracle/wss_username_token_over_ssl_client_policy.

    Web service: oracle/wss_username_token_over_ssl_service_policy

For more information about attaching policies in Oracle Fusion Middleware 11*g*, see "Attaching Policies to Web Services" in *Oracle Fusion Middleware Security and Administrator's Guide for Web Services*.

13.2.10 ID Propagation with SAML Token (Sender Vouches) over SSL (WS-Security 1.0)

The following sections describe how to implement ID propagation with SAML token sender vouches over SSL that conforms to WS-Security 1.0, and compare the steps required in the OC4J 10*g* and Oracle WSM 11*g* environments.

For more information about the deployment descriptor elements, see "OracleAS Web Services Security Schema" in *Oracle Application Server Web Services Security Guide* in the 10*g* Release 3 (10.1.3.1.0) documentation library at:

http://www.oracle.com/technology/documentation/

13.2.10.1 OC4J 10*g*

Configure the application server for SSL and edit the deployment descriptors for the Web service and client, as described in the following sections.

Web Service Client (with sample data)

Define the <saml-token> and <signature> elements in the client deployment descriptor. For example:

```
<saml-token name="weblogic" issuer-name="www.oracle.com"
 name-format="UNSPECIFIED">
   <subject-confirmation-method>
     <confirmation-method>SENDER-VOUCHES</confirmation-method>
   </subject-confirmation-method>
</saml-token>
<signature>
   <add-timestamp created="true" expiry="28800" />
</signature>
```

Web Service (with sample data)

Define the <verify-saml-token> element in the service deployment descriptor. For example:

```
<verify-saml-token>
   <subject-confirmation-methods>
     <confirmation-method>SENDER-VOUCHES-UNSIGNED</confirmation-method>
   </subject-confirmation-methods>
</verify-saml-token>
<signature>
   <verify-timestamp expiry="28800" created="true" />
</signature>
```

13.2.10.2 Oracle WSM 11*g*

Perform the following steps:

1. Configure the application server for SSL.

2. Attach policies as follows:

   Client: oracle/wss_saml_token_over_ssl_client_policy.

   Web service: oracle/wss_saml_token_over_ssl_service_policy.

13.2.11 Log Information

The following sections describe how to enable the collection of log information, and compare the steps required in the OC4J 10*g* and Oracle WSM 11*g* environments.

13.2.11.1 OC4J 10*g*

Configure the logging and auditing interceptor.

13.2.11.2 Oracle WSM 11*g*

Attach the following policy to the Web service or client: oracle/log_policy.

For more information about attaching policies in Oracle Fusion Middleware 11*g*, see "Attaching Policies to Web Services" in *Oracle Fusion Middleware Security and Administrator's Guide for Web Services*.

14

Upgrading Other Oracle SOA Suite Technologies

This chapter provides important supplementary information upgrading Oracle SOA applications to Oracle Fusion Middleware 11*g*.

Use Chapter 8, "Overview of Upgrading Oracle SOA Suite, WebCenter, and ADF Applications" for the tasks required to upgrade any Oracle SOA Suite, WebCenter, and ADF application.

Use the following sections to understand tasks specific to upgrading Oracle SOA applications:

- Section 14.1, "Upgrade Tasks Associated with Oracle Enterprise Service Bus"

- Section 14.2, "Upgrade Tasks Associated with Technology Adapters"

- Section 14.3, "Upgrade Tasks Associated with Human Workflow Tasks"

- Section 14.4, "Upgrade Tasks Associated with Oracle Business Rules"

- Section 14.5, "Upgrading Oracle SOA Suite Client Applications"

14.1 Upgrade Tasks Associated with Oracle Enterprise Service Bus

Refer to the following sections for information about upgrading Oracle Enterprise Service Bus 10*g* to Oracle Mediator 11*g*:

- Upgrading Oracle Enterprise Service Bus Routing Rules That Are Not Exposed as Services

- Upgrading Oracle Enterprise Service Bus Domain Value Maps (DVMs) and Cross References

- Upgrading Oracle Enterprise Service Bus Projects with SOAP Headers to Oracle Mediator 11g

- Upgrading Projects with Multiple Routing Services That Use the Same Namespace

- Upgrading Routing Services to Use Asynchronous Routing

- Upgrading Filtering or Setting Message Headers in Oracle Enterprise Service Bus

- Using the Sync From ESB Server Command Before Upgrading ESB Projects

14.1.1 Upgrading Oracle Enterprise Service Bus Routing Rules That Are Not Exposed as Services

If you upgrade an Oracle Enterprise Service Bus 10*g* project that contains an ESB routing service that is not exposed as a Web service, then the routing service and related composites cannot be invoked after upgrade.

To resolve this issue, modify the routing service in Oracle Mediator 11*g* so it is exposed as a Web service. For more information, see "Creating Mediator Routing Rules" in the *Oracle Fusion Middleware Developer's Guide for Oracle SOA Suite*.

Alternatively, you can ensure that all routing services in your Oracle Enterprise Service Bus 10*g* projects are exposed as Web services before you begin the upgrade process.

14.1.2 Upgrading Oracle Enterprise Service Bus Domain Value Maps (DVMs) and Cross References

If you use domain value maps in your and Oracle Enterprise Service Bus 10g projects, then note that while the xPath functions for DVMs and cross-references are upgraded automatically, you must perform a post-upgrade task to upgrade the DVM and cross reference metadata to 11*g*.

For more information, see Section 9.4.8.7, "Upgrading Domain Value Maps (DVMs) and Cross References (XREFs) in the ESB Repository".

14.1.3 Upgrading Oracle Enterprise Service Bus Projects with SOAP Headers to Oracle Mediator 11*g*

If you used XSLT functions to manipulate SOAP headers in Enterprise Service Bus (ESB) 10*g*, then after you upgrade your application, those header manipulation will be modeled as assignments in Oracle Mediator 11*g*.

For complex header manipulations, you should manually verify the SOAP headers before deploying the upgraded Oracle Mediator 11*g* projects.

For more information, see "Creating Static Routing Rules" in the *Oracle Fusion Middleware Developer's Guide for Oracle SOA Suite*.

14.1.4 Upgrading Projects with Multiple Routing Services That Use the Same Namespace

Oracle Enterprise Service Bus 10*g* supports multiple routing services that have WSDLs that reference the same `targetNameSpace` within a single ESB project.

For example, at least one of the sample 10*g* applications available on the Oracle Technology Network (the `samples_102_responseForward` sample application) supported this scenario.

However, Oracle Mediator 11*g* no longer supports this configuration. During the upgrade of the application in Oracle JDeveloper 11*g*, whenever a second service is found to have the WSDL pointing to the same `targetNameSpace`, the namespace is modified by suffixing *_service-qname* to the namespace. In addition, a warning message is added to the log file.

For example, in the `Samples_102_ResponseForward` sample application, the `targetNameSpace` of the service DispatchShipmentDetails is:

```
@ http://oracle.com/esb/namespaces/Samples_102_ResponseForward
```

After the upgrade of the application, it appears as follows:

```
@ http://oracle.com/esb/namespaces/
  Samples_102_ResponseForward_Samples.102_ResponseForward.DispatchShipmentDetails
```

In this example, *Samples.102_ResponseForward.DispatchShipmentDetails* is the qname of service `DispatchShipmentDetails`.

14.1.5 Upgrading Routing Services to Use Asynchronous Routing

In Oracle Enterprise Service Bus 10*g*, there was no support for asynchronous routing services. Instead, when Oracle Enterprise Service Bus called an Oracle BPEL Process Manager asynchronous process, then a OneWay service was created.

If you created a project that uses such a OneWay service, then after you open and upgrade the application in Oracle JDeveloper 11*g*, you can remove the old routing service and recreate the routing service to use an Oracle Mediator 11*g* asynchronous routing instead of the OneWay invocation.

14.1.6 Upgrading Filtering or Setting Message Headers in Oracle Enterprise Service Bus

Oracle Enterprise Service Bus 10*g* supports filtering or setting message headers for standard and custom header properties for adapters, such as SOAP, JCA, JMS, and AQ adapters. Filtering in Oracle Enterprise Service Bus is done as part of the `<filterExpression>` tag of the routing rule.

Many filter expressions are automatically upgraded, but for complex filters, the upgrade might be partial.

When you open your Oracle Enterprise Service Bus projects in Oracle JDeveloper 11*g*, each routing rule is upgraded into case, and the filter expression is converted to condition. These changes can be found in the mplan file of the Routing Service.

For example, the following code snippet shows the filter expression in an Oracle Enterprise Service Bus 10*g* routing rule:

```
<filterExpression>{ehdr:getRequestHeader
    ('/fhdr:InboundFileHeaderType/fhdr:fileName') = 'Mobile.xml'};{ namespace
        fhdr=http://xmlns.oracle.com/pcbpel/adapter/file/ namespace
        ehdr=http://www.oracle.com/XSL/Transform/java
                        /oracle.tip.esb.server.headers.ESBHeaderFunctions }
</filterExpression>
```

During upgrade, this example is converted to the equivalent condition of the case in the mplan file:

```
<condition language="xpath" xmlns:ehdr="http://www.oracle.com/XSL
              /Transform/java/oracle.tip.esb.server.headers.ESBHeaderFunctions"
          xmlns:fhdr="http://xmlns.oracle.com/pcbpel/adapter/file/">
    <expression>$in.property.jca.file.FileName ='Mobile.xml'</expression>
</condition>
```

Setting message headers is accomplished as part of the transformation files and these are upgraded to assign statements.

The following code snippet shows an example of setting message headers in Oracle Enterprise Service Bus 10*g*:

```
<xsl:variable name="inputCountry" select="/imp1:CustomerData/Country"/>
```

```
<select="ehdr:setOutboundHeader
    ('/jhdr:JMSOutboundHeadersAndProperties
    /jhdr:JMSOutboundProperties/jhdr:Property[position()=1]/@value',$inputCountry,
    'jhdr=http://xmlns.oracle.com/pcbpel/adapter/jms/;')"/>
```

During upgrade, this example is converted to the following equivalent assign element
t in 11*g*:

```
<assign>
    <copy expression="$in.CustomerData/CustomerData/Country"
          target="$out.CustomerData/jhdr:JMSOutboundHeadersAndProperties
          /jhdr:JMSOutboundProperties/jhdr:Property[position()=1]/@value"/>
</assign>
```

Most of the standard header properties are upgraded to their equivalent 11*g* headers.
Customized headers are upgraded using a best effort approach, but it is not possible to
upgrade all the variations of customized headers.

As a result, you must manually upgrade any customized headers the equivalent
condition or assign element and equivalent mplan artifacts.

For more information, see "Getting Started with Oracle Mediator" in the *Oracle Fusion
Middleware Developer's Guide for Oracle SOA Suite*.

14.1.7 Using the Sync From ESB Server Command Before Upgrading ESB Projects

To avoid issues when upgrading your Oracle Enterprise Service Bus 10*g* projects to
Oracle SOA Suite 11*g*, make sure that you have synchronized the services on the ESB
server with the services currently in your ESB Oracle JDeveloper project.

To perform this task:

1. Click the **Sync from ESB Server** icon at the top of the Design tab in Oracle
   JDeveloper 10*g*.

2. Click **Yes** in the Confirm Sync from ESB Server dialog.

For more information, see "Syncing Services From ESB Server" in the *Oracle Enterprise
Service Bus Developer's Guide*, which is available in the Oracle Application Server 10*g*
Release 3 (10.1.3.1.0) on the Oracle Technology Network (OTN):

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

14.2 Upgrade Tasks Associated with Technology Adapters

The following sections describe upgrade tasks you might need to perform when
upgrading Technology Adapters to Oracle Fusion Middleware 11*g*:

- Verifying Adapters with the Oracle JDeveloper 11g Adapter Configuration Wizard

- Upgrading Adapter Header Variables

- Upgrading Non-Managed Connection Information in the Adapter JCA File

- Description Lost After Upgrade of Technology Adapter

- Upgrading an Inbound Synchronous Request-Reply Use Case

- Upgrading Projects That Use Outbound Dequeue

- Upgrading AQ Adapter Headers

- Additional Steps When Upgrading Technology Adapter Headers

- Upgrading an AQ Outbound Request-Reply Scenario
- Upgrading Database Adapters Used for Stored Procedures

14.2.1 Verifying Adapters with the Oracle JDeveloper 11*g* Adapter Configuration Wizard

After upgrading your Oracle SOA Suite 10*g* projects to Oracle SOA Suite 11*g*, Oracle recommends that you use the adapter configuration wizard to verify the upgrade of your technology adapters. This will ensure that all required fields in your 11*g* are populated and that the adapter connections to your new Oracle Fusion Middleware 11*g* environment are valid.

For more information, see the information the specific adapters you are using in the *Oracle Fusion Middleware User's Guide for Technology Adapters*.

14.2.2 Upgrading Adapter Header Variables

After you upgrade an application that uses adapter headers, you will likely receive an error after you open the application in Oracle JDeveloper 11*g*. The error will indicate that a variable or parameter was not recognized while it was parsing a header function.

This is caused by a change in the way adapter header variables are defined in Oracle Fusion Middleware 11*g*.

In Oracle Application Server 10*g* Release 3 (10.1.3), you create adapter header variables in a `wsdl` file that has a name such as the following, or a similar name for the direction and adapter type you are using:

```
fileAdapterOutboundHeader.wsdl
```

In Oracle Fusion Middleware 11*g*, adapter header variables are now properties that you set on the **Properties** tab of the Invoke dialog box in Oracle JDeveloper.

For example, the properties you can set for the JCA Adapter for Files/FTP are described in the section, "Inbound Headers" in the *Oracle Fusion Middleware User's Guide for Technology Adapters*.

After you open your project in Oracle JDeveloper 11*g*, the wsdl header files you created for Oracle Application Server 10*g* will no longer be attached to the project. You must define each of the variables in the wsdl files as properties for each adapter type that uses header variables.

For more information, refer to the appropriate chapter for each adapter type in the *Oracle Fusion Middleware User's Guide for Technology Adapters*.

14.2.3 Upgrading Non-Managed Connection Information in the Adapter JCA File

In Oracle Application Server 10*g*, you can use non-managed connections in the JCA file to point to your development resources. For example, data source definitions can be specified using the `non-managed-connection` element in the JCA file.

In Oracle Application Server 10*g*, using non-managed connections was reserved for development only. In Oracle Fusion Middleware 11*g*, non-managed connections are not supported.

Ideally, before upgrading your Oracle Application Server 10*g* projects to Oracle Fusion Middleware 11*g*, you should already be using JCA files that include only JNDI

references to connections that are defined properly in the datasources.xml or oc4j-ra.xml files.

If you attempt to upgrade applications that use non-managed connections for adapter connections, you must modify the projects after upgrade to use proper connections defined by using supported administration tools, such as the Oracle WebLogic Server Administration Console.

14.2.4 Description Lost After Upgrade of Technology Adapter

If you created an 10*g* Release 3 (10.1.3) technology adapter using the Adapter Configuration Wizard, and you entered a description for the adapter in the **Description** field of the wizard, the description will be lost when you open and migrate the project in Oracle JDeveloper 11*g*.

This is because the `documentation` element in removed from the WSDL file during the upgrade.

If the adapter is important, you can use the Oracle JDeveloper 11*g* Adapter Configuration Wizard to add the description after you have upgraded the project.

14.2.5 Upgrading an Inbound Synchronous Request-Reply Use Case

In Oracle SOA Suite 10*g*, if you used the MQSeries Adapter inbound synchronous request-reply use case, you will notice that there is only one message type, which is REQUEST.

In Oracle SOA Suite 11*g*, the MQAdapter supports two message types: REQUEST and NORMAL. The REQEUST MQ message type always has `replyToQueue` set on its header so there is no need for `FallbackQueueName` and `FallbackQueueManagerName`. On the other hand, a NORMAL Message may or may not have `FallbackQueueName` and `FallbackQueueManagerName`.

So, when a 10*g* inbound synchronous request-reply scenario is upgraded to 11*g*, then the Message Type remains as REQUEST type. Therefore, at runtime, there is no need for `FallbackQueueName` and `FallbackQueueManagerName`, even though these properties would resides in the `.jca` file. If you want to use these properties, you must set the message type to NORMAL.

For more information about the inbound synchronous request-reply use case, see "Oracle MQ Series Adapter Use Cases" in the *Oracle Fusion Middleware User's Guide for Technology Adapters*.

14.2.6 Upgrading Projects That Use Outbound Dequeue

If you used the MQSeries Adapter outbound dequeue use case, which was a documented use case scenario in 10*g* Release 3 (10.1.3), then note that these types of projects cannot be upgraded by Oracle JDeveloper 11*g*.

Instead, you must recreate these types of projects after the upgrade to Oracle JDeveloper 11*g*. While recreating the project you must ensure that you set the value for `UITransmissionPrimitive` to `SynchronousDequeue`.

For more information, see "Outbound Dequeue Scenario" in the *Oracle Fusion Middleware User's Guide for Technology Adapters*.

14.2.7 Upgrading AQ Adapter Headers

If you are upgrading an application that uses the Oracle Application Server Adapter for Advanced Queuing (AQ adapter), then note that the AQ Adapter header is accessed differently in Oracle Fusion Middleware 11*g*.

Specifically, you must perform the following post-upgrade steps for AQ adapter headers after you upgrade to Oracle SOA Suite 11*g*. This procedure is required for both Oracle BPEL Process Manager and for Oracle Enterprise Service Bus projects that are upgraded to Oracle Mediator.

- After upgrading to Oracle Mediator, assign the "jca.aq.HeaderDocument" value from inbound to outbound in the `.mplan` file of the Oracle Mediator file.

- After upgrading to Oracle BPEL Process Manager 11*g*:

  1. Define a new variable 'vInboundHeaders' of type string.

     For example:

     ```
     <variable name="vInboundHeaders" type="xsd:string"/>
     ```

  2. In the receive activity, make sure that the `jca.aq.HeaderDocument` property gets written to the above variable.

     For example:

     ```
     <bpelx:property name="jca.aq.HeaderDocument"
          variable="vInboundHeaders"/>
     ```

  3. At the time of invocation, make sure that the "vInboundHeaders" value is copied back to the `jca.aq.HeaderDocument` property:

     ```
     <invoke name="Invoke_1"
             inputVariable="Invoke_1_Enqueue_InputVariable"
             partnerLink="EnqueueClobPayload" portType="ns2:Enqueue_ptt"
             operation="Enqueue">
                 <bpelx:inputProperty name="jca.aq.HeaderDocument"
                                      variable="vInboundHeaders"/>
     </invoke>
     ```

14.2.8 Additional Steps When Upgrading Technology Adapter Headers

Use the following procedure when you are not using an AQ adapter, and as a result you are not obtaining the value of the `jca.aq.HeaderDocument` property. In this scenario, you must populate the property manually.

This involves two steps. Perform one step after you upgrade Oracle Enterprise Service Bus to Oracle Mediator 11*g*, and perform the second step after you upgrade to Oracle BPEL Process Manager 11*g*.

Note that these examples show how you might invoke Oracle B2B from the adapter header.

Example Steps to Perform After Upgrading to Oracle Mediator

After upgrading to Oracle Mediator, assign the `jca.aq.HeaderDocument` value from inbound to outbound in the `.mplan` file of the Oracle Mediator file.

For example:

```
<copy target="$out.property.jca.aq.HeaderDocument" expression="'&lt;?xml version =
  &quot;1.0&quot; standalone= &quot;yes&quot;?>&lt;tns:Header
  @ xmlns:tns=&quot;http://xmlns.oracle.com/pcbpel/adapter/aq/outbound/">
```

```
<tns:PayloadHeader>&lt;MSG_ID>&lt;
   /MSG_ID>&lt;INREPLYTO_MSG_ID>&lt;
   /INREPLYTO_MSG_ID>&lt;FROM_PARTY>GlobalChips&lt;
   /FROM_PARTY>&lt;TOPARTY>Acme&lt;
   /TO_PARTY>&lt;ACTION_NAME>&lt;/ACTION_NAME>&lt;DOCTYPE_NAME>ORDERS&lt;
   /DOCTYPE_NAME>&lt;DOCTYPE_REVISION>D98A&lt;
   /DOCTYPE_REVISION>&lt;MSG_TYPE>1&lt;
   /MSG_TYPE>&lt;PAYLOAD>&lt;
   /PAYLOAD>&lt;/tns:PayloadHeader>&lt;/tns:Header>'"/>
```

Example Steps to Perform After Upgrading to Oracle BPEL Process Manager

After upgrading to Oracle BPEL Process Manager 11*g*:

1. Define a new variable "headerVariable" of type `string`.

   For example:

   ```
   <variable name="headerVariable" type="xsd:string"/>
   ```

2. Create an assign block, such as the one shown in Example 14–1.

3. At the time of invocation, make sure that the "headerVariable" value is copied back to the `jca.aq.HeaderDocument` property:

   ```
   <invoke name="Send_PO" partnerLink="X12_4010_850_ENQ_Service"
       portType="ns2:Enqueue_ptt"
       operation="Enqueue"
       inputVariable="Send_PO_Enqueue_InputVariable">
       <bpelx:inputProperty name="jca.aq.HeaderDocument"
                            variable="headerVariable"/>
   </invoke>
   ```

Example 14–1   Assign Block When Upgrading Adapter Headings

```
<assign name="assign header ">
  <copy>
      <from expression="'&lt;?xml version = &quot;1.0&quot;standalone = &quot;yes&quot;
  standalone = &quot;yes&quot;?>&lt;tns:Header
  @ xmlns:tns=&quot;http://xmlns.oracle.com/pcbpel/adapter/aq/Inbound/">
  <tns:PayloadHeader>&lt;MSG_ID>12345&lt;
     /MSG_ID>&lt;INREPLYTO_MSG_ID>54321&lt;
     /INREPLYTO_MSG_ID>&lt;FROM_PARTY>Acme&lt;
     /FROM_PARTY>&lt;TO_PARTY>GlobalChips&lt;
     /TO_PARTY>&lt;ACTION_NAME>contentType:application/octet-stream;filename:abc.xml&lt;
     /ACTION_NAME>&lt;DOCTYPE_NAME>850&lt;
     /DOCTYPE_NAME>&lt;DOCTYPE_REVISION>4010&lt;
     /DOCTYPE_REVISION>&lt;MSG_TYPE>1&lt;/MSG_TYPE>&lt;PAYLOAD>&lt;
     /PAYLOAD>&lt;/tns:PayloadHeader>&lt;/tns:Header>'"/>
     <to variable="headerVariable"/>
  </copy>
</assign>
```

14.2.9 Upgrading Composite Applications That Rely on Payload Header Functionality

Section 14.2.7, "Upgrading AQ Adapter Headers" describes how to use the `jca.aq.HeaderDocument` property that is required in Oracle Fusion Middleware 11*g* when your application needs to access Payload Headers. The procedure described in that section must be performed for all Oracle Application Server 10*g* BPEL and ESB processes that access the payload headers functionality.

Note that in addition to the information in Section 14.2.7, the target namespace of `<Header>` and `<PayloadHeader>` elements have changed Oracle Fusion Middleware11*g* Release 1 (11.1.1.3.0). As a result, for any BPEL and ESB processes that involve accessing payload headers and that were designed using Oracle SOA Suite 10*g*, you must use the adapter wizard to regenerate its service and reference artifacts, also known as partner links in Oracle Application Server 10*g*.

You must regenerate the artifacts so the correct target namespace is used for the definition of the `<Header>` element. This step is required in addition to the steps covered in Section 14.2.7 to access payload headers.

In addition, you must change any Assign and Transformations that use the `<Header>` element. If you do not perform this step, then a namespace mismatch between design time and runtime results in XPath failures, as well as message enqueue failures.

14.2.10 Upgrading an AQ Outbound Request-Reply Scenario

In Oracle Fusion Middleware 11*g*, the procedure you use to create AQ outbound request-reply scenarios for Oracle BPEL Process Manager projects has changed.

As a result, when you upgrade an Oracle SOA Suite 10*g* application that uses this scenario, the WSDL file that is created during the upgrade does not contain the required binding information.

To correct this problem, do the following after you open and upgrade the application in Oracle JDeveloper 11*g*:

1. Add the JCA entry to the WSDL file created by the upgrade, as follows:

```
<?binding.jca CombinedRequestReply_aq.jca?>
<wsdl:definitions name="CombinedRequestReply" ... >
  .
  .
  .
</wsdl:definitions>
```

2. Add the binding entry in the `composite.xml` file:

```
<reference ui:wsdlLocation="CombinedRequestReply.wsdl"
          name="CombineRequestReply">
    <interface.wsdl interface="http://xmlns.oracle.com/pcbpel
                /adapter/aq/EnqueueRequest/#wsdl.interface(Enqueue_ptt)"
            callbackInterface="http://xmlns.oracle.com/pcbpel
                /adapter/aq/DequeueReply/#wsdl.interface(Dequeue_ptt)"
            xmlns:ns="http://xmlns.oracle.com/sca/1.0"/>
      <binding.jca config="CombinedRequestReply_aq.jca"/>
</reference>
```

3. Remove any unused WSDL references from the `.bpel` file.

14.2.11 Upgrading Database Adapters Used for Stored Procedures

As described in Section 9.2, "About Using the Oracle JDeveloper 11g Migration Wizard for Oracle SOA Suite Applications", Oracle recommends that you apply the latest patch sets to your Oracle SOA Suite environment and use the latest 10*g* Release 3 (10.1.3) Oracle JDeveloper before upgrading to 11*g*.

However, if you are upgrading from 10*g* Release 3 (10.1.3.3) or earlier, note the following if you are using a database adapter for stored procedures.

- In Oracle SOA Suite 10*g* Release 3 (10.1.3.3) and earlier, the XSD generator of the adapter generated an XSD that contained unqualified elements. You can identify unqualified elements by viewing the `InputParameters` root element in the XSD file. The unqualified element appears as follows:

  ```
  <db:InputParameters ... ></db:InputParameters>
  ```

  The `db:` namespace is specified in the definition of the schema, which is the same as the target namespace.

- In 10*g* Release 3 (10.1.3.4) and later, the XSD generator generates qualified elements, which appear as follows in the `InputParameters` root element:

  ```
  <InputParameters ..></InputParameters>
  ```

  Note that the `db:` namespace has been removed.

- If you were using a version of Oracle JDeveloper 10*g* Release 3 (10.1.3) that was incompatible with the Oracle SOA Suite version, you could also encounter a scenario where the elements are removed:

  ```
  <null>(some value)</null>
  ```

The solution is as follows:

- First, make sure that you are using a version of Oracle JDeveloper that matches the version of your Oracle SOA Suite environment.

- Second, after you upgrade to Oracle JDeveloper and Oracle SOA Suite10*g* Release 3 (10.1.3.4) or to Oracle SOA Suite and Oracle JDeveloper 11*g*, regenerate the XSD.

After you upgrade and regenerate the XSD, you must then modify any related assign activities and any xPath expressions accordingly.

14.3 Upgrade Tasks Associated with Human Workflow Tasks

The following upgrade tasks should be considered when you are upgrading Oracle SOA Suite applications that use human workflow tasks:

- Upgrading Task Details for an Associated Human Workflow Task

- Upgrading Multiple Projects That Share a Common Human Workflow Task

- Editing Human Workflow Tasks After Upgrade

14.3.1 Upgrading Task Details for an Associated Human Workflow Task

When you are upgrading Oracle SOA Suite 10*g* applications that include task details for a human task, note the following:

- In Oracle SOA Suite 10*g* Release 3 (10.1.3), task details were generated inside same composite project and was deployed during composite deployment.

- In Oracle SOA Suite 11*g*, task details are generated as a separate project that must be deployed separately.

As a result, during the upgrade of Oracle Human Workflow, Oracle JDeveloper 11*g* automatically creates a new project for the task details UI and upgrades the task detail pages. You must then deploy this project separately, after deploying the associated composite application.

14.3.2 Upgrading Multiple Projects That Share a Common Human Workflow Task

If you are upgrading an application that contains multiple Oracle JDeveloper projects that share a common human workflow task, then be sure to review the upgraded application carefully in Oracle JDeveloper 11*g*.

In some cases, because the projects are moved into a common directory during the upgrade, data associated with the workflow task in one project may overwrite data from another project if the workflow task in the second project uses the same name.

14.3.3 Editing Human Workflow Tasks After Upgrade

If you use Oracle JDeveloper 11*g* to upgrade an application with human workflow tasks, and then use the Task Editor to edit the task file, Oracle JDeveloper automatically generates a new payload XSD file.

However, the existing WorkflowTask XSD file contains references to the payload XSD file used in the 10*g* version of the application.

To fix this problem, edit the WorkflowTask XSD file to be sure it references the new payload XSD file.

For example:

- Change the following entry from:

  ```
  <include schemaLocation="***_payload.xsd" xmlns=""/>
  ```

  To:

  ```
  <include schemaLocation="***Payload.xsd" xmlns=""/>
  ```

- Change the following entry from:

  ```
  <element name="payload" type="tns:payloadType" minOccurs="0"
  maxOccurs="1"/>
  ```

  To:

  ```
  <element name="payload" type="tns:***PayloadType" minOccurs="0"
  maxOccurs="1"/>
  ```

In these examples, replace the asterisk with the appropriate name and type defined in the payload XSD file.

14.4 Upgrade Tasks Associated with Oracle Business Rules

When you upgrade projects that use Oracle Business Rules, refer to the following documentation resources:

- Section 9.4.1.1, "Upgrading to the Oracle Fusion Middleware Java API for Oracle Business Rules"
- Appendix B, "Upgrading Oracle Business Rules Dictionaries and Projects"

14.5 Upgrading Oracle SOA Suite Client Applications

While you are in the process of upgrading your Oracle SOA Suite applications to 11*g*, and while your Oracle SOA Suite 10*g* environment is still up and running, you should begin to review, upgrade, and test the client applications that depend upon your Oracle SOA Suite environment.

Use the following list to analyze your client applications for required updates that will allow them to work with your newly upgraded 11*g* Oracle SOA Suite environment:

- Review the client applications to understand which remote Oracle SOA Suite APIs they are using.

  For more information, refer to Section 9.4.1, "Understanding Oracle SOA Suite API Changes for Oracle Fusion Middleware 11g".

- Search your client applications for any references to Oracle SOA Suite HTTP URLs. The syntax of the Oracle SOA Suite 11*g* HTTP URLs has changed from 10*g*.

  To obtain the new URL that a client can use to access a deployed Oracle SOA Suite 11*g* application, use the home page of the application deployment in Oracle Enterprise Manager Fusion Middleware Control. For more information, see "Monitoring SOA Composite Applications" in the *Oracle Fusion Middleware Administrator's Guide for Oracle SOA Suite and Oracle Business Process Management Suite*.

- Review your client applications for references to message targets, such as JMS queues and data source. Any references to these resources must be updated to point to the upgraded, Oracle SOA Suite 11*g* environment.

- If a client application is using Oracle Service Bus (OSB) to invoke an Oracle BPEL Process Manager process with the optimized transport, you must modify the client application so it uses SOAP/HTML to invoke the Oracle BPEL Process Manager process.

  For more information, see the Oracle Service Bus 10*g* Release 3 (10.3) documentation, which is available on the Oracle Technology Network (OTN):

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

15

Considerations When Upgrading Oracle WebCenter Applications

This chapter describes how to upgrade Oracle WebCenter 10.1.3.x applications to Oracle WebCenter Portal 11*g*.

For information about the high-level tasks required to upgrade any Oracle SOA, Oracle WebCenter, and ADF application, see Chapter 8, "Overview of Upgrading Oracle SOA Suite, WebCenter, and ADF Applications."

Use the following sections to understand the tasks specific to upgrading Oracle WebCenter applications:

- About Upgrading WebCenter Applications
- Prerequisites for Upgrading WebCenter Applications
- Upgrading WebCenter Consumer Applications
- Upgrading Portlet Producers
- Migrating Data

> **Note:** This chapter uses recently rebranded Oracle WebCenter Portal product names where appropriate. For more information, see Section 3.3.1, "New Product Names for Oracle WebCenter and Oracle Enterprise Content Management Suite."

15.1 About Upgrading WebCenter Applications

This section lists the high-level upgrade tasks and provides an overview of the different WebCenter application templates.

15.1.1 Tasks Involved in Upgrading WebCenter Applications

Based on the functionality of your WebCenter 10.1.3.x applications, and the WebCenter services that they are configured to use, you may need to perform the following tasks to upgrade your applications to Oracle WebCenter Portal 11*g*:

- Upgrade WebCenter consumer applications by using Oracle JDeveloper
- Upgrade WebCenter portlet producer applications and their customizations, if WebCenter applications contain portlets.
- Migrate the data related to WebCenter services, if your WebCenter consumer applications use the specified services.

> **Note:** In this chapter, a *WebCenter consumer application* refers to an application that provides the user interface and contains JSP pages and components required by the application. A *portlet producer application* refers to an application that contains Oracle WebCenter server-side producer components and the portlets that can be consumed by a WebCenter consumer application.

15.1.2 About WebCenter Application Templates

Oracle WebCenter 10.1.3.x applications and Oracle WebCenter Portal 11*g* applications are based on different application templates. An understanding of various templates is important before you attempt to upgrade your WebCenter applications.

In JDeveloper 10.1.3.x, WebCenter applications are created using the **WebCenter Application [Portlet, Content Repository, JSF]** application template. Both WebCenter consumer applications and portlet producer applications are based on the same template.

By contrast, Oracle WebCenter Portal 11*g* applications are based on the **WebCenter Portal – Framework Application** template for consumer applications, or the **WebCenter Portal – Portlet Producer Application** template for portlet producer applications.

In JDeveloper 10.1.3.x, WebCenter applications consist of three projects: **Model**, **Portlets**, and **ViewController**. In JDeveloper 11*g*, applications created using the **WebCenter Portal – Framework Application** template contain the **Portal** project (which includes features like site navigation and page hierarchies) and the **PortalWebAssets** project (which includes static application resources like HTML and image files). Applications created using the **WebCenter Portal – Portlet Producer Application** template contain only one project, **Portlets**, scoped for creation of JSR 286 (standards-based) and Oracle PDK-Java portlets.

15.2 Prerequisites for Upgrading WebCenter Applications

Before you attempt to upgrade your WebCenter 10.1.3.x applications, perform the following tasks:

1. Verify that the application is running successfully on Oracle Application Server 10*g*.

    For information, see Section 8.2, "Task 2: Verify that the Applications Are Up and Running Successfully on Oracle Application Server 10g."

2. Verify that your Oracle WebCenter 10.1.3.x environment is upgraded to the Oracle WebCenter Portal 11*g* environment.

    For information, see Section 8.4, "Task 4: Verify That You Have Upgraded Your 10g Environment to 11g."

3. Install Oracle JDeveloper 11*g*.

    For information, see Section 8.5, "Task 5: Install and Start Oracle JDeveloper 11g."

4. Install the WebCenter Portal extension bundle.

    To work with WebCenter applications, you must install the WebCenter Portal extension, which is a JDeveloper add-in that provides all WebCenter Portal capabilities in JDeveloper.

    To install Oracle WebCenter Portal 11*g* extension:

a. Start JDeveloper 11*g*.

> **Note:** If you intend to upgrade application settings from JDeveloper 10.1.3.x, be sure to close and remove your WebCenter applications from the IDE in JDeveloper 10.1.3.x before you start JDeveloper 11*g*. This gives you more control in deciding when your applications will get migrated, and will not invoke application migration during JDeveloper startup.
>
> You can remove applications from the IDE by right-clicking an application name in the Application Navigator and selecting **Delete** from the shortcut menu. Removing an application removes it only from the IDE and not from your local storage system.

b. If the Select Role dialog displays, select **Default Role** to enable all technologies, and click **OK**.

c. If this is the first time you started JDeveloper 11*g*, it prompts you to import preferences from a previous version of JDeveloper.

In the Confirm Import Preferences dialog, use the Search icon to select the JDeveloper version from which you want to copy preferences. Then, click **Yes** to automatically migrate portlet customization data of preconfigured portlet producers like OmniPortlet and Web Clipping, from the older JDeveloper version listed in the dialog. You can choose a different JDeveloper version by clicking the **Show All Installations** button and then selecting the required version.

If you select **No**, be aware that portlet customizations are not migrated from the previous version of JDeveloper. In this case, you must manually migrate customizations. For information about migrating customizations manually from the default location, see Section 15.4.2.4.1, "Migrating Customizations from the Default Location."

> **Note:** The JDeveloper migration utility migrates portlet customizations only for preconfigured portlet producers from the default location, `10.1.3.x_jdev_install_dir/portal/portletdata`.
>
> If you used a nondefault location to store customizations of preconfigured portlet producers, you must manually migrate the customizations. To migrate customizations of portlet producers other than preconfigured portlet producers, you need to use the preference store migration utility.
>
> For information about:
>
> - Migrating preconfigured portlet producers' customizations manually from the default location, see Section 15.4.2.4.1, "Migrating Customizations from the Default Location."
>
> - Migrating preconfigured portlet producers' customizations from a nondefault location, see Section 15.4.2.4.2, "Migrating Customizations from a Nondefault Location."
>
> - Migrating customizations for portlet producers other than preconfigured portlet producers, see Section 15.4.3.1, "Migrating Customizations."

d. From the **Help** menu, select **Check for Updates**.

e. On the Welcome page of the Check for Updates wizard, click **Next**.

f. On the Source page, select **Search Update Centers** and click **Next**.

g. On the Updates page, search for the WebCenter Portal extension, select it, and click **Finish**.

> **Note:** For more information about obtaining and installing WebCenter Portal extension, see the WebCenter Portal page on Oracle Technology Network (OTN) at:
>
> http://webcenter.oracle.com

h. Click **Yes** when prompted to restart JDeveloper 11*g*.

i. In the Confirm Import Preferences dialog, click **Yes** when asked if you want to import preferences from a previous version of JDeveloper. These preferences include the portlet customization data.

JDeveloper is now configured to create WebCenter Portal consumer and Portlet Producer applications.

15.3 Upgrading WebCenter Consumer Applications

Table 15–1 lists the tasks involved in upgrading a WebCenter 10.1.3.x consumer application to Oracle WebCenter Portal 11*g*.

Table 15–1   Task Flow for Upgrading a WebCenter 10.1.3.x Consumer Application

Task	Sub Task	When to Perform the Task?
Preparing Your Application for Upgrade	Backing Up Your Applications	Always
	Preparing Applications That Rely on Oracle Content DB	If your WebCenter 10.1.3.2 or 10.1.3.3 application contains Oracle Content Database (Oracle Content DB) connections.
	Preparing Applications That Contain Portlets	If your WebCenter application contains portlets.
Upgrading the Application	Upgrading Your WebCenter Application	Always
Performing Post Application Upgrade Tasks	Configuring Application Settings for Customizable Components	If your WebCenter application contains customizable components.
	Upgrading Oracle Portal Connections	If your WebCenter application contains Oracle Portal connections.
	Configuring ADF Security	If your WebCenter application is secured
	Upgrading Producer Registrations of Preconfigured Portlet Producers	If preconfigured portlet producers are not deployed to the default port, 6688.
	Redeploying Your Applications	Always

This section contains the following subsections:

- Preparing Your Applications for Upgrade

- Upgrading Your WebCenter Application

- Performing Post Application Upgrade Tasks

15.3.1 Preparing Your Applications for Upgrade

When you open a WebCenter 10.1.3.x application in JDeveloper 11*g*, the application is automatically upgraded to Oracle WebCenter Portal 11*g*. However, you must back up your application before you upgrade it. Further, you may need to perform certain tasks before the upgrade, depending on whether your application contains portlets or relies on Oracle Content DB.

This section describes the following tasks:

- Backing Up Your Applications

- Preparing Applications That Rely on Oracle Content DB

- Preparing Applications That Contain Portlets

> **Note:** Make sure that you keep your JDeveloper 10.1.3.x installation when you upgrade your applications. To prepare your applications for migration, you may be required to perform certain tasks in JDeveloper 10.1.3.x.

15.3.1.1 Backing Up Your Applications

It is important to back up your application because after you upgrade a WebCenter 10.1.3.x application to Oracle WebCenter Portal 11*g*, you cannot open it in an earlier release of JDeveloper 11*g*. Also, the changes made to the application during the upgrade cannot be reverted. If you use a source control system, a separate backup may not be necessary.

15.3.1.2 Preparing Applications That Rely on Oracle Content DB

> **Note:** If your application does not contain Oracle Content DB connections, you may skip this section.

To work with WebCenter 10*g* applications that rely on Oracle Content DB:

- Ensure that you have Oracle Content Server installed. Oracle WebCenter Portal 11*g* does not support Oracle Content DB as a content repository. If your WebCenter 10*g* applications rely on Oracle Content DB for content integration, then you must use Oracle Content Server 10.1.3.5.1 or Oracle WebCenter Content Server 11*g* as a content repository for such applications. For information about installing or upgrading Oracle Content Server, see Section 7.6.2, "Upgrading Oracle Content Server."

- Migrate your application data from Content DB to Oracle Content Server, as described in Section 15.5.2, "Migrating Data from Oracle Content DB."

- Upgrade your application, as described in this section.

You cannot directly upgrade WebCenter 10.1.3.2 or 10.1.3.3 applications, which rely on Oracle Content DB, to Oracle WebCenter Portal 11*g*. You must upgrade these applications to Oracle WebCenter 10.1.3.4 to get the support for adding Oracle Content Server as the connection type for content repository data controls.

To upgrade your WebCenter 10.1.3.2 or 10.1.3.3 application to Oracle WebCenter 10.1.3.4:

1. Open your application in JDeveloper 10.1.3.4 and follow the instructions in the upgrade wizard.

2. Edit the existing content repository data control in the upgraded application to point to Oracle Content Server. To do this, you must change the connection type to Oracle Content Server and redefine the custom attribute definitions. In the upgraded WebCenter 10.1.3.4 application, you must use the same name and value for custom attributes that you used in your WebCenter 10.1.3.2 or 10.1.3.3 application.

   For information about content repository connections, see the sections "Editing Content Repository Data Controls" and "How to Create a Content Repository Connection Based on the Oracle Content Server Adapter" in *Oracle Fusion Middleware Developer's Guide for Oracle WebCenter Portal*.

After you have performed these tasks, your WebCenter 10.1.3.4 application is ready to be upgraded to Oracle WebCenter Portal 11*g*, as described in Section 15.3.2, "Upgrading Your WebCenter Application."

15.3.1.3 Preparing Applications That Contain Portlets

> **Note:**   If your WebCenter 10*g* applications do not contain portlets, you can skip this section.

In JDeveloper 11*g*, WebCenter applications do not contain the Portlets project. However, if your WebCenter 10.1.3.x application contains portlets, it will continue to include the Portlets project even after it has been upgraded. You must manually remove the Portlets project from your WebCenter application before you upgrade it.

> **Note:**   To upgrade portlet producers, you will need to create a portlet producer application in Oracle WebCenter Portal 11*g*. You will need a copy of the Portlets project of your 10.1.3.x application while upgrading portlet producers as described in Section 15.4.2.2, "Upgrading Portlet Producers Created in JDeveloper."

To remove the Portlets project from your WebCenter 10.1.3.x application:

1. Start JDeveloper 10.1.3.x.

> **Note:**   To remove the Portlets project, you must use JDeveloper 10.1.3.x, and not JDeveloper 11*g*.

2. Open the WebCenter 10.1.3.x application that you want to upgrade.

3. In the **Application Navigator**, select the **Portlets** project.

4. From the **File** menu, select **Erase from Disk**.

5. Click **Yes** to delete the project from the application.

6. Save your application.

15.3.2 Upgrading Your WebCenter Application

> **Note:**   Grants are not migrated properly if a 10.1.3.x WebCenter application contains grants without any permissions. Prior to upgrading your application, you must inspect the `app-jazn-data.xml` file in the 10.1.3 workspace and remove any grants that have empty permission set.

After you have prepared your WebCenter 10.1.3.x application, you can upgrade it to Oracle WebCenter Portal 11*g*.

To upgrade your WebCenter 10.1.3.x application to Oracle WebCenter Portal 11*g*:

1. Start JDeveloper 11*g*.

2. Open your WebCenter 10.1.3.x application.

This invokes the upgrade wizard, which is displayed every time a WebCenter 10.1.3.x application is opened in JDeveloper 11*g*.

3. On the Welcome page, click **Next**.

4. On the Confirmation page, the **Yes** option is selected by default, as shown in Figure 15–1. Click **Next** to confirm that you want to upgrade the application and its projects.

   If you select **No**, the migration process is aborted and the application is not opened in JDeveloper.

Figure 15–1 Migration Wizard - Confirmation Page
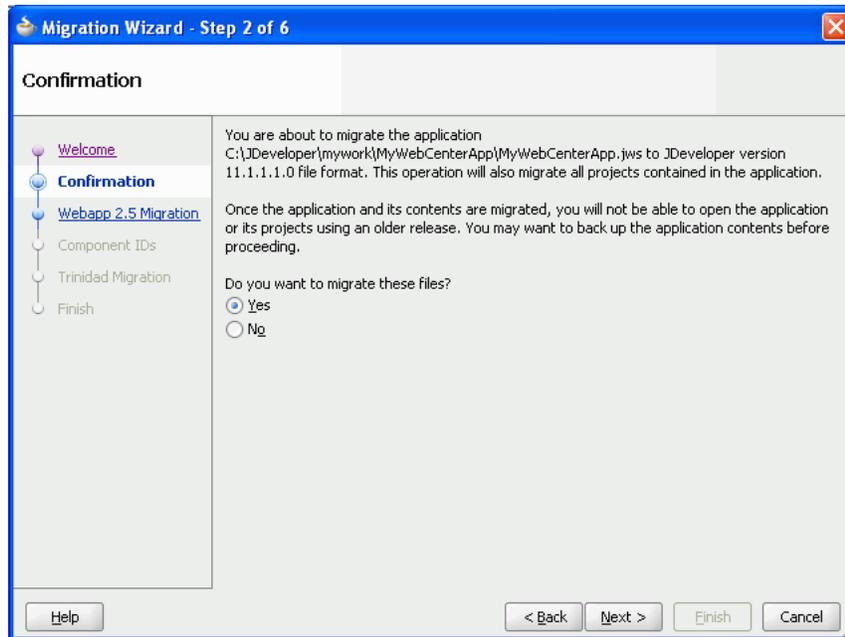
5. On the Webapp 2.5 Migration page, specify whether you want to migrate projects created using JavaServer Pages Standard Tag Library (JSTL) version 1.0 or 1.1, as shown in Figure 15–2. To accept the default setting, click **Next**.

Figure 15–2   Migration Wizard - Webapp 2.5 Migration Page
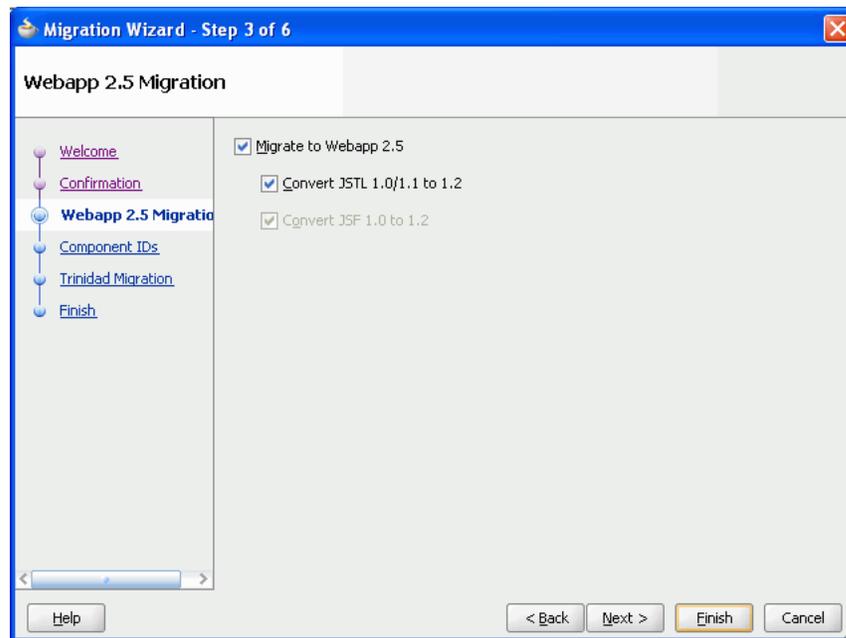
6. On the Component IDs page, specify whether you want to migrate and randomize component IDs. Click **Next** to accept the default settings. (Figure 15–3)

Figure 15–3   Migration Wizard - Component IDs Page
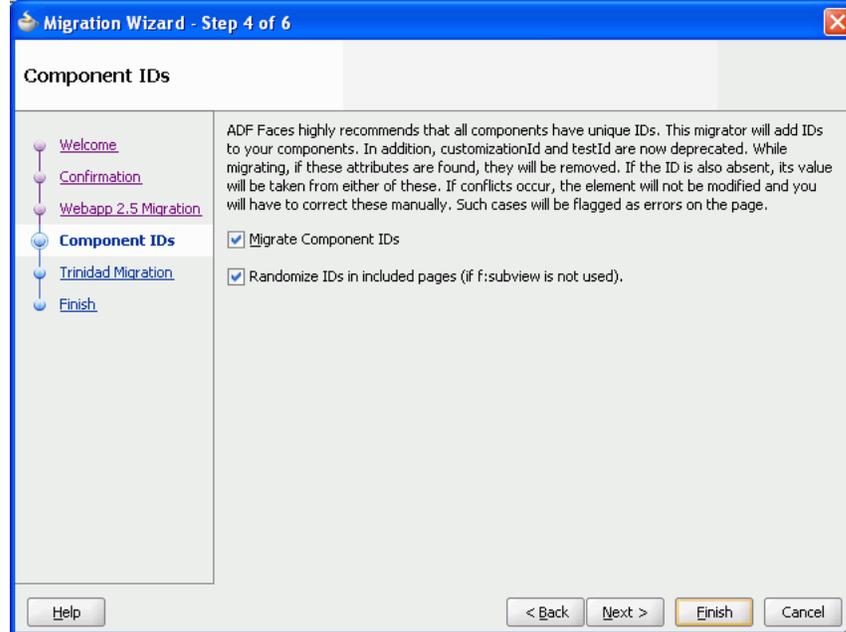
7. On the Trinidad Migration page, click **Next**. (Figure 15–4)

Figure 15–4   Migration Wizard - Trinidad Migration Page
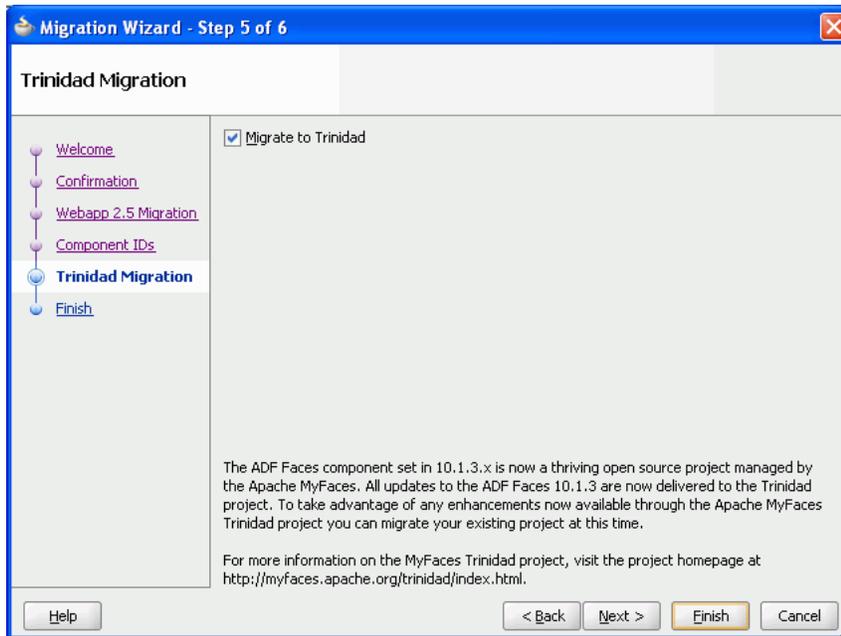
8. Click **Finish** to begin upgrading your WebCenter application. (Figure 15–5)

   It might take a while for the upgrade process to complete, depending on the size of the application. A progress dialog displays while the upgrade process executes. When the application upgrade is complete, the Migration Status dialog displays a list of projects that have been upgraded, as shown in Figure 15–6.

Figure 15–5   Migration Wizard - Finish Page
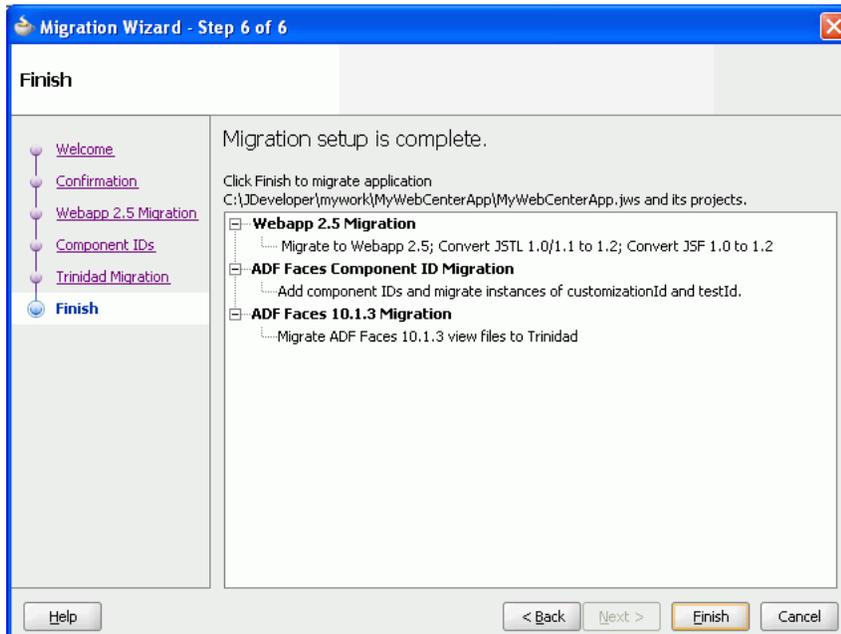
9. Click **OK**.

Figure 15–6   Migration Status
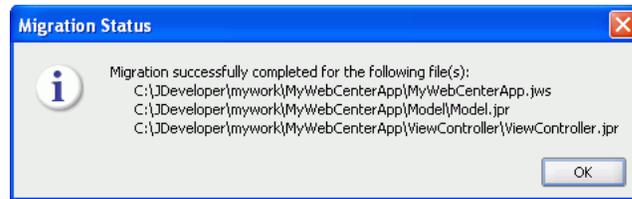
The upgraded application is opened and its projects are listed in the Application Navigator. Notice that Figure 15–6 and Figure 15–7 do not list the Portlets project in the list of migrated projects.

If there are any errors during application upgrade, they are listed in the Message - Log window.

Figure 15–7   Projects Upgraded to JDeveloper 11g
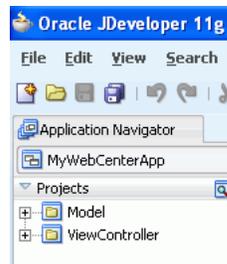
> **Note:** By default, a WebCenter 10.1.3.x application is configured to use the `suede` skin, whereas a WebCenter Portal 11*g* application is configured to use the `fusion` skin. When you upgrade a WebCenter 10.1.3.x application, the `suede` skin setting is retained in the upgraded application to provide the look and feel similar to that of a 10.1.3.x application. If you want to use the ADF rich skin `fusion` in your upgraded application, update the skin settings in the `trinidad-config.xml` file.

15.3.3 Performing Post Application Upgrade Tasks

While upgrading your application, the upgrade utility in JDeveloper 11*g* makes various changes to your application to configure it for Oracle WebCenter Portal 11*g*. Some of these changes are related to the following:

- Customizable components
- External applications
- Portlet component changes

For information about these changes, see Chapter 16, "Additional Oracle WebCenter Upgrade Details."

After upgrading your WebCenter consumer application, you may need to perform various post upgrade tasks. These tasks include:

- Configuring Application Settings for Customizable Components
- Moving Resource Catalogs from an Application's MDS to a Project Directory

- [Upgrading Oracle Portal Connections](#)
- [Configuring ADF Security](#)
- [Upgrading Producer Registrations of Preconfigured Portlet Producers](#)
- [Redeploying Your Applications](#)

15.3.3.1 Configuring Application Settings for Customizable Components

If your upgraded WebCenter application contains customizable components such as ShowDetailFrame or PanelCustomizable, you must update the `web.xml` to ensure that customizations or personalizations related to customizable components continue to work. To do this, you must replace `ComposerChangeManager` with `MDSDocumentChangeManager`.

In `web.xml`, replace the following context-parameter entries:

```
<context-param>
  <param-name>org.apache.myfaces.trinidad.CHANGE_PERSISTENCE</param-name>
  <param-value>oracle.adf.view.page.editor.change.ComposerChangeManager
  </param-value>
</context-param>
```

With:

```
<context-param>
  <param-name>org.apache.myfaces.trinidad.CHANGE_PERSISTENCE</param-name>
  <param-value>oracle.adf.view.rich.change.MDSDocumentChangeManager</param-value>
</context-param>
```

15.3.3.2 Moving Resource Catalogs from an Application's MDS to a Project Directory

In Oracle WebCenter Portal 11*g*, WebCenter applications support a Resource Catalog visual editor. In your upgraded application, if you want to use this visual editor, you must manually move resource catalogs from your upgraded application's MDS directory to one of its project directories.

To move resource catalogs from MDS to a project directory in an upgraded application:

1. On your file system, navigate to the resource catalog definitions of your upgraded application. The default location of resource catalogs is *app-root*/mds/oracle/adf/rc/metadata.

2. On your file system, move the catalog definitions to a Java package under the required project in your application.

   It is recommended that you do not change the relative paths for catalogs. For example, move:

   app_root/mds/oracle/adf/rc/metadata/default-catalog.xml

   to

   app_root/project/src/oracle/adf/rc/metadata/default-catalog.xml

3. Open `adf-config.xml` from the location *app-root*/.adf/META-INF.

4. Remove the entries highlighted in bold:

   ```
   <metadata-namespaces>
   ```

```
....
<namespace path="/oracle/adf/rc/metadata"
           metadata-store-usage="WebCenterFileMetadataStore"/>
....
</metadata-namespaces>
```

> **Note:** Even when you move resource catalogs from MDS to a project directory, they are loaded through MDS. To enable MDS to be able to locate your catalogs, the Java package you choose must not correspond to any MDS namespace mappings in `adf-config.xml`. MDS namespace mappings are stored under `<metadata-namespaces>` in `adf-config.xml`.

5. In your WebCenter 10*g* application, if catalogs were stored in a different MDS namespace, remove the mappings for those namespaces and/or adjust the new location of the catalogs so the new locations do not correspond with any MDS namespace mappings.

6. If you changed the relative location of the default catalog, update (or add) the following entries in `adf-config.xml`:

```
<rcv-config xmlns="http://xmlns.oracle.com/adf/rcs/viewer/adf-config">
  <default-catalog catalog-name="path_to_your_catalog_default-catalog.xml"/>
</rcv-config>
<adf-rcs-config xmlns="http://xmlns.oracle.com/adf/rcs/adf-config">
<rcs-config>
<catalog-config
default-scope="path_prefix"/>
</rcs-config>
</adf-rcs-config>
```

You need to prefix *path_prefix* to all catalog IDs to create an MDS reference. The value of *path_prefix* may be "/" in which case all catalog paths must be fully qualified MDS references (including those returned by your `ResourceCatalogSelector`).

7. If you have a `CatalogSelector` class, update the implementation to reflect the new catalog locations. If you have a `CatalogSelector` class, the `<rcv-config>` element contains the following entry:

```
<catalog-selector class-name="name_of_your_CatalogSelectorClass"/>
```

8. Save `adf-config.xml`.

15.3.3.3 Upgrading Oracle Portal Connections

For content repository connections based on the Oracle Portal adapter, the data source needs to be upgraded to a database connection. When you start JDeveloper 11*g* for the first time, it asks you whether you want to upgrade settings from a previous version of JDeveloper.

- If you choose **Yes**, database connections are automatically upgraded and are accessible from IDE Connections in JDeveloper 11*g*.

- If you choose **No**, database connections are not upgraded.

When you upgrade your WebCenter application, the upgrade utility uses the upgraded database connections if you chose **Yes** at the JDeveloper prompt for

upgrading the settings. If you did not choose to upgrade the settings from a previous version of JDeveloper, the upgrade utility attempts to create a database connection by using the information stored in the application's `data-sources.xml` file(s). In this case, the password for the database connection is not available. Therefore, after upgrading your application, you must specify the password for the database connection by using the Edit Database Connection wizard.

In the unlikely event where the database connection used by the Oracle Portal adapter was not automatically upgraded or created, then after upgrading the application, you must create a new database connection by using the Create Database Connection wizard, which is accessible through the Edit Content Repository Connection wizard. For information about how to create a database connection, see the "How to Create a Content Repository Connection Based on the Oracle Portal Adapter" section in *Oracle Fusion Middleware Developer's Guide for Oracle WebCenter Portal*.

15.3.3.4 Configuring ADF Security

If your WebCenter 10.1.3.x application is ADF-secured, you must reconfigure security after upgrading the application.

When you upgrade an ADF-secured WebCenter application, ADF security policies are upgraded. The policies defined in the `approot/.adf/META-INF/app-jazn-data.xml` are upgraded to `approot/src/META-INF/jazn-data.xml`. However, the users and enterprise roles defined in the policies are not upgraded.

Figure 15–8 shows the properties in `jazn-data.xml` of an upgraded WebCenter application.

Figure 15–8   Security Settings of an Upgraded WebCenter Application
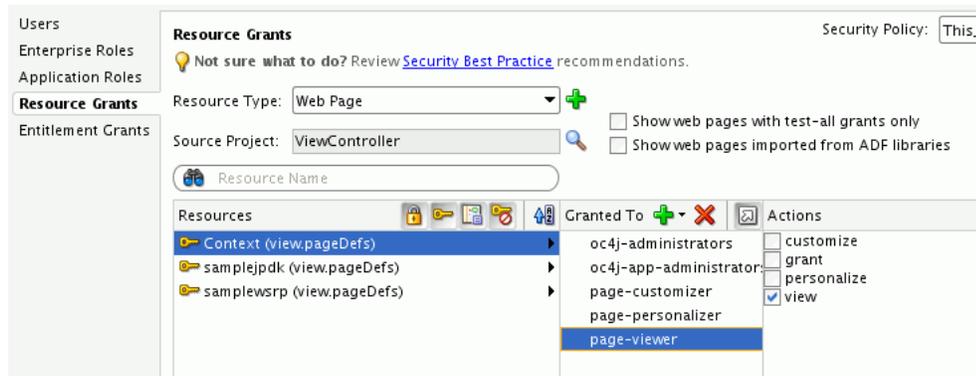
To reconfigure ADF security in an upgraded WebCenter application:

- Create a new realm in the jazn-data editor, if it does not already exist. Then re-create the required users and enterprise roles that existed in the security policies of your WebCenter 10.1.3.x application. Assign the newly created users to the enterprise roles as required. It is recommended that you name the realm as `jazn.com`.

    Creation of users and roles is necessary only to test the application in Integrated WebLogic Server (WLS) in JDeveloper. Typically, the required users and enterprise roles used in application policies are expected to exist in the production setup where the application is deployed.

> **Note:** The `valid-users` role, which specifies all authenticated
> users and usually maps to the `users` role in `weblogic.xml`, is not
> created in `web.xml` of the upgraded application, thereby restricting
> access to all authenticated users. If you want all authenticated users to
> have access to the upgraded application, you must manually create
> the `valid-users` role in `web.xml` and map it to the `users` role in
> `weblogic.xml`.

- Optionally, you can reconfigure application authorization data to use application
  roles, which are supported by Oracle WebCenter Portal 11*g* applications. You can
  reconfigure ADF security to use application roles instead of enterprise roles.

For more information about how to configure ADF security, see *Oracle Fusion
Middleware Fusion Developer's Guide for Oracle Application Development Framework*.

Considerations When Upgrading ADF-Secured WebCenter Applications   Grants are
not migrated properly if a 10.1.3.x WebCenter application contains grants without any
permissions. Prior to upgrading your application, you must inspect the
`app-jazn-data.xml` file in the 10.1.3 workspace and remove any grants that have
empty permission set.

In Oracle WebCenter 10.1.3.x, the ADF framework performed the rowset, attribute,
and method permission checks in addition to page permission checks. If a 10.1.3
WebCenter application grants the 'read' permission on the rowset and attribute and
the 'invoke' permission on the method for all users, then the application functions as
expected in Oracle WebCenter Portal 11*g* without any additional setup.

However, if the 10.1.3.x WebCenter application was designed to allow only certain
users to view the rowset, attribute, or invoke method, then a special flag needs to be
set to support this style of security. If this flag is not set, then anyone who has page
access can view attributes and rowsets and invoke methods because in Oracle
WebCenter Portal 11*g* the permission check is performed only on pages and task
flows. The flag must be set for each application in the `adf-config.xml` file, as
shown in the following example:

```
<sec:adf-security-child xmlns="http://xmlns.oracle.com/adf/security/config">
 <JaasSecurityContext
  initialContextFactoryClass="oracle.adf.share.security.JAASInitialContextFactory"
  jaasProviderClass="oracle.adf.share.security.providers.jps.JpsSecurityContext"
  authorizationEnforce="true"/>
 <contextEnv name="oracle.adf.security.metadata" value="false"/>
 <CredentialStoreContext
  credentialStoreClass=
  "oracle.adf.share.security.providers.jps.CSFCredentialStore"
  credentialStoreLocation="../../src/META-INF/jps-config.xml"/>
</sec:adf-security-child>
```

You must also ensure that there are no duplicate `JaasSecurityContext` and
`CredentialStoreContext` elements in the `adf-config.xml` file.

15.3.3.5 Upgrading Producer Registrations of Preconfigured Portlet Producers

During application upgrade, the port numbers of all preconfigured portlet producers
(such as Web Clipping and OmniPortlet) are updated. All producer registrations from
preconfigured portlet producers that existed in Oracle Application Server 10.1.3.x
environment with port 6688 are migrated to port 7101. If you did not use default port

numbers in your Oracle Application Server 10.1.3.x environment, you must manually change those port numbers to appropriate port numbers.

When you upgrade a WebCenter application, port changes are not made to registrations of portlet producers other than preconfigured portlet producers.

15.3.3.6 Redeploying Your Applications

After upgrading your application to Oracle WebCenter Portal 11*g*, you must recompile the application. To prepare your application for redeployment, create a WebLogic Managed Server instance and provision it with a required set of shared libraries. Also, create and register the Metadata Service (MDS) repository for your application on the WebLogic Domain's Administration Server instance. You must then redeploy the application and verify that it has been deployed properly. For information, see the "Deploying WebCenter Portal: Framework Applications" chapter in *Oracle Fusion Middleware Administrator's Guide for Oracle WebCenter Portal*.

15.4 Upgrading Portlet Producers

If your WebCenter 10.1.3.x application uses portlets, then in addition to upgrading your WebCenter consumer application, you may need to upgrade portlet producers to Oracle WebCenter Portal 11*g*.

Table 15–2 lists the tasks involved in upgrading portlet producers used by your WebCenter 10.1.3.x applications.

Table 15–2   Task Flow for Upgrading Portlet Producers

Task	SubTask	When to Perform the Task?
Preparing Your Application	Determining WebCenter Consumer Application and Portlet Producer Compatibility	Always
Upgrading Your Portlet Producer Application	Upgrading Portlet Producers Created in JDeveloper	If your WebCenter application contains portlets provided by portlet producers developed in JDeveloper 10.1.3.x.
	Upgrading Portlet Producers Created Outside of JDeveloper	If your WebCenter application contains portlets provided by portlet producers developed outside of JDeveloper.
	Upgrading Preconfigured Portlet Producers	If your WebCenter application contains portlets provided by preconfigured portlet producers.
Performing Post Upgrade Tasks	Migrating Customizations	If your WebCenter application contains portlets other than the portlets provided by preconfigured portlet producers.
	Redeploying a Portlet Producer Application	Always

15.4.1 Determining WebCenter Consumer Application and Portlet Producer Compatibility

In most cases, WebCenter consumer applications or portlet producer applications deployed to Oracle Application Server 10.1.3.x are compatible with applications upgraded to Oracle Fusion Middleware 11*g*. For example, a WebCenter consumer application that has been upgraded and deployed to Oracle WebLogic Server in Oracle Fusion Middleware 11*g* may still use portlets from PDK-Java producer applications running on Oracle Application Server 10.1.3.x.

Oracle Fusion Middleware 11*g* supports backward compatibility for WebCenter consumer applications and portlet producer applications. Based on the compatibility supported, you may decide to upgrade only your WebCenter consumer applications, only portlet producer applications, or both. Table 15–3 lists the compatibility between different versions of WebCenter consumer applications and portlet producer applications.

Table 15–3 shows that a WebCenter 10.1.3.x consumer application and a PDK-Java or a WSRP 1.0 WebCenter Portal 11*g* portlet producer application are compatible. In such a case where backward compatibility is supported, you may choose not to upgrade your WebCenter 10.1.3.x consumer application to WebCenter Portal 11*g*. The table further shows that a WebCenter Portal 11*g* consumer application and a PDK-Java or a WSRP 1.0 WebCenter 10.1.3.x portlet producer application are compatible. In such a case, you may choose not to upgrade your 10.1.3.x portlet producer application.

However, WebCenter 10.1.3.x consumer applications are not compatible with WSRP 2.0 WebCenter Portal 11*g* portlet producer applications. Similarly, WebCenter Portal 11*g* consumer applications are not compatible with WSRP 2.0 WebCenter 10.1.3.x portlet producer applications. In such cases, all 10.1.3.x WSRP 2.0 applications must be upgraded to Oracle Fusion Middleware 11*g* for these applications to work together.

Table 15–3   Compatibility Between Different Versions of WebCenter Consumer and Portlet Producer Applications

Version of WebCenter Consumer Application[1]	Version of Portlet Producer Application	Is this Combination of Applications Supported in Oracle Fusion Middleware 11*g*?
10.1.3.*x*	PDK-Java 11*g*	Yes
10.1.3.*x*	WSRP 1.0 deployed to Oracle WebLogic Server 11*g*	Yes
10.1.3.*x*	WSRP 2.0 deployed to Oracle WebLogic Server 11*g*	No
11*g*	PDK-Java 10.1.3.x	Yes
11*g*	WSRP 1.0 deployed to Oracle Application Server 10.1.3.x	Yes
11*g*	WSRP 2.0 deployed to Oracle Application Server 10.1.3.x	No

[1]   In this column, 11*g* refers to a WebCenter consumer application upgraded to or originally created in Oracle WebCenter Portal 11.1.1.7.0.

15.4.2 Upgrading Portlet Producer Applications

To upgrade portlet producer applications, you may need to perform various tasks depending on the type of portlet producers used in your WebCenter 10.1.3.x application. This section describes those tasks. It contains the following subsections:

- About Upgrading JSR168 Portlet Producers to JSR286
- Upgrading Portlet Producers Created in JDeveloper
- Upgrading Portlet Producers Created Outside of JDeveloper
- Upgrading Preconfigured Portlet Producers

15.4.2.1 About Upgrading JSR168 Portlet Producers to JSR286

Oracle WebCenter 10*g* supports Java portlets based on the Java Portlet Specification, JSR 168; Oracle WebCenter Portal 11*g* supports Java portlets based on Java Portlet Specification version 2, or JSR 286. JSR 286 is an extension of JSR 168, and is backward compatible with JSR 168.

In Oracle WebCenter 10*g*, Oracle JSF Portlet Bridge is based on and conforms to JSR 301, whereas in Oracle WebCenter Portal 11*g*, Oracle JSF Portlet Bridge conforms to JSR 329.

In JDeveloper 11*g*, when you open for the first time an existing portlet producer application containing JSR 168 portlets, portlets are automatically upgraded to be JSR 286 compliant. If the application is a portlet bridge application, it is further automatically upgraded to be JSR 329-compliant.

In most cases, the upgraded portlets continue to work exactly as they did before. However, there are a few cases in which JSR 168 portlets function differently when upgraded to JSR 286; these portlets must invoke a JSR 168 compatibility mode to run under JSR 286.

For more information, see the "JSR-286/JSR-168 Portlet Compatibility" section in *Oracle Fusion Middleware Portlet Development Guide for Oracle WebLogic Portal*.

In Oracle WebCenter 10*g*, a portlet producer application contains the `portlet.xml` and `oracle-portlet.xml` files. When you upgrade a portlet producer application, the `oracle.portlet.xml` file is deleted, and all its details are moved to `portlet.xml`. The navigation parameters stored in `oracle.portlet.xml` are converted into public render parameters and are added to `portlet.xml`. For information about how JSR 168 parameters are handled in an upgraded JSR 286-compliant portlet producer application, see Section 16.4, "Migration of JSR 168 Portlet Producers to JSR 286: Handling of Portlet Elements."

15.4.2.2 Upgrading Portlet Producers Created in JDeveloper

If your WebCenter 10.1.3.x application contains portlets and was created by using JDeveloper, you must create a portlet producer application in Oracle WebCenter Portal 11*g* to upgrade portlet producers. You must also manually add the Portlets project of the WebCenter 10.1.3.x application to the newly created portlet producer application.

> **Note:** If your WebCenter 10*g* application does not contain any portlet producers, its Portlets project will be empty.

To upgrade a portlet producer application created in JDeveloper 10.1.3.x:

> **Note:** To complete this procedure, you need a copy of the Portlets directory of your WebCenter 10.1.3.x application.

1. Create a portlet producer application in JDeveloper 11*g* by using the **WebCenter Portal – Portlet Producer Application** template. For information, see the section "Creating a Portlet Producer Application" in the *Oracle Fusion Middleware Developer's Guide for Oracle WebCenter Portal*.

2. In the newly created portlet producer application, select the **Portlets** project.

3. From the **File** menu, choose **Delete Project**.

4. In the Confirm Delete Project dialog, select the **Remove project and delete all of its contents (including source directories)** radio button and click **Yes**, as shown in Figure 15–9.

Figure 15–9   Deleting the Portlets Project
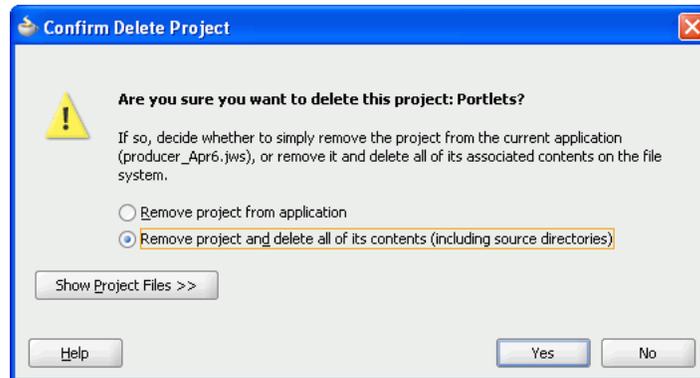
5. Click **Yes** if the message to confirm the delete operation displays.

6. Save your application.

7. On the file system, copy the `Portlets` directory from the backed up copy of your WebCenter 10.1.3.x application to the root directory of the newly created portlet producer application.

8. In JDeveloper 11*g*, open `portlets.jpr` of the portlet producer application.

   JDeveloper prompts you to upgrade the Portlets project.

9. Click **Yes** to upgrade the 10.1.3.x **Portlets** project to the 11*g* portlet producer application, as shown in Figure 15–10.

Figure 15–10   Upgrading the Portlets Project
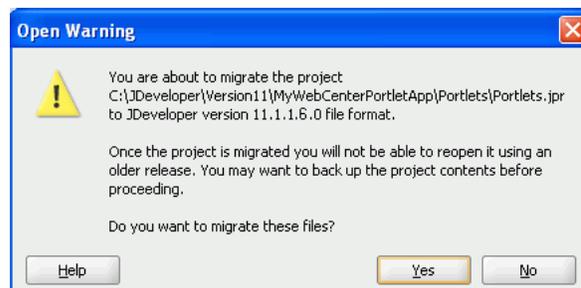
After the **Portlets** project is upgraded, the Migration Status dialog shows that `portlets.jpr` has been upgraded successfully.

10. Click **OK**.

15.4.2.3 Upgrading Portlet Producers Created Outside of JDeveloper

If you have a JSR 168 Java portlet producer application or a PDK-Java portlet application packaged as a WAR file or as a WAR file within an EAR file, then you must update the application with Oracle WebCenter Portal 11*g* specific elements, such as Oracle WebLogic Server application descriptors.

The following sections describe how to upgrade a PDK-Java or JSR 168 application, which is in the form of an EAR or a WAR archive file, to Oracle WebCenter Portal 11*g*.

15.4.2.3.1 Creating a JDeveloper Application from an EAR File If you have a JSR 168 portlet application or a PDK-Java portlet application packaged as a WAR file within an EAR file, then to upgrade that EAR to Oracle WebCenter Portal 11*g*, you first need to create a JDeveloper 10.1.3.x application based on the archive file. You can then upgrade the application by opening it in JDeveloper 11*g*.

To create an application from an EAR file in JDeveloper 10*g* and upgrade it to Oracle WebCenter Portal 11*g*:

1. Open JDeveloper 10.1.3.x.

   > **Note:**   You must use JDeveloper 10.1.3.x to prepare the EAR file for upgrade.

2. From the **File** menu, select **New**.

3. In the New Gallery dialog, expand **General**, select **Applications**, then **Application from EAR File**, and click **OK**.

4. Select **Application from EAR File,** and click **OK**.

5. On the Welcome page in the Create Application from EAR File wizard, click **Next**.

6. On the Location page, in the **EAR File** field, enter the path to the EAR file.

7. Select the **Copy Files to Application** checkbox.

8. Select **Finish** to create the application in JDeveloper.

9. Save the application.

10. Save a backup copy of the application.

11. Open JDeveloper 11*g*.

12. Open the WebCenter 10.1.3.x application that you created from the EAR file.

13. Follow the instructions to upgrade the application to JDeveloper 11*g*.

14. Save the application.

    The application can now be redeployed.

15.4.2.3.2 Creating a JDeveloper Application from a WAR File To create a JDeveloper file from a JSR 168 or a PDK-Java application packaged as a WAR file:

1. Open JDeveloper 10.1.3.x.

> **Note:** You must use JDeveloper 10.1.3.x to prepare the WAR file for upgrade.

2. From the **File** menu, choose **New**.

3. In the New Gallery dialog, expand **General**, select **Applications**, then **Application**, and click **OK**.

4. In the Create Application wizard, click **OK**.

5. In the Create Project wizard, click **Cancel** so that no new project is created in the application.

6. In the **Application Navigator**, select the newly created application.

7. From the **File** menu, choose **New**.

8. In the New Gallery, expand **General**, select **Project**, then **Project from WAR File**, and click **OK**.

9. On the Welcome page in the Create Project from WAR File wizard, click **Next**.

10. On the WAR Location page, in the **WAR File** field, specify the path to the WAR file.

11. Click **Finish** to create the project.

12. Save the application.

13. Back up your application.

14. Open JDeveloper 11*g*.

15. Open the WebCenter 10.1.3.x application that you created from the WAR file.

16. Follow the instructions to upgrade the application in JDeveloper 11*g*.

17. Save the application.

    The application can now be redeployed.

15.4.2.4 Upgrading Preconfigured Portlet Producers

Oracle WebCenter Portal 11*g* provides various preconfigured portlet producers. These include OmniPortlet, Web Clipping, WSRP Parameter Form Portlet, sample WSRP portlet producers, and sample PDK-Java portlet producers.

Oracle WebCenter enables you to customize portlets. Customizations can include preferences such as user data and portlet and producer settings. Customizations can be stored in a database or a file system. Customizations related to preconfigured portlet producers are saved in the customization store of these producers, and not within your application projects. By default, in Oracle WebCenter 10*g*, customizations for preconfigured portlet producers are stored at the following location:

```
10.1.3.x_jdev_install_dir/portal/portletdata
```

Where, `10.1.3.x_jdev_install_dir` refers to the JDeveloper 10.1.3.x installation directory.

So, when you upgrade a WebCenter application containing portlets provided by preconfigured portlet producers, you must ensure that portlet customizations are also migrated.

15.4.2.4.1 Migrating Customizations from the Default Location After installing JDeveloper 11*g*, when you start it the first time, it prompts whether you want to migrate settings from a previous release. If you select **Yes**, portlet customizations of preconfigured portlet producers, along with other JDeveloper system properties, are automatically migrated from a previous JDeveloper installation to JDeveloper 11*g*. However, this migrates portlet customizations only from the default location, `10.1.3.x_jdev_install_dir`/portal/portletdata.

If you choose not to migrate portlet customizations and select **No**, you must migrate customizations later manually. The manual procedure involves copying the `10.1.3.x_jdev_install_dir`/portal/portletdata directory to the `11g_jdev_install_dir`/jdeveloper/portal/portletdata directory.

15.4.2.4.2 Migrating Customizations from a Nondefault Location You may choose to store portlet customizations at a different location instead of the default location, `10.1.3.x_jdev_install_dir`/portal/portletdata.

To migrate customizations from a nondefault location, you must perform either of the following tasks:

- Migrating customizations to a new location: Copy the customization store directory to the required path and configure your upgraded portlet producer to point to the new location of the customization store. If customizations are stored in a database, configure your upgraded portlet producer to access that database.

- Using customizations from the existing location: Configure your upgraded portlet producer to access the customization store from the old location.

For information about storing customizations at a nondefault location, see Section 16.3, "Preconfigured Portlet Producers: Customization Store's Location."

15.4.3 Performing Post Upgrade Tasks for Portlet Producers

After upgrading portlet producers, you need to migrate customizations of portlet producers other than preconfigured ones and redeploy your portlet producer applications.

15.4.3.1 Migrating Customizations

After upgrading Oracle PDK-Java and WSRP portlet producers, you must migrate their customizations if the customizations are not shared or accessible to the upgraded portlet producers. These customizations are of portlet producers other than the preconfigured portlet producers.

To migrate such customizations, you can use the preference store migration utility. For information, see the "Portlet Preference Store Migration Utilities" section in *Oracle Fusion Middleware Developer's Guide for Oracle WebCenter Portal*.

15.4.3.2 Redeploying a Portlet Producer Application

You can deploy your upgraded portlet producer application to any Oracle WebLogic Server managed server configured to support Oracle WebCenter portlet producers. For deployment, you can use Oracle Enterprise Manager Fusion Middleware Control, Oracle WebLogic Server Administration Console, or Oracle WebLogic Scripting Tool (WLST). For information, see the "Deploying Portlet Producer Applications" section in *Oracle Fusion Middleware Administrator's Guide for Oracle WebCenter Portal*.

You can also deploy portlet producer applications to an Oracle WebLogic Server instance directly from a development environment by using JDeveloper, provided you have the required credentials to access the WebLogic server. For information, see the

"Deploying a Portlet Producer Application to an Oracle WebLogic Managed Server Instance" section in *Oracle Fusion Middleware Developer's Guide for Oracle WebCenter Portal*.

15.5 Migrating Data

If your WebCenter 10*g* application contains WebCenter services that rely on a back-end component, you may need to prepare the back-end component for upgrade, and migrate the data. This section describes how to migrate your WebCenter 10*g* data to the required servers. It contains the following subsections:

- Section 15.5.1, "Migrating Data from Oracle WebCenter Wiki and Blog Server"
- Section 15.5.2, "Migrating Data from Oracle Content DB"

15.5.1 Migrating Data from Oracle WebCenter Wiki and Blog Server

> **Note:** If Oracle WebCenter Wiki and Blog Server is not installed in your WebCenter 10*g* environment, you can skip this section.

In your WebCenter 10*g* applications, you expose wiki and blog functionality through the Wiki and Blog services. Oracle WebCenter 10*g* relies on Oracle WebCenter Wiki and Blog Server to provide the wiki and blog functionality. Oracle WebCenter Portal 11*g* exposes the wiki and blog functionality through the Documents service, and requires Content Server 11*g* as the content repository for this purpose.

When you upgrade a WebCenter 10.1.3.x application containing the Wiki and Blog services, you must also migrate the wiki data from Oracle WebCenter Wiki and Blog Server to Oracle WebCenter Content Server 11*g*. You can migrate the wiki and blog data any time after upgrading the middle tier.

For information about how to migrate the wiki and blog data from Oracle WebCenter Wiki and Blog Server to Oracle WebCenter Content Server 11*g*, refer to the section "Migrating Oracle Wiki Pages and Blogs" in *Oracle Fusion Middleware Patching Guide*.

15.5.2 Migrating Data from Oracle Content DB

If your WebCenter 10*g* applications rely on Oracle Content DB, you must use Oracle Content Server version 10.1.3.5.1 or 11*g* as your content repository. You can migrate Content DB data any time after upgrading the middle tier.

To migrate Oracle Content DB data to Oracle Content Server 10.1.3.5.1 by using a WebDav client:

1. Open the following WebDAV locations with a user account that has access to all relevant content:

   - Location on Oracle Content DB: `http://server:host/content/dav`

   - Location on Oracle Content Server:
     `http://server:host/content-server-root/idcplg/webdav`

2. Copy the content from the Oracle Content DB location to the Oracle Content Server location.

When you migrate content by using WebDav, be aware of the following:

- The Trash folder gets copied.

- For the versioned documents, only the latest version gets copied.

- Metadata and access control setting are not preserved for the copied content.

> **Note:** You cannot directly migrate data from Oracle Content DB to Oracle WebCenter Content Server 11*g*. If you want to use Oracle WebCenter Content Server 11*g* as your content repository, you must first migrate Content DB data to Oracle Content Server 10.1.3.5.1, and then upgrade to Oracle WebCenter Content Server 11*g*. For information about upgrading from Oracle Content Server 10*g* to 11*g*, refer to *Oracle Fusion Middleware Upgrade Guide for Oracle WebCenter Content*.

16

Additional Oracle WebCenter Upgrade Details

This chapter describes the changes that the upgrade utility in Oracle JDeveloper 11*g* automatically makes to a WebCenter 10.1.3.x application while upgrading the application. It also describes how you can change the location of customization store for preconfigured portlet producers.

The chapter contains the following sections:

- Automated Changes in Upgraded WebCenter Applications
- Content Integration Connections: Migration of Credentials to External Applications
- Preconfigured Portlet Producers: Customization Store's Location
- Migration of JSR 168 Portlet Producers to JSR 286: Handling of Portlet Elements

> **Note:** This chapter uses recently rebranded Oracle WebCenter Portal product names where appropriate. For more information, see Section 3.3.1, "New Product Names for Oracle WebCenter and Oracle Enterprise Content Management Suite."

16.1 Automated Changes in Upgraded WebCenter Applications

To upgrade a WebCenter 10.1.3.x application to Oracle WebCenter Portal 11*g*, you use JDeveloper 11*g*. While upgrading an application, the upgrade utility in JDeveloper 11*g* automatically makes certain changes, such as those related to customizable components, external applications, and portlet components.

This section describes the following changes:

- Customizable Component Changes
- External Application Changes
- Portlet Component Changes

16.1.1 Customizable Component Changes

In WebCenter applications, you can use Oracle Composer to add components to pages that users can customize at runtime. Oracle Composer provides components like Panel Customizable and Show Detail Frame that users can add to applications to support page personalization and customization.

When you upgrade a WebCenter 10.1.3.x application that uses customizable components, JDeveloper automatically makes various changes related to customizable components to prepare the application to run on Oracle WebLogic Server. These changes include:

- General Changes
- Panel Customizable Changes
- Show Detail Frame

16.1.1.1 General Changes

When you upgrade your WebCenter application, JDeveloper makes the following general changes to your application:

- The namespace of customizable components in JSPX files is renamed from:

  `http://xmlns.oracle.com/adf/faces/customizable`

  To

  `http://xmlns.oracle.com/adf/faces/customizable/html`

- From `web.xml`, entries for the following are removed:

    - Context parameter `oracle.mds.web-app-root`

    - Filter `mdsFilter`

    - Filter mapping for `mdsFilter`

    - Servlet `jsp2`, servlet mapping for `jsp2`, and resource proxy

- The Java Customizable Components library is removed and the WebCenter Customizable Components library is added.

- The old Customizable Components tag libraries are removed and the new version Customizable Components (HTML) is added.

- In import statements and declarations, if any, in Java files, the class name of Panel Customizable is changed to `oracle.adf.view.html.component.customizable.PanelCustomizable`.

- In import statements and declarations, if any, in Java files, the class name of Show Detail Frame is changed to `oracle.adf.view.html.component.customizable.ShowDetailFrame`.

16.1.1.2 Panel Customizable Changes

A Panel Customizable component is a container for a group of Oracle ADF components, task flows, and portlets that you can make customizable at runtime. When you upgrade a WebCenter application, JDeveloper removes various attributes from Panel Customizable because they are no longer supported in Oracle WebCenter Portal 11*g*. Table 16–1 lists Panel Customizable attribute changes in an upgraded WebCenter application.

Table 16–1   Panel Customizable Attribute Changes in an Upgraded WebCenter Application

Attribute	Attribute Availability in an Upgraded WebCenter Application
DisplayHeader ExpansionMode Text Icon IsSeededInteractionAvailable IsMinimizable IsMaximizable IsMovable IsShowContentEnabled ContentInlineStyle DisclosureListener Background isEditable isHelpAvailable	Removed. All these attributes are removed from an upgraded WebCenter application. A header is no longer supported for Panel Customizable. Therefore, DisplayHeader and certain other attributes, such as ExpansionMode, Text, and so on, that are dependent on the header being available are also removed.
DisplayScrollBar	Removed. However, to render scrollbars for the Panel Customizable content area, you can set the Layout attribute to Scroll.
Width Height	Deprecated. The usage of the Width and Height attributes is discouraged. It is recommended that you use the inlineStyle attribute to specify the width or height of Panel Customizable. For example, you can set inlineStyle="width:300px;height:500px;".

Example 16–1 shows the sample Panel Customizable code in the source view of a WebCenter 10.1.3.x application.

Example 16–1   Panel Customizable Code in a WebCenter 10.1.3.x Application

```
<cust:panelCustomizable id="panelCustomizable1"
  text="panelCustomizable 1"
  displayHeader="true"
  isMaximizable="true"
  isMinimizable="true"
  width="100%"
  height="50px">
  displayScrollBar="auto"
  isShowContentEnabled="true"/>
```

Example 16–2 shows the sample Panel Customizable code after a WebCenter 10.1.3.x application has been upgraded to Oracle WebCenter Portal 11*g*. Notice that attributes such as isMinimizable and displayHeader are removed from the Panel Customizable code.

Example 16–2   Panel Customizable Code of a WebCenter Portal 11g Application

```
<cust:panelCustomizable id="panelCustomizable1"
xmlns="http://xmlns.oracle.com/adf/faces/customizable/html"/>
```

16.1.1.3 Show Detail Frame

A Show Detail Frame component renders a border or chrome around a child component in your WebCenter application pages. It provides a header with an Actions menu that contains User Interface (UI) controls to customize the display of the child component. Table 16–2 lists the changes that JDeveloper makes to the attributes of the Show Detail Frame component when you upgrade your WebCenter application.

Table 16–2   Changes Made to Show Detail Frame Attributes in Upgraded WebCenter Applications

Attribute	Description
`isMaximizable`	This attribute is removed from the upgraded application.
`isSeededInteractionAvailable`	`IsSeededInteractionAvailable=true` is migrated to the following: ■ `showMoveAction=menu` ■ `showMinimizeAction=chrome` ■ `showRemoveAction=chrome` `IsSeededInteractionAvailable=false` is migrated to the following: ■ `showMoveAction=none` ■ `showMinimizeAction=none` ■ `showRemoveAction=none`
`isMovable`	This attribute is renamed to `showMoveAction`. The upgrade utility migrates `isMovable=None|Menu` to `showMoveAction=none|menu`, respectively.
`isMinimizable`	This attribute is renamed to `showMinimizeAction`. The upgrade utility migrates the `isMinimizable=false|true` to `showMinimizeAction=none|chrome`, respectively.
`expansionMode`	This attribute is removed from the upgraded application if `expansionMode=maximized`. If the value was set to `normal` or `minimized`, then the attribute is retained.
`ContentInlineStyle`	This attribute is renamed to `InlineStyle`.

Example 16–3 shows the sample code of Show Detail Frame in the source view of a WebCenter 10.1.3.x application page. Example 16–4 shows the sample Show Detail Frame code after this application is upgraded to Oracle WebCenter Portal 11*g*. Notice that `isMaximizable` is removed and the `isSeededInteractionAvailable`, `isMinimizable`, `isMovable`, and `ContentInlineStyle` attributes are renamed.

Example 16–3   Show Detail Frame Code of a WebCenter 10.1.3.x Application

```
<cust:showDetailFrame id="showDetailFrame1"
  text="showDetailFrame 1"
  isSeededInteractionAvailable="false"
  isMaximizable="false"
  isMinimizable="true"
  isMovable="true">
  contentInlineStyle="background-color:rgb(0,255,255)"/>
```

Example 16–4   Show Detail Frame Code of a WebCenter Portal 11g Application

```
<cust:showDetailFrame id="showDetailFrame1"
  text="showDetailFrame 1"
  displayActions="never"
  showMoveAction="none"
  showMinimizeAction="none"
  contentStyle="background-color:rgb(0,255,255)"
  xmlns="http://xmlns.oracle.com/adf/faces/customizable/html"/>
```

16.1.2 External Application Changes

You can associate external applications with portlet producers. An external application is any application that implements its own authentication process. That is, an application that does not take part in the single sign-on process of Oracle WebCenter. For more information, see the "Working with External Applications" section in *Oracle Fusion Middleware Developer's Guide for Oracle WebCenter Portal*.

This section contains the following subsections:

- Overview of External Applications Changes in Oracle Application Server 10g and Oracle Fusion Middleware 11g
- External Application Upgrade Changes

16.1.2.1 Overview of External Applications Changes in Oracle Application Server 10*g* and Oracle Fusion Middleware 11*g*

Compared to Oracle Application Server 10*g*, in Oracle Fusion Middleware 11*g*, external applications have undergone some changes. In Oracle Fusion Middleware 11*g*, external applications let you specify shared and public credentials. External applications also let you define an internal name (that serves the purpose of external application identifier and cannot be changed after creation) and a display name (that can be changed). Another distinguishing feature is that in Oracle Application Server 10*g*, external applications use Metadata Services (MDS) to store metadata; in Oracle Fusion Middleware 11*g*, external applications are stored as connections similar to other services like content repository connections.

Table 16–3 compares how the distinguishing attributes of external applications map between Oracle Application Server 10.1.3.x and Oracle Fusion Middleware 11*g*. The table shows sample values for the `name`, `ExtAppId`, and `GUID` attributes.

Table 16–3   Distinguishing External Applications Attributes in Oracle Application Server 10.1.3.x and Oracle Fusion Middleware 11g

External Application Attribute	Sample Attribute Value in Oracle Application Server 10.1.3.x	Sample Attribute Value in Oracle Fusion Middleware 11*g*
Name	app1	app1 (Also used as the display name of the external application)
extAppId	/oracle/adf/externalApp s/extApp921389557420089 8560.xml	app1
SharedCredEnabled	Attribute not supported	False
PublicCredEnabled	Attribute not supported	False
loginEnabled	Attribute not supported	True

Table 16–3 (Cont.) Distinguishing External Applications Attributes in Oracle Application Server 10.1.3.x and Oracle Fusion Middleware 11g

External Application Attribute	Sample Attribute Value in Oracle Application Server 10.1.3.x	Sample Attribute Value in Oracle Fusion Middleware 11*g*
GUID	Attribute not supported	ExtAppbc638395-2860-41ae-a9f f-edfc6b79110f

16.1.2.2 External Application Upgrade Changes

If your WebCenter 10.1.3.x application relies on external applications, then when you upgrade that application, the associated external applications are also upgraded automatically. For example, when migrating a WebCenter 10.1.3.x application that uses PDK-Java producers dependent on external applications, the external applications are automatically upgraded. While upgrading a WebCenter application, the upgrade utility performs the tasks for an upgraded external application:

- Uses the name of the 10.1.3.x external application as the internal name (`ExtAppId`) of the upgraded application.

- Updates the producer metadata with the new external application ID. An external application is referenced in the portlet producer metadata through its ID.

- Migrates the external application metadata from MDS store configured in the WebCenter application's `adf-config.xml` to `connections.xml`.

- Adds the automated login configuration to the project that contains external application portlets.

- Adds the new attributes, listed in Table 16–3, to the upgraded external application.

- Removes the credential provisioning page configured for the project containing external application portlets and the managed bean and navigation rule defined in each project. In Oracle Fusion Middleware 11*g*, the credential provisioning page is packaged in the external application library.

- Configures external application libraries, WebCenter External Application Service, and WebCenter External Application Service View.

16.1.3 Portlet Component Changes

When you upgrade a WebCenter application that uses portlets, JDeveloper automatically makes various changes related to portlet tags in the upgraded application.

16.1.3.1 Portlet Tag Changes

In WebCenter 10.1.3.x applications, the `adfp:portlet` tag represents a portlet in the namespace `http://xmlns.oracle.com/adf/faces/portlet`.

In WebCenter Portal 11*g* applications, portlet tags are converted to Trinidad HTML Portlet tags. The `adfp:portlet` portlet tag is migrated to `adfph:portlet` in the namespace `http://xmlns.oracle.com/adf/faces/portlet/html`.

16.1.3.2 Attribute Changes

Table 16–4 lists the attributes that are renamed or removed from an upgraded WebCenter application containing portlets.

Table 16–4   Attribute Changes in an Upgraded WebCenter Application Containing Portlets

Attribute	Change in the Upgraded Application
`isSeededInteractionAvailable`	This attribute is renamed to `displayActions`. The upgrade utility migrates `isSeededInteractionAvailable=false|true` to `displayActions=onHover|always`, respectively.
`isMinimizable`	This attribute is renamed to `showMinimizeAction`. The upgrade utility migrates `isMinimizable=false|true` to `showMinimizeAction=none|chrome`, respectively.
`isMaximizable`	This attribute is renamed to `showMaximizeAction`. This change is applicable only for HTML portlets. The attribute is not available for rich-text portlets. The upgrade utility migrates `isMaximizable=false|true` to `showMaximizeAction=none|chrome`, respectively.
`isMovable`	This attribute is renamed to `showMoveAction`. The upgrade utility migrates `isMovable=False|True` to `showMoveAction=none|menu`, respectively.
`contentInlineStyle`	This attribute is renamed to `InlineStyle`.
`text`	This attribute is renamed to `title`.
`portletType`	This attribute is removed.
`isLinkModeAvailable`	This attribute is removed.
`showRemoveAction`	This attribute is removed from an HTML portlet. In Oracle WebCenter Portal 11*g*, this attribute is available only for rich-text portlets.
`expansionMode`	In JDeveloper 11*g*, the values allowed for this attribute are `normal` and `minimized`. In a WebCenter 10.1.3.x application, if the value is set to `maximized`, then the attribute is removed from the upgraded application.

16.2 Content Integration Connections: Migration of Credentials to External Applications

Content integration capabilities of Oracle WebCenter enable you to integrate decentralized content located across various content repositories, such as Oracle Content Server, Oracle Portal, or your file system, into your WebCenter applications. This section describes the authentication methods supported in Oracle WebCenter 10.1.3.x and Oracle WebCenter Portal 11*g* applications. It also describes how credentials are migrated to external applications for content repository connections when you upgrade a WebCenter 10.1.3.x application.

To integrate content from a content repository into your WebCenter application, you need to create a connection to the required repository. In Oracle WebCenter Portal 11*g*, while creating a content repository connection, you can set the authentication method to any of the following:

- **Identity Propagation**: In this method, no credentials are passed to the content repository. Instead, the repository connector uses the current user's identity as determined from the Java security context. This method is generally used when the application and the content repository use the same identity store to authenticate users.

- **External Application**: This method can be used in all other cases where the current user's identity should not be propagated directly to the content repository.

In Oracle WebCenter 10.1.3.x, the authentication method used for connecting to a content repository is determined through the setting specified for JAAS security. Select the **Use JAAS for security** checkbox if you want to transfer the user's identity to the repository without passing the credentials and to use the trust relationship between the WebCenter application and the content repository. If you do not use JAAS security, you need to specify credentials and leave the checkbox unselected.

Figure 16–1 shows the content repository connection details of a WebCenter 10.1.3.x application that does not use JAAS security. In Figure 16–1, the **Use JAAS for security** checkbox is not selected and credentials are specified in the **Username** and **Password** fields.

Figure 16–1   Authentication Method of a WebCenter 10.1.3.x Application
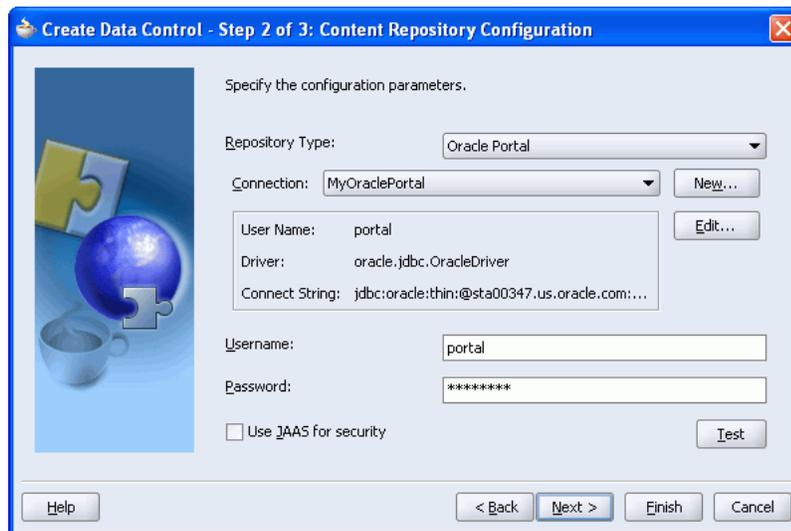
If you upgrade a WebCenter 10.1.3.x application in which JAAS security is used (that is, the **Use JAAS for security** checkbox is selected), then **Identity Propagation** is automatically set as the authentication method for the connection in the upgraded application. (The **Use JAAS for security** feature is same as **Identity Propagation**.)

If the **Use JAAS for security** checkbox was not selected in your WebCenter 10.1.3.x application, when you upgrade the application, the authentication method is set to **External Application**. Further, a new external application with public credentials is created automatically by using credentials from the 10.1.3.x content repository connection. Figure 16–2 shows the content repository connection details of an upgraded application. It shows the **External Application** authentication method selected and a new external application created for the upgraded application. Figure 16–3 shows the public credentials automatically specified for the newly created external application.

Figure 16–2   Content Repository Connection Configuration of an Upgraded Application
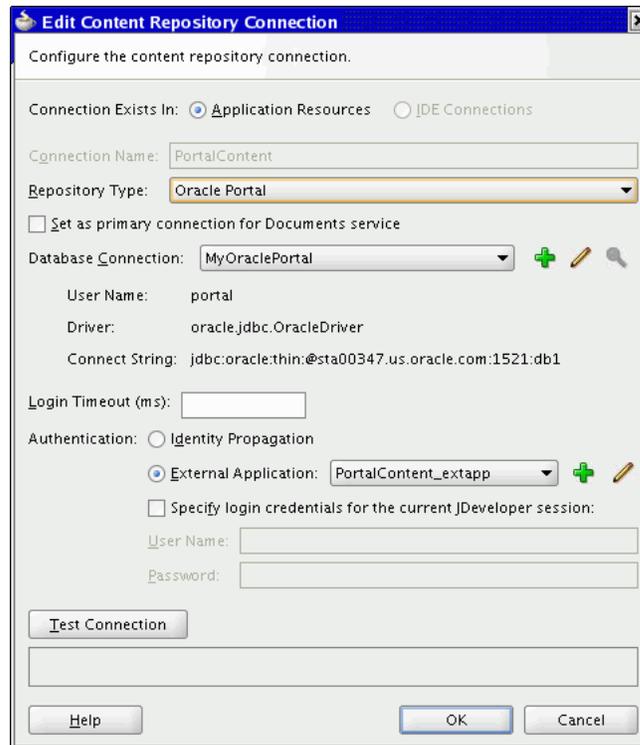
Figure 16–3   Public Credentials Specified for an External Application
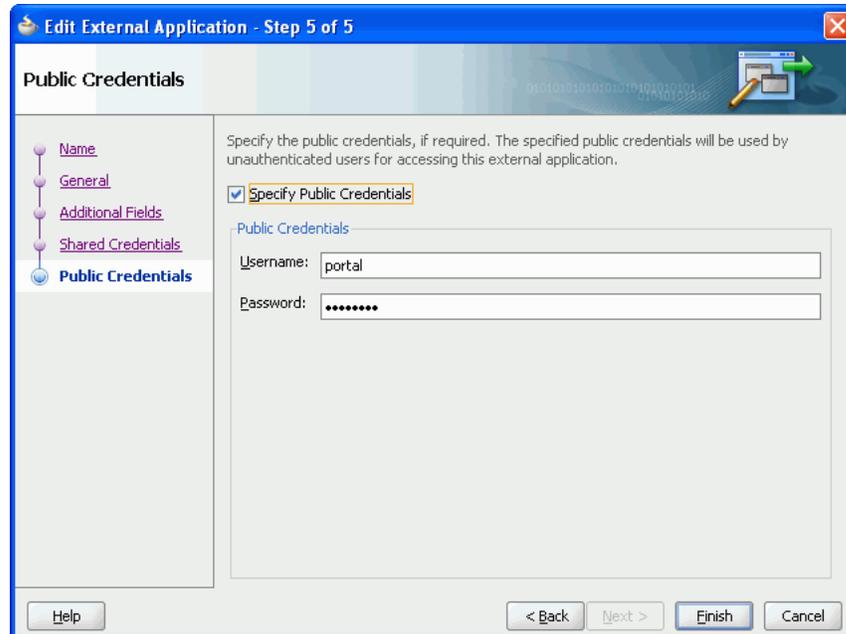
> **Note:** If your application contains a content repository connection that uses credentials, during application upgrade if there is a problem in retrieving credentials from the credential store or in creating the external application, you must edit the upgraded content repository connection. In the Edit Content Repository Connection wizard, you must select External Application as the authentication method, and then create an external application with the required credentials.

16.3 Preconfigured Portlet Producers: Customization Store's Location

By default, customizations related to preconfigured portlet producers are stored at the following location in Oracle WebCenter Portal 11*g*:

```
11g_jdev_install_dir/jdeveloper/portal/portletdata
```

You may choose to store portlet customizations at a different location. This section describes how to change the location of a customizations store for various preconfigured portlet producers. It contains the following subsections:

- WSRP Portlet Producers
- Web Clipping Portlet Producer
- OmniPortlet and other PDK-Java Portlet Producers
- OmniPortlet Repository Changes

16.3.1 WSRP Portlet Producers

To specify a file preference store location for portlet customizations of your WSRP portlet producers, update your producer's `WEB-INF/web.xml` with the following entry:

```
<env-entry>
    <env-entry-name>oracle/portal/wsrp/server/fileStoreRoot</env-entry-name>
    <env-entry-type>java.lang.String</env-entry-type>
    <env-entry-value>{YOUR_PORTLET_DATA_PATH}</env-entry-value>
</env-entry>
```

Where `YOUR_PORTLET_DATA_PATH` refers to the path where you want to store portlet customizations.

If you want to use a database preference store for your WSRP portlet producer, then you must set up the database preference store. For information, see the "Setting Up a Preference Store" section in the *Oracle Fusion Middleware Developer's Guide for Oracle WebCenter Portal*.

16.3.2 Web Clipping Portlet Producer

To specify the Web Clipping customization store, update the following entry in `mds-config.xml`:

```
<property name="metadata-path" value="{YOUR_PORTLET_DATA_PATH}"/>
```

For information about Web Clipping portlet configuration, see the "Web Clipping Portlet Configuration Tips" section in the *Oracle Fusion Middleware Developer's Guide for Oracle WebCenter Portal*.

16.3.3 OmniPortlet and other PDK-Java Portlet Producers

For OmniPortlet, you specify the customization store location in `provider.xml`. The file is available at the following location in the Fusion Middleware 11*g* installation:

```
MW_HOME/user_projects/domains/wc_domain/servers/WLS_Portlet/tmp/_WL_
user/portalTools_11.1.1.2.0/RANDOMLY_GENERATED_
DIRECTORY/war/WEB-INF/providers/omniPortlet/provider.xml
```

> **Tip:** To locate OmniPorlet's `provider.xml`, search for "omniPortlet/provider.xml" in the installation directory of Oracle WebLogic Server.

To specify a file preference store location for portlet customizations of your OmniPortlet portlet producers, update the following entry in `provider.xml`:

```
<preferenceStore class="oracle.webdb.reformlet.ReformletFilePreferenceStore">
    <name>omniPortletprefStore</name>
    <useHashing>true</useHashing>
    <rootDirectory>{YOUR_PORTLET_DATA_PATH}</env-entry-value></rootDirectory>
</preferenceStore>
```

You can change the file preference store location for other PDK-Java portlet producers in the same way.

If you use a database preference store for OmniPortlet and other PDK-Java portlet producer, then you must set up the database preference store. For more information, see the "Setting Up a Preference Store" section in the *Oracle Fusion Middleware Developer's Guide for Oracle WebCenter Portal*.

16.3.4 OmniPortlet Repository Changes

In Oracle WebCenter Portal 11*g*, OmniPortlet connection details are stored in the Oracle Platform Security Service (OPSS) credential store. In earlier versions of Oracle WebCenter, connection details for OmniPortlet are stored in the Web Clipping repository. After you upgrade an Oracle WebCenter application containing OmniPortlet, connection details for OmniPortlet are automatically migrated from the Web Clipping repository when you run the OmniPortlet for the first time after upgrade. The first run of OmniPortlet can be either accessing the OmniPortlet Test Page or displaying an OmniPortlet instance on an application page. The upgrade process is recorded in the log file of Oracle WebLogic Server. After application upgrade, in OmniPorlet's `provider.xml`, the `vaultId` tag is marked with the prefix `MIGRATED_FROM_VAULT_ID_old_vault_id`.

If there is any error during application upgrade, you may need to run the upgrade procedure again. For example, if application upgrade fails due to the Web Clipping repository being unavailable, you must ensure that the database is up and running and then upgrade your application again.

To re-run the upgrade procedure for an application that contains an OmniPortlet:

1. From the OmniPorlet's `provider.xml`, remove the `MIGRATED_FROM_VAULT_ID_old_vault_id` prefix.

2. Restart the OmniPortlet producer.

3. Access the OmniPortlet Test Page by using the following URL format:

   ```
   http://host:port/portalTools/omniPortlet/providers/omniPortlet
   ```

Where, *host* and *port* refer to the host name and port number of the system where you installed Oracle WebCenter.

16.4 Migration of JSR 168 Portlet Producers to JSR 286: Handling of Portlet Elements

In JDeveloper 11*g*, when you open for the first time an existing portlet producer application that contains JSR 168 portlets, the portlets are automatically upgraded to be JSR 286 compliant.

When you upgrade a 10*g* portlet producer application, the WebCenter migrator deletes `oracle-portlet.xml` and upgrades `portlet.xml`. It converts the navigation parameters stored in `oracle-portlet.xml` into public render parameters and adds them to `portlet.xml`. If `weblogic-portlet.xml` exists, it is retained as is.

The migration of a JSR 168 portlet producer application to JSR 286 is fully automated; it does not require you to perform any additional manual steps. However, based on the warnings that may be displayed in the log panel during migration, you may want to make additional changes to portlets. The following tables describe how various parameters in the `oracle-portlet.xml` file of a JSR 168 portlet producer application are handled in `portlet.xml` of a JSR 286 portlet producer application:

- Table 16–5: Describes handling of `portlet-app-extension` elements
- Table 16–6: Describes handling of `portlet-extension` elements
- Table 16–7: Describes handling of `navigation-parameters` elements

Table 16–5   Handling of portlet-app-extension Elements of oracle-portlet.xml during Portlet Producer Upgrade

`portlet-app-extension` **Element**	**XML Type**	**Handling of the element in** `portlet.xml` **of the JSR286-Compliant Upgraded Application**
`<export-size-limit>`	int	Ignored
`<allow-export>`	boolean	Converted to a portlet application-level `com.oracle.portlet.allowWsrpExport` Container Runtime Option (CRO) with the same boolean value.
`<allow-import>`	boolean	Ignored
`<allow-copy-portlets>`	boolean	Ignored
`<export-encryption-data>`	complexType	Ignored
`<use-jazn-security>`	complexType	Ignored

Table 16–5 (Cont.) Handling of portlet-app-extension Elements of oracle-portlet.xml during Portlet Producer Upgrade

portlet-app-exten sion **Element**	**XML Type**	**Handling of the element in** `portlet.xml` **of the JSR286-Compliant Upgraded Application**
`<strict-authentic ation>`	boolean	Converted to the following portlet application-level CRO with the opposite boolean value: `com.oracle.portlet.useWsrpUserCont extForUserAuthenticationInfo` If this element does not exist in `oracle-portlet.xml` but `oracle-portlet.xml` does exist and the `portlet.xml` schema version is 1.0, the following portlet application-level CRO is created with the value `true`: `com.oracle.portlet.useWsrpUserCont extForUserAuthenticationInfo`

Table 16–6 Handling of the portlet-extension Elements of oracle-portlet.xml

portlet-extension **Element**	**XML Type**	**Handling of the element in** `portlet.xml` **of the JSR286-Compliant Upgraded Application**
`<portlet-name>`	string	Used for correlating the `portlet-extension` entry in `oracle-portlet.xml` with the appropriate portlet entry in `portlet.xml`
`<published-events >`	complexType	Ignored
`<handled-events>`	complexType	Ignored
`<navigation-param eters>`	complexType	Each `navigation-parameters` element is converted into a JSR286 public render parameter
`<portlet-id>`	string	Converted to a portlet-level `com.oracle.portlet.wsrpPortletHandle` CRO with the value "`E:i`" + portlet-id value + "`:default`". If the parameter is not present, a portlet-level `com.oracle.portlet.wsrpPortletHandle` container runtime option is created with the value "`E:n`" + portlet-name (from `portlet.xml`) + "`:default`"
`<allow-export>`	boolean	Ignored
`<allow-import>`	boolean	Ignored
`<hide-portlet>`	boolean	Converted to a portlet-level `com.oracle.portlet.offerPortletOverWs rp` CRO with an opposite boolean value.
`<require-iframe>`	boolean	Converted to a portlet-level `com.oracle.portlet.requireIFrame` CRO with the same boolean value.
`<minimum-wsrp-ver sion>`	int	Converted to a portlet-level `com.oracle.portlet.minimumWsrpVersion` CRO with the same integer value.

Table 16–7 describes the handling of navigation parameters of `oracle-portlet.xml` in the `portlet.xml` file when you upgrade a portlet producer application. When a JSR 168 portlet producer application is upgraded, the following changes are made while converting `oracle-portlet.xml`'s navigation parameters into public render parameters:

- If the portlet has any declared `supported-public-render-parameter` in `portlet.xml`, all navigation parameters from `oracle-portlet.xml` are ignored and a warning message is logged.

- If portlet application-level `public-render-parameter` descriptions in `portlet.xml` have the same identifier as the local part of the navigation parameters' name in `oracle-portlet.xml`, the navigation parameters that conflict are ignored and a warning message is logged.

- If any navigation parameters specified in any `portlet-extension` element in `oracle-portlet.xml` have different namespace parts but the same local part of the `<name>` element, all overlapping navigation parameters except one are ignored and a warning message is logged.

Table 16–7   Handling of the navigation-parameters Element available in oracle-portlet.xml

`navigation-parameter` **Element**	**XML Type**	**Handling in** `portlet.xml` **of a JSR286 Container**
`<name>`	QName	The full QName is used as the `public-render-parameter` QName in `portlet.xml` at the portlet application-level. The local part of the QName is used as the `public-render-parameter` identifier at the portlet application-level, and as the `supported-public-render-parameter` value at the portlet level.
`<type>`	QName	Ignored
`<label>`	localizedString	Converted to the description element. The `<label>`, `<hint>`, and `<usage>` elements are merged in the given sequence to create the description element of the public-render-parameter in `portlet.xml`.
`<hint>`	localizedString	Converted to the description element. The `<label>`, `<hint>`, `<usage>` elements are merged in the given sequence to create the description element of the public-render-parameter in `portlet.xml`.
`<usage>`	string	Converted to the description element. The `<label>`, `<hint>`, `<usage>` elements are merged in the given sequence to create the description element of the public-render-parameter in `portlet.xml`.
`<aliases>`	QName	Converted to alias declarations on the `public-render-parameter` declaration in `portlet.xml` at the portlet application level.

A

Example: Upgrading and Redeploying SOA Order Booking

This appendix provides the step-by-step procedure for upgrading a complex Oracle SOA Suite (SOA) application from Oracle Application Server 10*g* Release 3 (10.1.3.3) to Oracle Fusion Middleware Release 11 (11.1.1.0.0).

This procedure is meant to serve as an overall example of the Oracle SOA Suite environment and application upgrade process.

Specifically, this appendix contains the following topics:

- About the SOA Order Booking Application
- Task 1: Verify That the SOA Order Booking and Its External Partners are Up and Running
- Task 2: Install and Upgrade the Oracle Fusion Middleware SOA Suite
- Task 3: Use the Oracle WebLogic Suite Administration Console to Configure Required Resources
- Task 4: Install Oracle JDeveloper 11g
- Task 5: Set Up Connections in Oracle JDeveloper Release 11
- Task 6: Open the SOA Order Booking Application in Oracle JDeveloper 11g and Migrate the Application Projects
- Task 7: Compile and Deploy Each of the Application Projects on Oracle Fusion Middleware 11g
- Task 8: Test the SOA Order Booking Application in Fusion Middleware Control

A.1 About the SOA Order Booking Application

Oracle provided the SOA Order Booking application with Oracle Application Server 10g Release 3 (10.1.3.1.0) as a sample application that demonstrates many of the key features and components of the Oracle Application Server 10g Release 3 (10.1.3.1.0) release.

The SOA Order Booking application is described in detail in the *Oracle SOA Tutorial* and in the Oracle SOA Suite Quick Start Guide, which are both available in the Oracle Application Server 10*g* Release 3 (10.1.3.1.0) Documentation Library on the Oracle Technology Network:

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

The Oracle Application Server 10*g* Release 3 (10.1.3.4) patch set introduced a set of improvements and fixes to common problems. In addition, Oracle released a new version of the SOA Order Booking that was tested and improved for 10*g* Release 3 (10.1.3.4).

This appendix assumes you are familiar with the SOA Order Booking application and the technologies it demonstrates.

Figure A–1 shows the login page of the SOA Order Booking Demo, deployed on Oracle Application Server 10*g* Release 3 (10.1.3.4).

Figure A–1   SOA Order Booking Application Deployed on 10g Release 3 (10.1.3.4)

A.2 Task 1: Verify That the SOA Order Booking and Its External Partners are Up and Running

Before you can migrate the SOA Order Booking application to 11*g*, you must review the following prerequisites:

- Task 1a: Verify That the SOA Order Booking Is Successfully Deployed on Oracle Application Server 10g Release 3 (10.1.3.4)

- Task 1b: Verify That SOA Order Booking Is Currently Running on Oracle Application Server 10g Release 3 (10.1.3.4)

- Task 1c: Verify That All External Partners Are Available

A.2.1 Task 1a: Verify That the SOA Order Booking Is Successfully Deployed on Oracle Application Server 10*g* Release 3 (10.1.3.4)

Before you can migrate the SOA Order Booking application to 11*g*, it is assumed that:

- You have already installed and configured Oracle Application Server 10*g* Release 3 (10.1.3.1.0), and you have applied the 10*g* Release 3 (10.1.3.4) patch set.

- You have already installed and configured Oracle JDeveloper 10*g* Release 3 (10.1.3.4).

- You have downloaded the updated 10*g* Release 3 (10.1.3.4) version of the SOA Order Booking demo application.

  You can download the 10*g* Release 3 (10.1.3.4) SOA Order Booking demo application from the following location on the Oracle Technology Network (OTN):

  `http://www.oracle.com/technology/products/ias/bpel/`

- You have used the instructions in the 10*g* Release 3 (10.1.3.1.0) *Oracle SOA Suite Tutorial* or *Oracle SOA Suite Quick Start Guide* to deploy the SOA Order Booking application on Oracle Application Server 10*g* Release 3 (10.1.3.4).

  This includes installing the database schemas for the SOA Order Booking 10*g* Release 3 (10.1.3.4) application in an available database.

A.2.2 Task 1b: Verify That SOA Order Booking Is Currently Running on Oracle Application Server 10*g* Release 3 (10.1.3.4)

Before you migrate the SOA Order Booking application to 11*g*, test the application on 10*g* Release 3 (10.1.3.4) and make sure that it is functioning properly. This ensures that the database where the SOA Order Booking schemas are installed is up and running and the external services and external partners are up and running.

For example, you should be able to log in to the application, browse the list of products (Figure A–2), and place an order. You should also be able to monitor the BPEL processes using the BPEL Worklist Manager.

Figure A–2 Browsing Items in the SOA Order Booking Application

For more information, see "Running and Monitoring the SOA Order Booking Application" in the Oracle Application Server 10*g* Release 3 (10.1.3) *Oracle SOA Suite Quick Start Guide*.

The *Oracle SOA Suite Quick Start Guide* is included in the Oracle Application Server 10*g* Release 3 (10.1.3) documentation library, which is available on the Oracle Technology Network (OTN):

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

A.2.3 Task 1c: Verify That All External Partners Are Available

Before you can upgrade the SOA Order Booking application to 11*g*, you must ensure that the external partners (in the form remote Web services) are up and running and available.

Oracle JDeveloper 11*g* must be able to access these external partners, as well as the Oracle BPEL Process Manager and Oracle Enterprise Service Bus services deployed on 10g; otherwise, it can not properly migrate the projects in your application to 11*g*.

To verify that the required Web services are available, you should review each of the projects in your Oracle SOA Suite applications before you migrate them to 11*g*. For example, you can review the Oracle BPEL Process Manager configuration files for each project and verify that the Web services URIs referenced in those configuration files are valid and working.

For example, in the SOA Order Booking application, you can review the contents of the SOAOrderBooking project, as follows:

1. Open the SOA Order Booking 10*g* Release 3 (10.1.3.4) application in Oracle JDeveloper 10*g* Release 3 (10.1.3).

2. Expand the **SOAOrderBooking** project, and then expand the **Integration Content** folder.

3. Expand the **bpel** folder and double-click the `bpel.xml` file (Figure A–3).

Figure A–3   Selecting a bpel.xml File to Test WSDL URLs
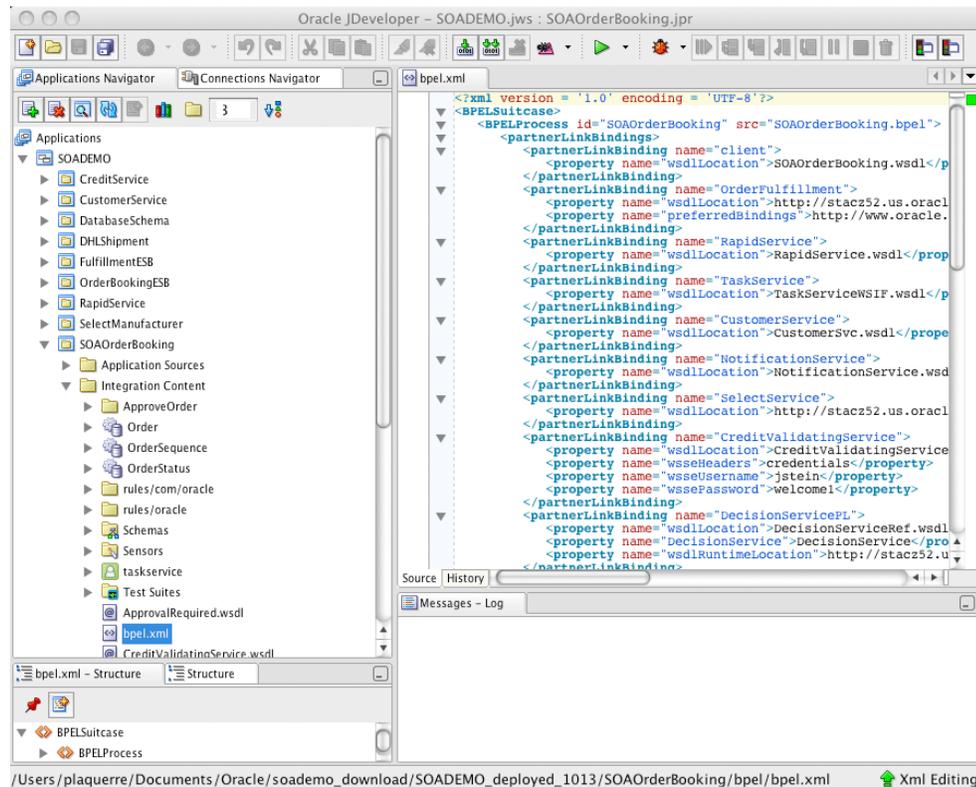
4. Inside the bpel.xml file, search for the following entry:

```
<partnerLinkBinding name="OrderFulfillment">
        <property  name="wsdlLocation">
```

```
http://host.domain:port/esb/wsil/Fulfillment/OrderFulfillment?wsdl
        </property>
        <property name="preferredBindings">
                http://www.oracle.com/esb/
        </property>
</partnerLinkBinding>
```

5. Copy the URL in the `wsdlLocation` element to your browser and verify that the URL is valid and displays the output shown in Figure A–4.

6. If the Web Services URL do not display correctly, troubleshoot and solve the problem before attempting to migrate the application to 11*g*.

For example, make sure the database where the SOA Order Booking schemas are installed is up and running and that all components of the application have been deployed successfully.

7. Repeat this procedure for the other URLs in the `bpel.xml` file.

8. Perform a similar test for the WSDL URLs in the following projects:

- SOAOrderBooking:

  ```
  CreditValidatingService.wsdl
  CustomerSvc.wsdl
  RapidService.wsdl
  ```

- OrderBookingESB:

  ```
  OrderBooking_OrderBookingProcess.esbsvc
  OrderBooking_OrderBookingService.esbsvc
  ```

Figure A–4   OrderFulfillment WSDL URL in Internet Explorer
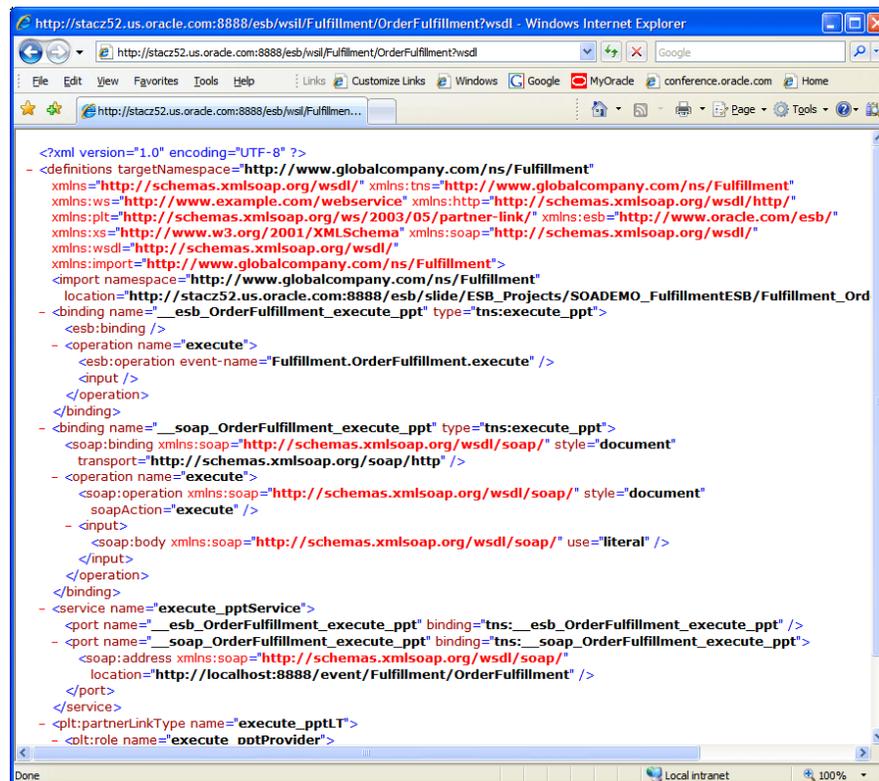
A.3 Task 2: Install and Upgrade the Oracle Fusion Middleware SOA Suite

After you verify that the SOA Order Booking application is running successfully on your Oracle Application Server 10*g* Release 3 (10.1.3) environment, you can then install a new Oracle Fusion Middleware 11*g* environment and upgrade the environment.

Follow the instructions in Chapter 6, "Upgrading an Oracle SOA Suite 10g Environment" to prepare the Oracle Fusion Middleware environment for the SOA Order Booking migration and redeployment.

This task includes:

- Installing and configuring the Oracle SOA Suite 11*g* middle tiers
- Upgrading the Oracle SOA Suite schemas to 11*g*

A.4 Task 3: Use the Oracle WebLogic Suite Administration Console to Configure Required Resources

The SOA Order Booking demo application requires that a data source and a set of JMS resources be available on the Oracle WebLogic Server managed server where SOA Order Booking is deployed.

Refer to the following sections for information on how to use the Oracle WebLogic Server Administration Console to create these resources:

- Task 3a: Create the SOADEMO Data Source
- Task 3b: Configure the DbAdapter Application
- Task 3c: Create the SOA Order Booking JMS Resources

A.4.1 Task 3a: Create the SOADEMO Data Source

Before you can deploy the SOA Order Booking sample application on Oracle WebLogic Server, you must first define an Oracle WebLogic Server data source that references the database where the SOA Order Booking schemas are installed.

These are the same schemas you installed previously, when you deployed the SOA Order Booking demo application on Oracle Application Server 10*g* Release 3 (10.1.3.4), as described in Section A.2, "Task 1: Verify That the SOA Order Booking and Its External Partners are Up and Running".

To create the data source:

1. Log in the Administration Console.

2. In the Domain Structure tree, expand **Services > JDBC,** then select **Data Sources**.

3. On the Summary of Data Sources page, click **New**.

4. Oracle WebLogic Server displays the JDBC Data Source Properties page.

    Enter the following properties on this page:

    - Name: `soademoDS`
    - JNDI Name: `jdbc/soademoDS`
    - Database Type: **Oracle**
    - Database Driver: **Oracle's Driver (Thin XA)**

5. Click **Next** to display the Transaction Options page:

6. Click **Next** to display the Connection Details page ().

   ■ **Database Name**: *Oracle_database_system_identifer_(SID)*

   ■ **Host Name**: *Host_where_the_database_is_installed*

   ■ **Port**: *Database_listening_port_for_example_1521*

   ■ **Database User Name**: soademo

   ■ **Password and Confirm Password**: soademo

7. Click **Next** to test the database connection.

8. Click **Next** to display the Select Targets page, and select the **soa_server1**.

9. Click **Finish** to deploy the data source on soa_server1.

A.4.2 Task 3b: Configure the DbAdapter Application

To connect to the database, the SOA Order Booking application requires some changes to the default DbAdapter application that is installed and configured as part of a Oracle SOA Suite 11*g* domain.

Refer to the following sections for instructions on how to configure the DbAdapter for the SOA Order Booking application:

■ Creating a New Outbound Connection Pool for the DbAdapter Application

■ Configuring the New Connection Pool for the DbAdapter Application

A.4.2.1 Creating a New Outbound Connection Pool for the DbAdapter Application

Use the following procedure to create a new outbound connection pool in the DbAdapter application:

1. Create a directory where the deployment plan for the DbAdapter application can be saved.

   For example:

   MW_HOME/Oracle_SOA1/soa/DBPlan

2. In the Domain Structure tree, click **Deployments**.

3. Click the DbAdapter application (click the name, not the check box).

4. Click the **Configuration** tab, and then click the **Outbound Connection Pools** tab.

5. Click **New**.

6. Select the radio button for javax.resource.cci.ConnectionFactory and click **Next**.

7. Enter the JNDI Name as follows: eis/DB/soademoDS

   Note that this JNDI name must match the value you enter in your database connection, which you create when upgrading your application with Oracle JDeveloper later in this procedure.

8. Click **Finish**.

   At this point, you should be asked for the deployment plan location.

9. Click on the path to the directory you created earlier and enter the deployment plan name Plan.xml.

10. Verify that the correct deployment plan is selected.

11. Click **Next** and then click **Finish**.

A.4.2.2 Configuring the New Connection Pool for the DbAdapter Application

Use the following procedure to configure the new connection pool so it references the data source:

1. Click the **Configuration** tab, expand the connection factory, and click your new connection pool (click on the name, not the check box).

2. In the Properties table, select the box to the right of **xADataSource**.

   The edit box appears.

3. Enter the name of the data source name that you created previously:

   ```
   jdbc/soademoDS
   ```

4. Press ENTER to apply the value and click **Save**.

   You must use the ENTER key for the value to be entered in the field, and you must click **Save** to save your changes.

5. Return to the Deployments page and click **Deployments** in the left navigation bar.

6. Select the check box next to **DbAdapter**.

7. Click **Update**.

8. Select **Redeploy this application** and confirm the deployment plan location, and click **Finish**.

9. Verify your changes to ensure that the configuration is correct; otherwise, deployment errors will occur later:

   - Confirm that the connection pool is added, by returning to the **DbAdapter** > **Configuration** > **Outbound Connection Pools** page and expanding the connection factory.

   - Confirm the value of the **xADataSource** property that you entered previously.

   - Return to the DbAdapter Outbound Connection Pools tab, and review the connection factory property value for the `eis/DB/soademoDS` connection pool.

A.4.3 Task 3c: Create the SOA Order Booking JMS Resources

The SOA Order Booking sample application uses message queues to transmit order and fulfillment details. As a result, you must configure a JMS queue before you can deploy the application successfully.

You can use WLST commands and scripts to configure JMS, but these instructions show how to use the Oracle WebLogic Administration Console to configure JMS.

The following sections describe how to configure JMS for the SOA Order Booking Demo sample application:

- Creating the JMS Queue for the SOA Order Booking Demo

- Creating a JMS Connection Factory for the SOA Order Booking Demo

- Creating an Outbound Connection Pool for the JmsAdapter Application

- Configuring the JmsAdapter Connection Pool

- Redeploying the Modified JmsAdapter Application

A.4.3.1 Creating the JMS Queue for the SOA Order Booking Demo

To create the JMS destinations required for the SOA Order Booking Demo:

1. In the Oracle WebLogic Administration Console, navigate to the Summary of JMS Modules page and click the **SOAJMSModule** JMS module.

    This module is created and configured during the Oracle SOA Suite 11*g* installation and configuration.

2. On the JMS Module Configuration page, click **New.**

3. From the list of resource types, select **Queue**.

4. On the JMS Destination Properties page, enter a name and JNDI location for the the destination queue.

    For the purposes of this procedure, enter the following values:

    Name: `demoFulfillmentQueue`

    JNDI Name: `jms/demoFulfillmentQueue`

5. Click **Next**.

6. Select **SOASubDeployment** from the **SubDeployments** drop-down menu.

7. Select the **SOAJMSServer** JMS Server and click **Finish**.

A.4.3.2 Creating a JMS Connection Factory for the SOA Order Booking Demo

To create the connection factories required by the SOA Order Booking Demo sample application:

1. In the Oracle WebLogic Administration Console, navigate to the Summary of JMS Modules page and click the **SOAJMSModule** JMS module.

2. On the JMS Module Configuration page, click **New.**

3. From the list of resource types, select **Connection Factory** and click **Next**.

4. On the Connection Factory Properties page, enter a name and a JNDI location for the JMS connection factory.

    Use the following values:

    Name: `demoCF`

    JNDI Name: `jms/demoCF`

5. Click **Next**.

6. Click **Finish**.

    You should see both the queue and the connection factory listed in the Summary of Resources.

A.4.3.3 Creating an Outbound Connection Pool for the JmsAdapter Application

Use the following procedure to create a new connection pool. The connection pool is configured in the JmsAdapter application and uses a Deployment Plan:

1. Create a directory for the JmsAdapter application deployment plan.

    For example:

    `MW_HOME/Oracle_SOA1/soa/JMSPlan`

2. In the Oracle WebLogic Server Administration Console, click **Deployments**.

3. Click **JmsAdapter** (click the name, not the check box).

4. Click the **Configuration** tab and then the **Outbound Connection Pools** tab

5. Click **New**.

6. Select the factory `oracle.tip.adapter.jms.IJmsConnectionFactory`.

7. Click **Next**.

8. Enter `eis/demo/Queue`.

9. Click **Finish**.

At this point, you should be asked for the deployment plan location.

10. Click the path to the directory you created earlier and enter `Plan.xml` for the plan name.

If there is a plan file selected with the radio button at the bottom of the screen, the one you enter will take precedence.

11. Click **Next** and then click **Finish**.

A.4.3.4 Configuring the JmsAdapter Connection Pool

Use the following procedure to configure the connection pool you created in the previous section:

1. In the Oracle WebLogic Server Administration Console, click **Deployments**.

2. Click **JmsAdapter** (click the name, not the check box).

3. Click the **Configuration** tab and then the **Outbound Connection Pools** tab

4. Expand the **`oracle.tip.adapter.jms.IJmsConnectionFactory`** connection factory.

5. Click **`eis/demo/Queue`**.

6. Change the value of the **ConnectionFactoryLocation** property to `jms/demoCF` and press ENTER.

You must use the ENTER key for the value to be entered in the field.

7. Click **Save**.

You must use the Save button to save the value.

A.4.3.5 Redeploying the Modified JmsAdapter Application

Use the following procedure to redeploy the JmsAdapter application:

1. Click **Deployments** in the Oracle WebLogic Server Administration Console navigation tree.

2. Select the check box next to **JmsAdapter**.

3. Click **Update** at the top of the Deployment table.

4. Verify that the correct deployment plan is selected.

5. Select **Redeploy this application** and confirm the deployment plan location, and click **Finish**.

A.5 Task 4: Install Oracle JDeveloper 11*g*

You can install Oracle JDeveloper 11*g* from the ZIP file on the Oracle Fusion Middleware Companion CD or by downloading the Oracle JDeveloper Studio Edition Version 11.1.1.1.0.0 ZIP file from the Oracle Technology Network (OTN):

```
http://www.oracle.com/technology/software/products/jdev/index.html
```

You can install JDeveloper on the same machine that is running Oracle Fusion Middleware 11*g*, or you can install it on a different machine from the one where Oracle Fusion Middleware is running.

For complete information, see *Oracle Fusion Middleware Installation Guide for Oracle JDeveloper*.

A.6 Task 5: Set Up Connections in Oracle JDeveloper Release 11

In Oracle JDeveloper Release 11 (11.1.1.0.0), you set up connections to the database, to Oracle Application Server. These connections enable you to view the data in the SOADEMO schema, and to deploy the applications to Oracle Fusion Middleware from JDeveloper.

This task includes the following subtasks:

- Task 5a: Start Oracle JDeveloper 11g
- Task 5b: Verify That You Have the Required Oracle JDeveloper Extensions
- Task 5c: Create a Connection to the SOADEMO Schema
- Task 5d: Create a Connection to Oracle WebLogic Server

A.6.1 Task 5a: Start Oracle JDeveloper 11*g*

To start Oracle JDeveloper Studio on Windows, use any of the following methods:

- From your **Start** Menu, select **All Programs**, then select **Oracle Fusion Middleware 11.1.1.x.x**, and then select **Oracle JDeveloper Studio 11.1.1.x.x**.

- You can also start Oracle JDeveloper from the command line by running one of the following commands:

```
install_dir\jdeveloper\jdeveloper.exe
install_dir\jdeveloper\jdev\bin\jdevw.exe
install_dir\jdeveloper\jdev\bin\jdev.exe (to display a console window
                                     for internal diagnostic information)
```

To start Oracle JDeveloper on UNIX systems, run the file:

```
 install_dir/jdeveloper/jdev/bin/jdev
```

A.6.2 Task 5b: Verify That You Have the Required Oracle JDeveloper Extensions

To upgrade your Oracle SOA Suite applications to 11*g*, you must have the Oracle SOA Composite Editor extension for Oracle JDeveloper 11*g*.

To verify that the extension is installed, select **About** from the Oracle JDeveloper **Help** menu, and click the **Version** tab. You should see an entry in the list of components called `SOA Composite Editor`.

If this component does not appear on the Version tab of the About dialog box, close the About dialog and select **Check for Updates** from the **Help** menu. Use the Check for

Updates wizard to locate and install the latest version of the SOA Composite Editor extension.

A.6.3 Task 5c: Create a Connection to the SOADEMO Schema

In JDeveloper, create a database connection to the SOADEMO schema:

> **Note:** In this tutorial, the database connection is named **SOADEMO**. You can use a different name if you want, but using the same naming conventions will make it easier to follow the instructions.

1. In Oracle JDeveloper, select **View > Database Navigator**.

2. Right-click the **IDE Connections** node and select **New Connection**.

3. In the resulting dialog box, enter the following values:

   - **Connection Name**: enter **SOADEMO**.

   - **Connection Type**: select **Oracle (JDBC)**.

   - **Username**: enter **SOADEMO**.

   - **Password**: enter **soademo**.

   - **Role**: leave blank.

   - **Save Password:** select the check box.

   - **Enter Custom JDBC URL:** clear the check box.

   - **Driver**: select **thin**.

   - **Host Name**: enter the name of the machine running the database where you installed the SOADEMO schema.

   - **JDBC Port**: enter the port number for the database. The default value is `1521`.

   - **SID**: enter the system identifier for the database. The default value is `ORCL`.

   If you are unsure about the database connection values, check with your database administrator.

4. Click **Test Connection**.

   If the test is not successful, check that the database is available and that the connection values are correct.

5. Click **Finish**. The connection appears below the **Database** node in the Connection Navigator.

6. You can now examine the schema from Oracle JDeveloper. In the Connection Navigator, expand **Database > SOADEMO**. Expand the **Sequences** and **Tables** nodes and verify that the tables match those listed in Section 1.4, "Contents of the SOADEMO Schema" of the 10*g* Release 3 (10.1.3.1.0) *Oracle SOA Tutorial*.
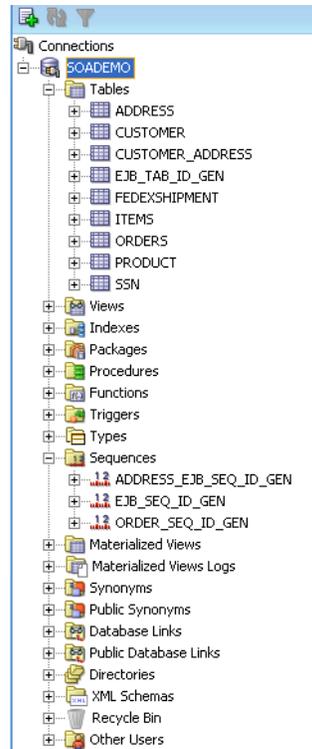
A.6.4 Task 5d: Create a Connection to Oracle WebLogic Server

In JDeveloper, create a connection to the Oracle WebLogic Server domain where you installed and configured Oracle SOA Suite 11*g*:

1. In Oracle JDeveloper, select **View > Application Server Navigator**.

2. Right-click the **Application Servers** folder under the Application Server node and select **New Application Server**.
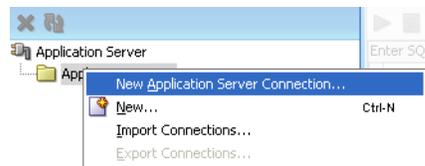
3. Click **Next** on the Welcome page of the Application Server Connection Wizard.

4. Enter the following values:

    ■ **Connection Name**: enter `SoademoApplicationServer`.

    ■ **Connection Type**: select **WebLogic 10.3**.

    Click **Next**.

5. In Step 2, Authentication, enter the following values:

    ■ **Username**: Enter **weblogic**. This is the name of the default Oracle WebLogic Server administration user.

    ■ **Password**: Enter the password for the `weblogic` user. This is the password that you set when you configured the Oracle SOA Suite 11*g* domain.

6. Click **Next** and enter the host, port, and name of the Oracle WebLogic Server domain you installed and configured in Section A.3, "Task 2: Install and Upgrade the Oracle Fusion Middleware SOA Suite".

7. Click **Next**. The Test Connection page appears.

8. Click **Test Connection**.

9. Click **Finish**.

   The connection appears below the **Application Servers** node in the Connection Navigator.

A.7 Task 6: Open the SOA Order Booking Application in Oracle JDeveloper 11*g* and Migrate the Application Projects

The following sections describe how you can open an application created in a previous version of Oracle JDeveloper and use the Migration Wizard to upgrade the application to Oracle Fusion Middleware 11*g*:

- Opening and Migrating the Application

- Viewing the Migration Log Files

> **Important:** The SOA Order Booking application and its projects rely upon external partners. These remote Web services must be up and running and available to the SOA Order Booking application before you migrate the application to Oracle Fusion Middleware 11*g*.
>
> You cannot successfully open the application in Oracle JDeveloper 11*g* if the external partner URI's are not available. For more information, see Task 1: Verify That the SOA Order Booking and Its External Partners are Up and Running.

A.7.1 Opening and Migrating the Application

To open and migrate the application to Oracle JDeveloper 11*g*:

1. Make a backup copy of the soademo folder that contains the 10*g* Release 3 (10.1.3.4) SOA Order Booking Demo application source files.

   The contents of the soademo folder will be modified during the upgrade. The backup copy allows you to restore the original 10*g* Release 3 (10.1.3) version of the demo if necessary.

   Note that for this procedure, the working folder (the folder containing the files you will upgrade) is referred to as the SOADEMO_HOME directory.

2. Review the contents of the SOADEMO_HOME directory and remove the following file if it exists:

   SOADEMO_HOME/SOAOrderBooking/bpel/rules/com

   This includes the subdirectories under the `bpel/rules/com/` directory.

3. Select **Open** from the File menu and select the Oracle JDeveloper 10g Release 3 (10.1.3.1.0) `soademo.jws` file.

   Oracle JDeveloper displays Step 1, the Welcome page, of the Migration Wizard.

   Click **Next**.

4. In Step 2 (Figure A–5), select **Yes** to migrate the project files to Oracle JDeveloper Release 11 (11.1.1.0.0).

Click **Next**.

Figure A–5   Step 2 of the Oracle JDeveloper 11g Migration Wizard
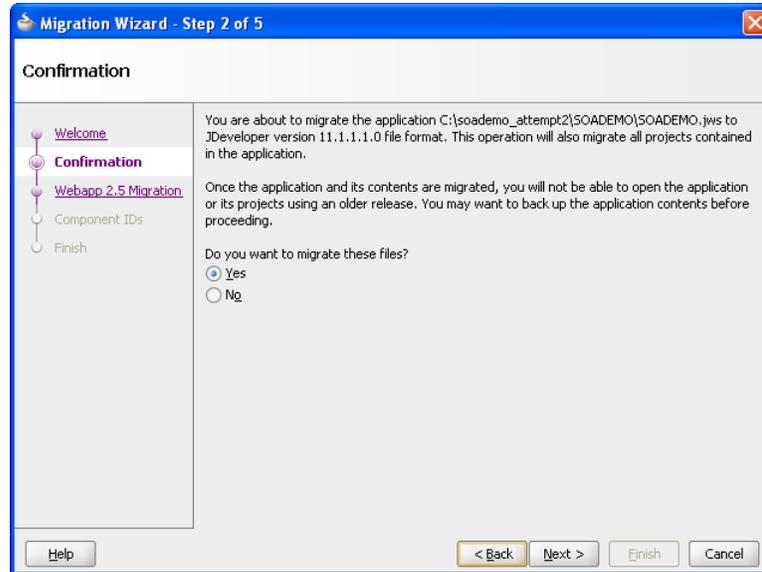
5. In Step 3, clear the **Migrate to Webapp 2.5** check box.

> **Note:**   Clearing the **Migrate to WebApp 2.5** check box is important because the SOA Order Booking demo application uses Web services that you will upgrade manually later in this procedure. To be sure the Migration wizard does not attempt to upgrade these automatically, be sure to clear the **Migrate to WebApp 2.5** check box.

Click **Next**.

Figure A–6   Clearing the WebApp 2.5 Option in the Oracle JDeveloper Migration Wizard
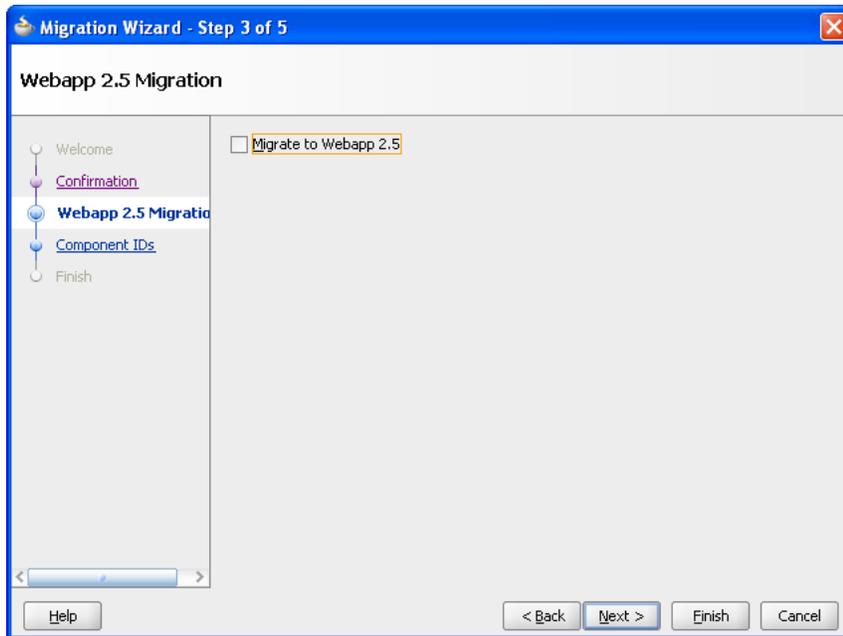
6. In Step 4, review the items on the page, accept the default selections, and click **Next**.

7. In Step 5, review the setup options you selected, and click **Finish** to upgrade the application to Oracle JDeveloper 11*g*.

   JDeveloper displays a Migration Status dialog box that indicates the progress of the migration. In addition, information about the migration is displayed in the logging pane in the main Oracle JDeveloper window.

8. When the migration is complete, click **Finish**.

A.7.2 Viewing the Migration Log Files

After the migration, you can review the log files generated during the application migration. The log files describe the various changes that Oracle JDeveloper 11*g* makes to the projects as they are opened and migrated to Oracle JDeveloper 11*g*.

The log files are located in the following directory:

JDEV_HOME\upgrade\logs\

You an identify the different log files for each application by viewing the date and time stamp in each log file name. For example:

SOAOrderBooking2009-04-29-12-46-35PM.log

In addition, the Messages tab in the Oracle JDeveloper main window displays links to the individual log files for each project that was migrated. Click any of the links to display a migration log file.

A.8 Task 7: Compile and Deploy Each of the Application Projects on Oracle Fusion Middleware 11*g*

After you have opened the SOADEMO application and migrated the projects to Oracle JDeveloper 11*g*, you can now recompile each of the projects and redeploy them one by one.

Review, compile, and redeploy each project in the following order:

- About the CreditService, RapidService, and CustomerService Projects

- SelectManufacturer

- DHLShipment

- FulfillmentESB

- CreditService

- RapidService

- CustomerService

- SOAOrderBooking

- OrderBookingESB

- ApproveOrderTaskForm

A.8.1 About the CreditService, RapidService, and CustomerService Projects

The CreditService, RapidService, and CustomerService projects in the SOA Order Booking application are Java EE Web services that are referenced by the other projects in the application.

As a result, for the purposes of this example, it is not necessary to upgrade these three projects to 11*g*. Instead, you can continue to use the Oracle Application Server 10*g* versions of these services, which are currently running in your 10*g* environment.

If you choose to upgrade these three Web services projects, note the following:

- The instructions for modifying the Web services projects in this Appendix should be adequate if you are familiar and experienced with developing Java EE applications and Web services; however, they are not designed to serve as a tutorial for upgrading Java EE projects.

- If you choose to upgrade the CustomerService Web services project, some additional changes will be required to the SOAOrderBooking project, which are not documented in detail in this Appendix.

A.8.2 SelectManufacturer

The SelectManufacturer project is an asynchronous BPEL process. This means that it contains a receive activity to initiate the BPEL process flow and an invoke activity to call back the client asynchronously with the results (that is, the quote) at the end of the flow.

To compile and redeploy the SelectManufacturer project:

1. Right-click the project in the Application Navigator.

2. From the context-menu, select **Make SelectManufacturer.jpr.**

   The project should compile successfully.

3. Right-click the project again and select **Deploy** > **SelectManufacturer**.

   Oracle JDeveloper displays the first page of the Deployment wizard.

4. Follow the instructions on the wizard pages to deploy the project on the **SoademoApplicationServer**.

A.8.3 DHLShipment

The DHLShipment project was created as an Oracle Enterprise Service Bus (ESB) service in Oracle Application Server 10*g* Release 3 (10.1.3). It should require no changes after opening the project in Oracle JDeveloper 11*g*.

To compile and redeploy the DHLShipment project:

1. Right-click the project in the Application Navigator.

2. From the context-menu, select **Make DHLShipment.jpr**.

3. Right-click the project again and select **Deploy** > **DHLShipment**.

   Oracle JDeveloper displays the first page of the Deployment wizard.

4. Follow the instructions on the wizard pages to deploy the project on the **SoademoApplicationServer**.

A.8.4 FulfillmentESB

The FulfillmentESB project was created in 10*g* Release 3 (10.1.3) as an Oracle Enterprise Service Bus (ESB) project.

After an order has been approved, the SOAOrderBooking project invokes the FulfillmentESB project to determine the shipping method for the order.

Orders are sent to the USPS through a file adapter. The order information is written to a file. For Fedex, orders are sent through a database adapter. The order information is written to a database. The FulfillmentESB project also sends orders to a JMS queue.

Before you can compile and deploy this project, you need to make some minor changes in Oracle JDeveloper 11*g*:

- Modifying the FulfillmentESB Project in Oracle JDeveloper 11g

- Compile and Deploy the FulfillmentESB Project

A.8.4.1 Modifying the FulfillmentESB Project in Oracle JDeveloper 11*g*

Make the following changes to the FulfillmentESB project:

1. Expand the **FullfillmentESB** project and double-click **composite.xml**.

2. Oracle JDeveloper displays the FulfillmentESB composite in the SOA Composite Editor (Figure A–4).

   For more information, see "What Happens When You Create a SOA Application and Project" in the *Oracle Fusion Middleware Developer's Guide for Oracle SOA Suite*.

3. Double-click the icon that represents the **Fulfillment.FulfillmentBatch** reference in the right swimlane.

   This external reference represents the Fulfillment.FulfillmentBatch JMS adapter.

   When you double-click this reference, Oracle JDeveloper displays the Adaper Configuration Wizard.

4. Follow the instructions in the Adapter Configuration Wizard to reconfigure the JMS Adapter for your Oracle Fusion Middleware 11*g* environment.

   You can accept all the default settings in the wizard, except for the following:

   ■ Select **Oracle Weblogic JMS** as the Oracle Enterprise Messaging service.

   ■ Select the Application Server connection that you created in "Task 5d: Create a Connection to Oracle WebLogic Server".

   ■ Click the **Browse** button next to the queue name and select **demoFulfillmentQueue**.

     This should cause `jms/demoFullfillmentQueue` to be entered into the Queue Name field.

   ■ Set the JNDI name to `eis/demo/Queue`.

5. When you return to the Composite Editor, double-click the icon that represents the **Fulfillment.FedexShipment** reference in the right swimlane.

   This external reference represents a database adapter.

   When you double-click this reference, Oracle JDeveloper displays the Adaper Configuration Wizard.

6. Follow the instructions in the Adapter Configuration Wizard to reconfigure the database adapter for your Oracle Fusion Middleware 11*g* environment.

   You can accept all the defaults in the wizard, except for the following changes:

   ■ Select `soademo` as your database connection.

   ■ Set the JNDI name to `eis/DB/soademoDS`.

7. Update the Fulfillment.USPSShipment external reference (which represents a file adapter), as follows:

   a. Create a directory on disk for the file adapter; for example:

      ```
      MW_HOME/Oracle_SOA1/soa/USPSShipment
      ```

   b. Return to the Composite Editor for the FulfillmentESB project, and double-click the **FulfillmentUSPSShipment** external reference.

      Oracle JDeveloper displays the Adapter Configuration Wizard.

   c. Follow the instructions on the screen until the File Configuration screen appears.

   d. In the **Directory for Outgoing Files (physical path)** field, enter the path to the directory you created in Step 7a.

   e. Click **Next** and follow the rest of the adapter configuration instructions.

Figure A–7   Composite Editor for the FulfillmentESB Project of the SOA Order Booking Application
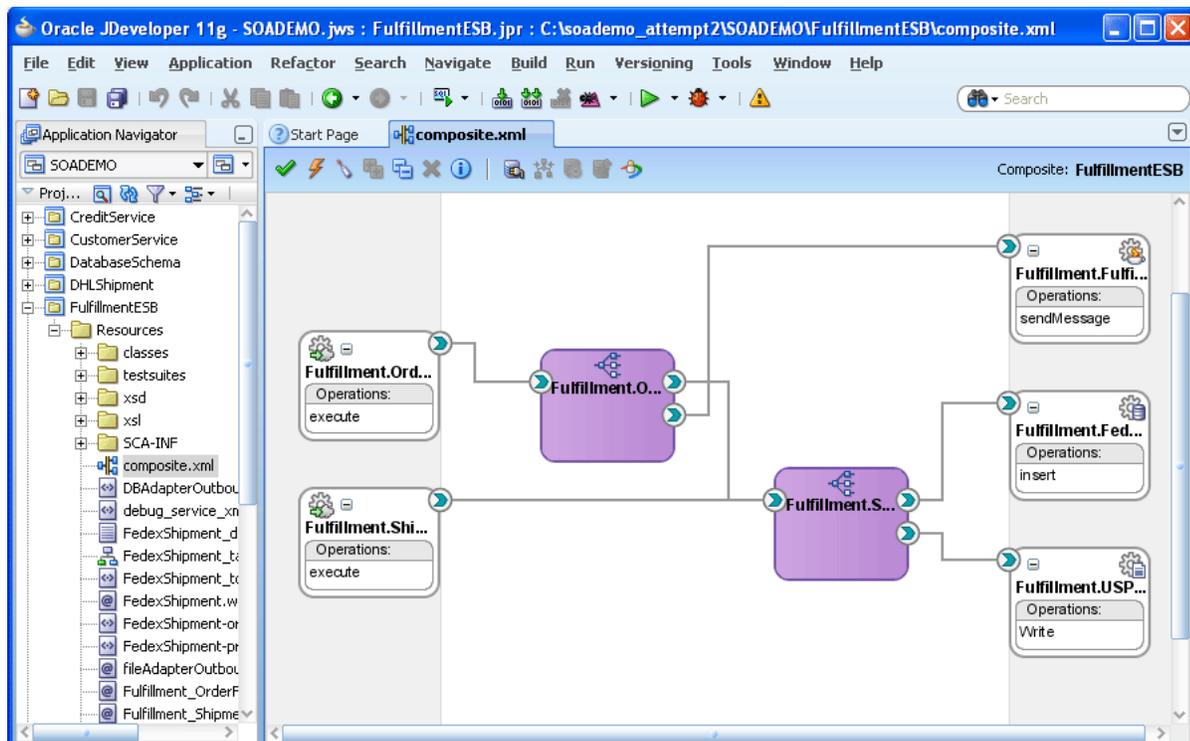
A.8.4.2 Compile and Deploy the FulfillmentESB Project

To compile and redeploy the Fulfillment project:

1. Right-click the project in the Application Navigator.

2. From the context-menu, select **Make FulfillmentESB.jpr**.

3. Right-click the project again and select **Deploy** > **FulFillmentESB**.

   Oracle JDeveloper displays the first page of the Deployment wizard.

4. Follow the instructions on the wizard pages to deploy the project on the **SoademoApplicationServer**.

A.8.5 CreditService

The CreditService project checks whether a customer's credit card is valid or not. In this SOA Order Booking application, the code simply checks the value of the credit card number. If the credit card number is less than 12345678, then the card is invalid. Otherwise, it is valid.

This project is a Java EE Web services project. Before you proceed, refer to Section A.8.1, "About the CreditService, RapidService, and CustomerService Projects".

Before you can compile and deploy this project, you must upgrade the Web service from OC4J JAX-RPC to Oracle WebLogic Server JAX-RPC:

- Upgrading the CreditService Web Services

- Compiling and Deploying the CreditService Project

A.8.5.1 Upgrading the CreditService Web Services

To upgrade the OC4J JAX-RPC Web service to Oracle WebLogic Server JAX-RPC:

1. In the Application Navigator, expand the CreditService project, and select the following:

   Application Sources > org.soademo.creditservice > CreditService

2. Right-click the Credit Service Web service and select **Upgrade Web Service to WLS JAX-RPC Configuration** from the context menu.

   Oracle JDeveloper displays the Confirm Upgrade dialog box.

3. Click Yes to upgrade the CreditService Web service to Oracle WebLogic Server JAX-RPC.

4. When the upgrade is complete, save the project.

Figure A–8   Confirmation Dialog Box When Upgrading the CreditService Web Service
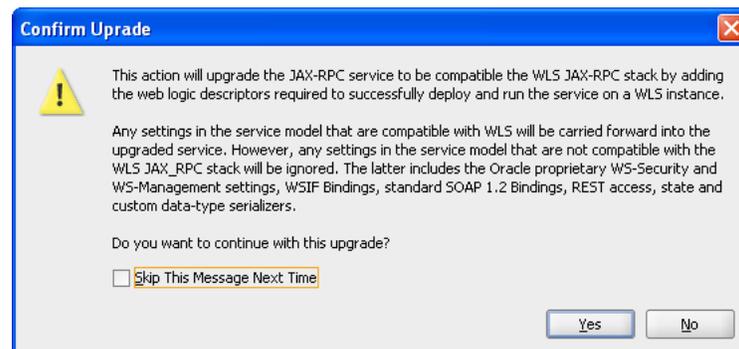
A.8.5.2 Compiling and Deploying the CreditService Project

To compile and redeploy the CreditService project:

1. Right-click the project in the Application Navigator.

2. From the context-menu, select **Make CreditService.jpr**.

3. Right-click the project again and select **Deploy** > **CreditService**.

   Oracle JDeveloper displays the first page of the Deployment wizard.

4. Follow the instructions on the wizard pages to deploy the project on the **SoademoApplicationServer**.

A.8.6 RapidService

The RapidService project represents a supplier that provides price quotes for customer orders. The other supplier which it competes with is SelectManufacturer. The supplier that provides the lower quote for a customer's order gets to fulfill the order.

This project is a Java EE Web services project. Before you proceed, refer to Section A.8.1, "About the CreditService, RapidService, and CustomerService Projects".

Before you can compile and deploy this project, you must modify it using Oracle JDeveloper 11*g*:

■ Upgrade the RapidService Web Service from OC4J JAX-RPC to Oracle WebLogic Server JAX-WS

■ Compiling and Deploying the RapidService Project

A.8.6.1 Upgrade the RapidService Web Service from OC4J JAX-RPC to Oracle WebLogic Server JAX-WS

Before you can compile and deploy this project, you must upgrade the OC4J JAX-RPC Web services to Oracle WebLogic Server JAX-WS.

The following procedure uses the "top-down" approach to regenerate the Web service using the existing WSDL:

1. Remove the existing JAX-RPC Web service files from the RapidService project:

    a. In Oracle JDeveloper 11*g* application navigator pane, open the RapidService project and expand following node:

       Application Sources > org.soademo.rapidservice

    b. Right-click the **RapidService** node and select **Delete** from the context menu.

    c. In the Delete Web Service? dialog box, click **Select All** to select all the files shown in the dialog box.

    d. Clear the check box for the RapidService.wsdl file, as shown in Figure A–9.

    e. Click **OK**.

Figure A–9   Deleting the RapidService Web Service
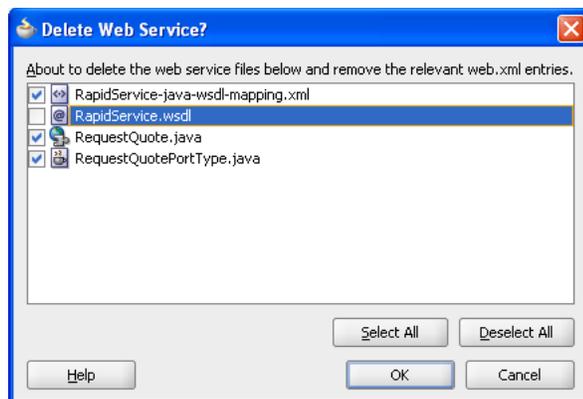
    f. Expand the contents of the **Types** node and delete Item.java, Quote.java.

    g. Expand the contents of the **Web Content** > **WEB-INF** node, and then delete oracle-webservices.xml, web.xml, and webservices.xml.

       Figure A–10 shows the files under RapidService that you must select and then delete.

Figure A–10   Removing the RapidService Web Services Files and Related Files
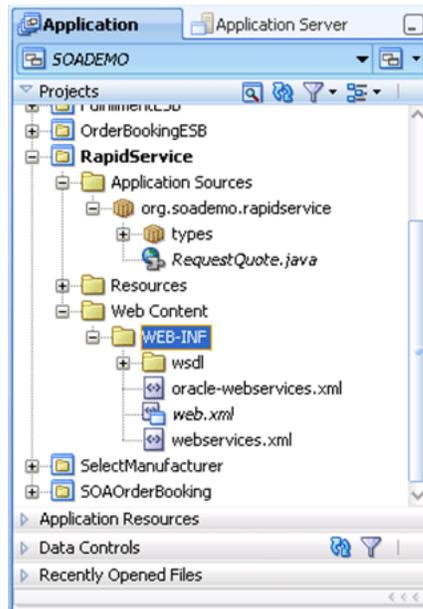
2. Remove JAX-RPC Web Services from the classpath:

    a. Right-click the RapidService project and select **Project Properties** from the context menu.

        Oracle JDeveloper displays the Project Properties window.

    b. Select **Libraries and Classpath**, select **JAX-RPC 10.1.3 Web Services**, click **Remove**, and the click **OK**.

3. Generate the new JAX-WS Web service:

    a. Right click the `RapidService.wsdl` file under the following location in the RapidService project:

        Application Sources > org.soademo.rapidservice > RapidService

    b. Select **Create Web Service** from the context menu.

        Oracle JDeveloper displays the Create Java Web Service From WSDL wizard.

    c. Follow the instructions on the screen to create the new JAX-WS Web service:

        Be sure to select **Java EE 1.5, with support for JAX-WS Annotations** on the Select Deployment Platform screen.

4. Modify the `poItemsQuote` method in the `RequestQuoteImpl.java` file as follows:

```
public Quote poItemsQuote(@WebParam(name = "param0", targetNamespace =
"http://rapidservice.soademo.org/types/")
        List<Item> param0) {
        Long totalPrice = new Long(0);
        for (int i = 0; i < param0.size(); i++) {
            Item localItem = param0.get(i);
            totalPrice += localItem.getQuantity() * 110;
        }
        Quote priceQuote = new Quote();
        priceQuote.setSupplierName("RapidDistributors");
        priceQuote.setSupplierPrice(totalPrice.toString());
```

```
                    return priceQuote;
        }
```

5. Modify the orderQuote method in the `RequestQuoteImpl.java` file, as follows:

```
public Quote orderQuote(@WebParam(name = "param0", targetNamespace =
            "http://rapidservice.soademo.org/types/")
    String param0, @WebParam(name = "param1", targetNamespace =
            "http://rapidservice.soademo.org/types/")
    String param1, @WebParam(name = "param2", targetNamespace =
            "http://rapidservice.soademo.org/types/")
    String param2, @WebParam(name = "param3", targetNamespace =
            "http://rapidservice.soademo.org/types/")
    String param3, @WebParam(name = "param4", targetNamespace =
            "http://rapidservice.soademo.org/types/")
    String param4) {
        Quote priceQuote = new Quote();
        priceQuote.setSupplierName("RapidDistributors");
        priceQuote.setSupplierPrice("5000");
    return priceQuote;
}
```

6. Save your changes to the RapidService project.

A.8.6.2 Compiling and Deploying the RapidService Project

To compile and redeploy the Fulfillment project:

1. Right-click the project in the Application Navigator.

2. From the context-menu, select **Make RapidService.jpr**.

3. Right-click the project again and select **Deploy** > **RapidService**.

    Oracle JDeveloper displays the first page of the Deployment wizard.

4. Follow the instructions on the wizard pages to deploy the project on the **SoademoApplicationServer**.

A.8.7 CustomerService

The CustomerService project is an EJB implementation that provides methods that enable client applications, such as the SOADemo-Client application, to retrieve customer information from the database and add customers to the database.

This project is a Java EE Web services project. Before you proceed, refer to Section A.8.1, "About the CreditService, RapidService, and CustomerService Projects".

Before you can compile and deploy the project, there some changes you have to make in Oracle JDeveloper 11g.

■   Modifying Specific CustomerService Project Class Files

■   Modifying the persistence.xml File in the CustomerService Project

■   Compiling and Deploying the CustomerService Project

A.8.7.1 Modifying Specific CustomerService Project Class Files

To modify the CustomerService project in Oracle JDeveloper 11g, you must modify the following files in the `/src/org/soademo/customerservice/` directory of the project:

■   **business/Customerservice.java:**

Remove "`throws RemoteException`" from the `finsCustomerById()` method.

- **client/CustomerServiceClient.java** and **business/CustomerServiceClient.java**

  a. Locate the section of the class file shown in Example A–1 and modify it so it reflects the changes shown in Example A–2.

  b. Change the lookup to the following:

  ```
  CustomerService customerService =
        (CustomerService)context.lookup(

  "CustomerService#org.soademo.customerservice.business.CustomerService");
  ```

- **business/CustomerserviceBean.java**:

  Modify this file so its contents reflects the changes shown in Example A–3.

Example A–1   Original CustomerService.java File

```
private static Context getInitialContext() throws NamingException {
    Hashtable en = new Hashtable();
    //  Standalone OC4J connection details
    env.put( Context.INITIAL_CONTEXT_FACTORY,
                    "oracle.j2ee.rmi.RMIInitialContextFactory" );
    env.put( Context.SECURITY_PRINCIPAL, "oc4jadmin" );
    env.put( Context.SECURITY_CREDENTIALS, "welcome1" );
    env.put(Context.PROVIDER_URL, "ormi://localhost:23791/CustomerService");
    return new InitialContext( env );
```

Example A–2   Modified CustomerService.java File

```
private static Context getInitialContext() throws NamingException {
        Hashtable env = new Hashtable();
        env.put( Context.INITIAL_CONTEXT_FACTORY,
"weblogic.jndi.WLInitialContextFactory" );
        env.put( Context.SECURITY_PRINCIPAL, "weblogic" );
        env.put( Context.SECURITY_CREDENTIALS, "weblogic" );
        env.put(Context.PROVIDER_URL, "t3://edith:7101");
        return new InitialContext( env );
    }
```

Example A–3   Updated CustomerServiceBean.java Class File

```
package org.soademo.customerservice.business;
import javax.ejb.Remote;
import javax.ejb.RemoteHome;
import javax.ejb.Stateless;
import javax.jws.WebMethod;
import javax.jws.WebParam;
import javax.jws.WebService;
import javax.persistence.EntityManager;
import javax.persistence.PersistenceContext;
import org.soademo.customerservice.persistence.Customer;
@Remote
@WebService(serviceName = "CustomerSvc",
            targetNamespace = "http://www.globalcompany.com/ns/customer")
@Stateless(name="CustomerService", mappedName = "CustomerService")
public class CustomerServiceBean implements CustomerServiceLocal,
        CustomerService {
    @PersistenceContext(unitName = "customerServiceUnit" )
```

```java
        private EntityManager em;
        public CustomerServiceBean() {
        }
        /**
         * @param entity
         * @return
         */
        public Object mergeEntity(Object entity) {
            return em.merge(entity);
        }
        public Object persistEntity(Object entity) {
            em.persist(entity);
            return entity;
        }
        public Object refreshEntity(Object entity) {
            em.refresh(entity);
            return entity;
        }
        public void removeEntity(Object entity) {
            em.remove(em.merge(entity));
        }
        /** <code>select object(cust) from Customer cust
                            where cust.custid = :custid</code> */
        @WebMethod
        public Customer findCustomerById(@WebParam(name = "custid",
                targetNamespace = "http://www.globalcompany.com/ns/customer")
                                String custid){
//      public Customer findCustomerById(String custid) {
            return
(Customer)em.createNamedQuery("Customer.findCustomerById").setParameter("custid",
                            custid).getSingleResult();
        }
//      public String getCustomerStatus(String CustomerID) {
        @WebMethod
        public String getCustomerStatus(@WebParam(name = "CustomerID",
                    targetNamespace = "http://www.globalcompany.com/ns/customer")
                    String CustomerID){return
                        findCustomerById(CustomerID).getStatus();
        }
 //     public String addNewCustomer(Customer customer) {
        @WebMethod
        public String addNewCustomer(@WebParam(name = "customer",
                    targetNamespace ="http://www.globalcompany.com/ns/customer")
                    Customer customer){
            em.persist(customer);
            //making the change to return the customer id
            // that client application can use to query orders
            return customer.getCustid();
            //return "New customer added sucessfully to customer database";
        }
 //     public Customer findCustomerByEmail(String email, String password) {
        @WebMethod
        public Customer findCustomerByEmail(@WebParam(name = "email",
                    targetNamespace = "http://www.globalcompany.com/ns/customer")
                    String email, @WebParam(name = "password",
                targetNamespace = "http://www.globalcompany.com/ns/customer")
                            String password){
                                return(Customer)em.createNamedQuery(
                                    "Customer.findCustomerByEmail").setParameter("email",
                            email).setParameter("password",
```

```
                                    password).getSingleResult();
        }
}
```

A.8.7.2 Modifying the persistence.xml File in the CustomerService Project

Modify the `src/META-INF/persistence.xml` so it uses Kodo explicitly in Oracle Weblogic Server.

Kodo an implementation of Sun Microsystem's Java Persistence API (JPA) specification and Java Data Objects (JDO) specification for transparent data objects. Kodo is available as a stand-alone product and is integrated within WebLogic Server.

Add the following entry inside the `<persistence-unit>` element in the persistence.xml file:

```
<provider>kodo.persistence.PersistenceProviderImpl</provider>
```

A.8.7.3 Compiling and Deploying the CustomerService Project

To compile and redeploy the Fulfillment project:

1. Right-click the project in the Application Navigator.

2. From the context-menu, select **Make CustomerService.jpr**.

3. Right-click the project again and select **Deploy** > **CustomerService**.

   Oracle JDeveloper displays the first page of the Deployment wizard.

4. Follow the instructions on the wizard pages to deploy the project on the **SoademoApplicationServer**.

A.8.8 SOAOrderBooking

The SOAOrderBooking project, which is a BPEL project, represents the main flow in the SOA Order Booking application. It sends the order information to the appropriate services at the appropriate times. For example, it contacts CreditService to check the customer's credit card, and if the credit card is acceptable, it contacts the suppliers (Select Manufacturer and Rapid Distributors) to get price quotes for the order.

Before you can compile and deploy this project from Oracle JDeveloper 11*g*, you must make some changes:

- Modifying the SOAOrderBooking Project

- Optionally, Changing SOAOrderBooking to Use Upgraded Web Services

- Compiling and Deploying the SOAOrderBooking Project:

A.8.8.1 Modifying the SOAOrderBooking Project

Make the following changes to the SOAOrderBooking project in Oracle JDeveloper 11*g*:

1. Expand the SOAOrderBooking project, and open the **SOA Content > ApproveOrder** folder.

2. Double-click the **ApproveOrder.task** file, which defines the human approval task.

3. Double-click the Manager icon under the Assignment and Routing section, select **User** from the **Identification Type** drop-down menu, and enter `weblogic` in the associated field.

4. Open the **SOA Content** folder, and double-click **composite.xml**.

Oracle JDeveloper displays the SOAOrderBooking composite in the SOA Composite Editor.

5. Double-click the icon that represents the **Order** reference.

   This external reference represents the Order database adapter.

   When you double-click this reference, Oracle JDeveloper displays the Adaper Configuration Wizard.

6. Follow the instructions in the wizard, and accept the default settings, except for the following changes:

   ■ Select `soademo` as your database connection.

   ■ Set the JNDI name to `eis/DB/soademoDS`.

7. Perform Steps 5 and 6 for the OrderSequence reference and the OrderStatus external references, which both also represents database adapters that must connect to the soademo database and schema.

8. Double click the **SelectService** reference.

   Oracle JDeveloper displays the Update Reference dialog box.

   Click on the WSDL icon next to the WSDL URL field to display the SOA Resource Browser.

   Select **Resource Palette** from the drop-down menu, and then select **SelectManufacturer** in the Resource Palette.

9. Repeat Step 8 for the **OrderFulfillment** reference, but this time select the **Fulfillment.OrderFulfillment_ep** entry point in FulfillmentESB from the resource palette.

10. Save the change you made to the project.

A.8.8.2 Optionally, Changing SOAOrderBooking to Use Upgraded Web Services

If you upgraded and redeployed the RapidService, CreditService, and CustomerService projects earlier in this procedure, you can now modify the SOAOrderBooking project to use the new 11*g* Web services, rather than the ones that are deployed and still running in your Oracle Application Server 10*g* environment.

Alternatively, you can continue to reference the existing 10*g* Web services. For more information, see Section A.8.1, "About the CreditService, RapidService, and CustomerService Projects".

To modify the SOAOrderBooking project so it uses the newly upgraded RapidService, CreditService, and CustomerService Web services:

1. Locate the new WSDL URL for the upgraded Web services projects:

   a. Open your Web browser and log in to the Fusion Middleware Control.

   ```
   For example: host24.mycompany.com:7001/em
   ```

   For more information, see "Getting Started Using Oracle Enterprise Manager Fusion Middleware Control" in the *Oracle Fusion Middleware Administrator's Guide*.

   b. Open the **Application Deployments** folder in the Target Navigation Pane.

   c. Click the **SOADEMO-RAPIDSERVICE-RapidService-WS** application in the list of deployed applications.

Fusion Middleware Control displays the Application Deployment page for the application you selected.

   d. Click the Test icon for the SOADEMO-RAPIDSERVICE-RapidService-WS in the Web Services table in the Entry Points section of the page.

   e. On the Test Web Services page, note the WSDL URL for the Web service.

   f. Repeat steps c through e for the CreditService and CustomerService applications.

2. Modify the SOAOrderBooking composite with the new WSDL URLs:

   a. Expand the **SOAOrderBooking** project and double-click **composite.xml**.

   b. Oracle JDeveloper displays the SOAOrderBooking composite in the SOA Composite Editor.

   c. Double-click the icon that represents the **RapidService** external reference in the right swimlane.

   This external reference represents the RapidService Web service.

   When you double-click this reference, Oracle JDeveloper displays the Update References dialog box.

   d. Enter the new 11*g* WSDL URL in the **WSDL URL** field and click **OK**.

   e. Repeat Steps c and d for the **CreditValidationService** and **CustomerService** external references.

   f. Save your changes to the project.

3. If you upgraded the CustomerService project to 11*g*, then you must also modify the `SOAOrderBooking.bpel` file to reflect the changes made to the CustomerService while upgrading it to 11*g*.

   This involves reconnecting the Web services binding, and updating the `customerServiceRequest`, `customerServiceResponse`, and subsequent variable queries.

A.8.8.3 Compiling and Deploying the SOAOrderBooking Project:

To compile and redeploy the SOAOrderBooking project:

1. Right-click the project in the Application Navigator.

2. From the context-menu, select **Make**.

   The project should compile successfully.

3. Right-click the project again and select **Deploy** > **SOAOrderBooking**.

   Oracle JDeveloper displays the first page of the Deployment wizard.

4. Follow the instructions on the wizard pages to deploy the project on the **SoademoApplicationServer**.

A.8.9 OrderBookingESB

The OrderBookingESB project is the entry point to the SOA Order Booking application.

The OrderBookingESB project invokes the SOAOrderBooking project, which is a BPEL project that defines the main flow of the SOA Order Booking application.

It was created in 10*g* Release 3 (10.1.3) as an Oracle Enterprise Service Bus (ESB) project.

Before you can compile and deploy this project from Oracle JDeveloper 11*g*, you must make some changes:

- Modifying the OrderBookingESB Project
- Compiling and Deploying the OrderBookingESB Project:

A.8.9.1 Modifying the OrderBookingESB Project

To modify the OrderBookingESB project in Oracle JDeveloper 11*g*:

1. Expand the **OrderBookingESB** project and double-click **composite.xml**.

2. Oracle JDeveloper displays the OrderBookingESB composite in the SOA Composite Editor (Figure A–4).

3. Double-click the icon that represents the **OrderBooking** external reference in the right swimlane.

    This external reference represents the OrderBooking Web service.

    When you double-click this reference, Oracle JDeveloper displays the Update References dialog box.

4. Click the WSDL icon next to the WSDL URL field to display the SOA Resource Browser.

    Select **Resource Palette** from the drop-down menu, expand the application server connection, and select the **SOAOrderBooking** > **Client** resource from the Resource Palette.

5. Save your changes to the project.

A.8.9.2 Compiling and Deploying the OrderBookingESB Project:

To compile and redeploy the OrderBookingESB project:

1. Right-click the project in the Application Navigator.

2. From the context-menu, select **Make**.

    The project should compile successfully.

3. Right-click the project again and select **Deploy** > **SOAOrderBookingESB**.

    Oracle JDeveloper displays the first page of the Deployment wizard.

4. Follow the instructions on the wizard pages to deploy the project on the **SoademoApplicationServer**.

A.8.10 ApproveOrderTaskForm

The ApproveOrderTaskForm project is created automatically when you open and upgrade the SOA Order Booking application in Oracle JDeveloper 11*g*.

No changes are required to this project. However, to deploy this project, you must use the deploy it as an application and not as a project:

1. Select the ApproveOrderTaskForm project.

2. From the **Application** menu, select **Deploy** > **ApproveOrderTaskForm**.

    Oracle JDeveloper displays the first page of the Deployment wizard.

3. Follow the instructions on the wizard pages to deploy the project on the **SoademoApplicationServer**.

A.9 Task 8: Test the SOA Order Booking Application in Fusion Middleware Control

Oracle Enterprise Manager Fusion Middleware Control provides a Web-based management console that you can use to manage your Oracle SOA Suite composite applications.

To test the SOA Order Booking application in Fusion Middleware Control:

1. Log in to the Fusion Middleware Control for the Oracle WebLogic Server domain where you deployed your SOA Order Booking application projects:

   The URL for Fusion Middleware Control is usually:

   ```
   http://hostname:7001/em
   ```

   For more information, see "Getting Started Using Oracle Enterprise Manager Fusion Middleware Control" in the *Oracle Fusion Middleware Administrator's Guide*.

2. In the Target Navigation Pane, expand the **SOA** folder, and then expand the **soa-infra (soa_server1)** node to display the SOA composite applications deployed to the domain.

3. Click the **OrderBookingESB** composite application.

   Fusion Middleware Control displays the Dashboard tab of the OrderBookingESB composite application home page, which you can use to manage and monitor a deployed SOA composite application.

4. Click **Test** in the button bar across the top of the page.

   Fusion Middleware Control displays the Test Web Services page, which displays the WSDL location, Operations list, and two tabs, Request and Response tabs.

5. Scroll down to the Input Arguments section of the page.

   From this section of the page, you can enter input data to test the composite application. The SOA Order Booking application ZIP file contains two text files that contain sample data you use to test the SOA Order Booking application.

6. Locate the following input sample test file in the root directory of the SOADEMO_HOME:

   ```
   sampleInput2.txt
   ```

7. Open `sampleInput2.txt` and select and copy the entire contents of the file.

8. Select **XML View** from the drop-down menu in the Input Arguments section of the Test Web Service page.

9. Select and delete the XML content displayed on the Request tab in the Input Arguments section of the page.

   Figure A–11 shows the XML content selected in the Input Arguments section of the page.

10. Paste the contents of the `sampleinput2.txt` file into the text field, and click **Test Web Service** at the bottom of the page.

   After a few seconds, a message appears indicating that the Web service invocation was successful. It also displays the return value obtained by the service.

11. Click the **Launch Message Flow Trace** link to see the flow of your composite and the status of each service, component, and reference.

Fusion Middleware Control displays the Flow Trace page (Figure A–12), which shows each step in the process flow. Using this view, you can monitor the application as it progresses through each component and task in the application.

You can also a click a component to drill down into the Instance Details for the component.

12. For further testing, you can perform this same procedure using `sampleinput1.txt`, which is in the same directory as `sampleinput2.txt` and submits an order of more than $1000.

For `sampleinput1.txt`, you'll notice that the work flow stops and does not continue, until you log into the following URL and approve the order:

```
http://domain:port/integration/worklistapp
```

Figure A–11   Selecting XML Content in the Input Argument Section of the SOA Composite - Test Web Service Page
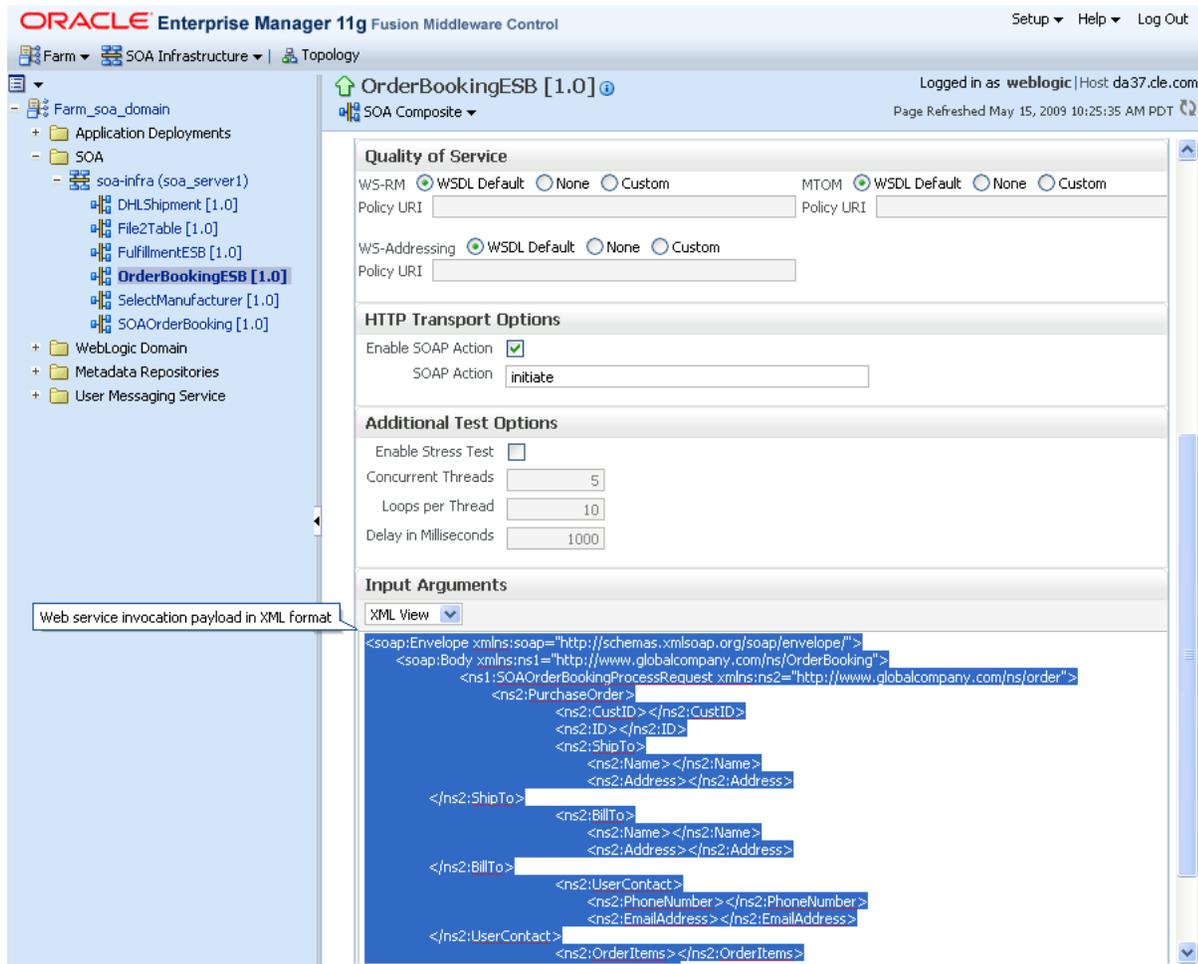
Figure A–12   Flow Trace of the SOA Order Booking Application in Fusion Middleware Control

B

Upgrading Oracle Business Rules Dictionaries and Projects

You can migrate Oracle Business Rules dictionaries and projects from previous releases. Before you can use a dictionary created with Oracle Business Rules Release 10.1.3.x with Oracle Business Rules for Oracle Fusion Middleware 11*g* Release 1 (11.1.1), you must migrate the dictionary to the new dictionary format. Depending on how the Release 10.1.3.x dictionary was created, there are several possible migration paths to migrate a dictionary from Oracle Business Rules Release 10.1.3.x format to Oracle Fusion Middleware 11*g* Release 1 (11.1.1).

This appendix includes the following sections:

- Section B.1, "Using Oracle JDeveloper to Migrate an Oracle Business Rules Dictionary".

- Section B.2, "Using Rule Migrator Tool to Migrate an Oracle Business Rules Dictionary"

- Section B.3, "Using MigrateRuleRepository with Oracle Business Rules SDK to Migrate a Dictionary"

- Section B.4, "Oracle Business Rules Manual Migration Tasks"

B.1 Using Oracle JDeveloper to Migrate an Oracle Business Rules Dictionary

When a dictionary is part of a an application created in a 10.1.3.x BPEL Decision Service, you can migrate to the Oracle Fusion Middleware 11*g* Release 1 (11.1.1) dictionary format when you open the BPEL process in Oracle JDeveloper.

To use Oracle JDeveloper to migrate a dictionary:

1. Start Oracle JDeveloper, as shown in Figure B–1, this displays the Oracle JDeveloper start page.

Figure B–1   Creating an Application in Oracle JDeveloper
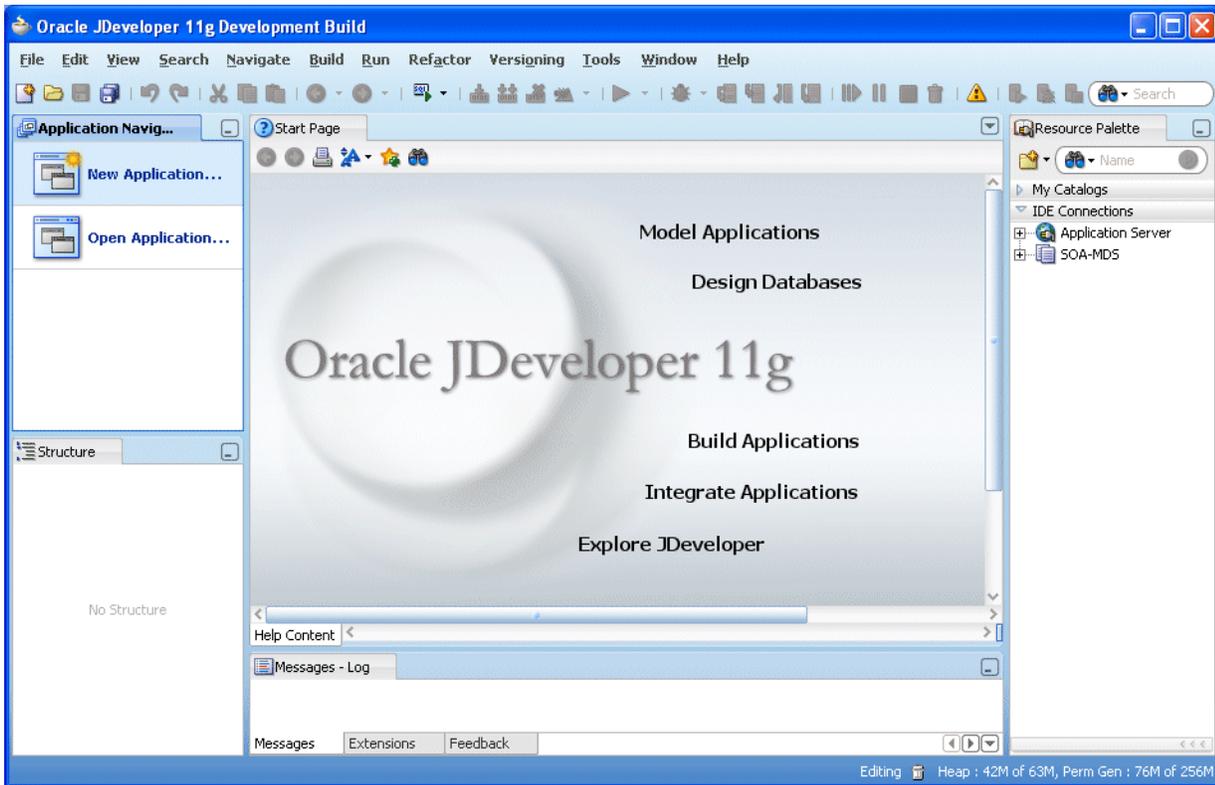
2. From the **File** menu select **Open**.

3. Navigate to your project and select **Open**.

   JDeveloper invokes the Oracle BPEL Process Manager migration utility to convert the project, including its rules dictionary.

4. Examine the **Messages - Log** window for any errors.

   In some cases, you may need to manually edit the migrated rules dictionary using the Oracle Business Rules Designer to complete the migration. For more information, see Section B.4, "Oracle Business Rules Manual Migration Tasks".

5. From the **File** menu select **Save All**.

B.2 Using Rule Migrator Tool to Migrate an Oracle Business Rules Dictionary

You can use the `MigrateRuleRepository` command-line tool to migrate a dictionary from 10.1.3.x to Oracle Fusion Middleware 11*g* Release 1 (11.1.1) format. Note that most migration scenarios require that you perform manual migration tasks because 10.1.3.x has concepts that have been superseded in Oracle Fusion Middleware 11*g* Release 1 (11.1.1) by updated and more powerful constructs. For more information on the manual migration tasks, see Section B.4, "Oracle Business Rules Manual Migration Tasks".

The command-line tool is a Java class called the following:

`oracle.rules.tools.migrator.MigrateRuleRepository`

It is located in the following directory inside the SOA Oracle home:

SOA_HOME/soa/modules/oracle.rules.migration_11.1.1/migrator.jar

To run the command-line tool, you must have a specific set of JAR files available in the classpath. The list of required JAR files and libraries is included in Section B.3.1, "How to Migrate a Dictionary with Oracle Business Rules SDK".

For example, to set the classpath and run the migrator on Linux:

```
> export CLASSPATH=required_jar_files
> java -cp $CLASSPATH oracle.rules.tools.migrator.MigrateRuleRepository OPTIONS
```

The following shows a sample `MigrateRuleRepository` command:

```
MigrateRuleRepository -all
        -origType FILE_REPO
        -origLocation /scratch/MyRepository
        -destLocation /scratch/MyMigratedDictionaries
        -importedClasspath /scratch/foo.jar:/scratch/myclasses
        -xsdDirectory /scratch/myschemas
        -xsdGeneration /scratch/jaxb_classes
```

Table B–1 lists the required `MigrateRuleRepository` command-line arguments.
Table B–2 lists the optional `MigrateRuleRepository` command-line arguments.
Table B–3 lists the WebDAV `MigrateRuleRepository` command-line arguments.

Table B–1   Required Migrate Rule Repository Options

Option	Description
`-origType {FILE_REPO	WEBDAV_REPO}`	Specify one of these strings to indicate the type of the origin repository.
`-origLocation` *origin-repo*	Argument specifies the Release 10.1.3.x repository containing a dictionary to be migrated. This is specified either with string that is either a URL to a WebDAV repository or a file path.
`-destLocation` *dir-name*	Specifies the destination directory in which the migrated dictionaries are placed. The migrated dictionaries are placed in a directory structure constructed with the `-destlocation`, `-destPackageName` and `-destDictionaryName` arguments.
	For example,
	`dir-name/dest-pkg/dest-dict-name.xml`
	The `pkg-name` is migrated into directories created in *dir-name*.
`-importedClasspath` *classpath*	Specifies the classpath to use for importing Java fact types during migration.
`-xsdDirectory` *schema-dir*	Specifies the schema directory which contains all schemas used in XML fact types.
`-xsdGeneration` *jaxb-class-destination*	Specifies the directory to generate JAXB classes into when importing schemas.
`-all`	Migrate all dictionaries in the origin repository.
	Either use `-all`, or specify the `-origDictionaryName` and the `-origVersionName`
`-origDictionaryName` *origin-dict*	Do not use with the `-all` option
	Specifies the name of the dictionary to be migrated.
`-origVersionName` *origin-version*	Do not use with `-all` option.
	Specifies the version of dictionary to be migrated.

Table B–2   Optional Migrate Rule Repository Options

Option	Description
`-destPackageName` *dest-pk*g	Specifies the package name to use when saving the migrated result dictionary in Oracle Business Rules Oracle Fusion Middleware 11*g* Release 1 (11.1.1) format. The migrated dictionaries are placed in a directory structure constructed with the `-destlocation`, `-destPackageName` and `-destDictionaryName` arguments.
`-destDictionaryName` *dest-dict*	Specifies the dictionary name to use when saving the Oracle Business Rules Oracle Fusion Middleware 11*g* Release 1 (11.1.1) format result dictionary. The migrated dictionaries are placed in a directory structure constructed with the `-destlocation`, `-destPackageName` and `-destDictionaryName` arguments.

Table B–3   WebDAV Only Migrate Rule Repository Options

Option	Arguments and Description
`-originPasswordSource` *webdav-password-src*	Argument specifies the Oracle Wallet file containing the WebDAV authentication information.
`-originProxyHost` *webdav-proxy-host*	When a proxy is involved, this option is required. Argument specifies the host to use as proxy for WebDAV access, if necessary.
`-originProxyPort` *webdav-proxy-port*	When a proxy is involved the argument specifies the port to use on proxy host for WebDAV access, if necessary.

B.3 Using MigrateRuleRepository with Oracle Business Rules SDK to Migrate a Dictionary

Using the `MigrateRuleRepository` programmatic interface with Oracle Business Rules SDK. Note that most migration scenarios require that you perform manual migration tasks.

B.3.1 How to Migrate a Dictionary with Oracle Business Rules SDK

To migrate using Java with Oracle Business Rules SDK, you use the `oracle.rules.tools.migrator.MigrateRuleRepository` class.

In the following example, you migrate the dictionary in the `how-to-rules-java` sample located here:
http://www.oracle.com/technology/products/ias/business_rules/index.html.

To migrate a dictionary using Oracle Business Rules SDK:

1. Start Oracle JDeveloper, as shown in Figure B–2, this displays the Oracle JDeveloper start page.

Figure B–2   Creating an Application in Oracle JDeveloper
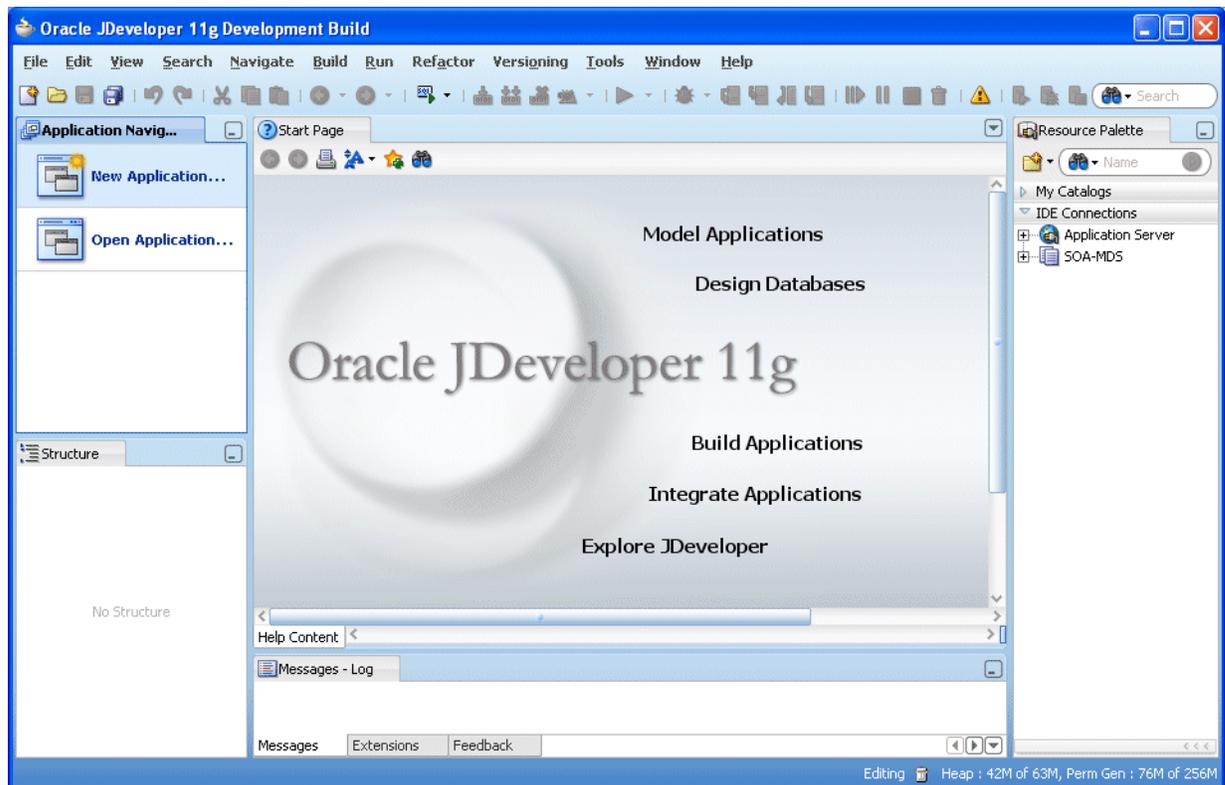
2. In the Application Navigator, click **New Application** if no applications have been created, or if applications have already been created click **Applications** and from the drop-down list choose **New Application**.

3. In the Create Application wizard, enter the name and location for the new application:

   a. In the **Application Name** field, enter an application name. For example, enter `MigrateApplication`.

   b. Enter or browse for a directory name, or accept the default.

   c. Enter an application package prefix or accept the default, no prefix.

   This should be a globally unique prefix and is commonly a domain name owned by your company. The prefix, followed by a period, applies to objects created in the initial project of an application.

   In this sample, you can use the prefix `com.example`.

   d. For an Oracle Business Rules project, select **Generic Application** for the application template, as shown in Figure B–3.

Figure B–3   Adding the Migrate Application
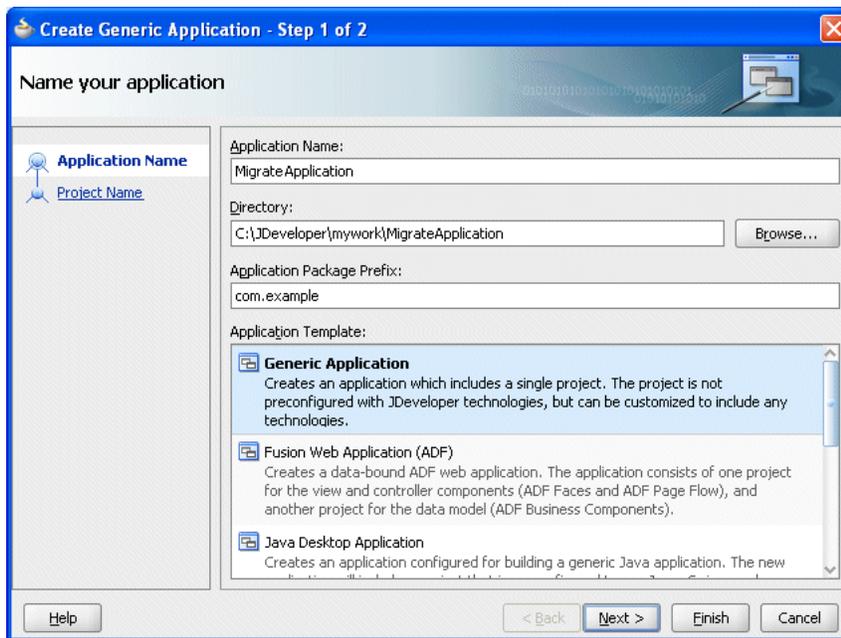
4. Click **Next**.

5. In the Create Generic Application wizard - Name your Generic project page, enter the name and location for the new project as shown in Figure B–4:

   - In the **Project Name** field, enter an application name. For example, enter `MigrateProject`.

   - Enter or browse for a directory name, or accept the default.

   - On the Project Technologies tab, in the Available list, select **Java** and click **Move** to add it to the Selected area.

Figure B–4   Specifying Technologies in a Project
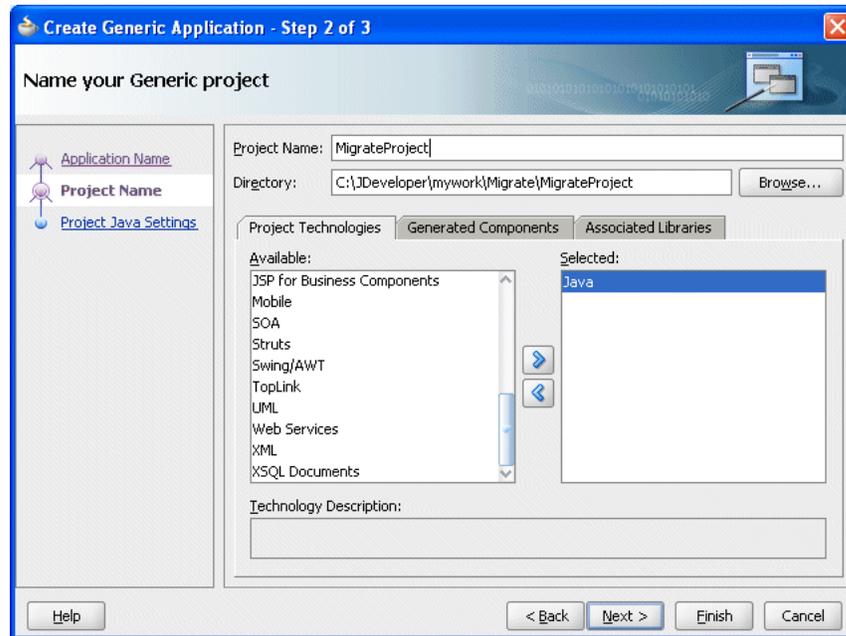
6. Click **Finish**.

7. In Oracle JDeveloper, select the project named **MigrateProject**.

8. Right-click and from the drop-down list select **Project Properties**.

9. Select the **Libraries and Classpath** item.

10. Click **Add JAR/Directory**.

    You must update your project classpath with the following JAR files.

    Note that there may be a JAVA_HOME installed next to JDEV_HOME, in your ORACLE_HOME directory.

    ```
    JAVA_HOME/lib/tools.jar
    JDEV_HOME/soa/modules/oracle.rules.migration_11.1.1/migrator.jar
    JDEV_HOME/soa/modules/oracle.rules.migration_11.1.1/rulesdk.jar
    JDEV_HOME/modules/oracle.adf.model_11.1.1/jr_dav.jar
    JDEV_HOME/modules/oracle.pki_11.1.1/oraclepki.jar
    JDEV_HOME/soa/modules/oracle.rules_11.1.1/rl.jar
    JDEV_HOME/soa/modules/oracle.rules.migration_11.1.1/webdavrc.jar
    ```

11. In the Add Archive or Directory dialog, select the required JAR file and click **Select** as Figure B–5 shows.

Figure B–5   Project Properties Dialog: migrator.jar
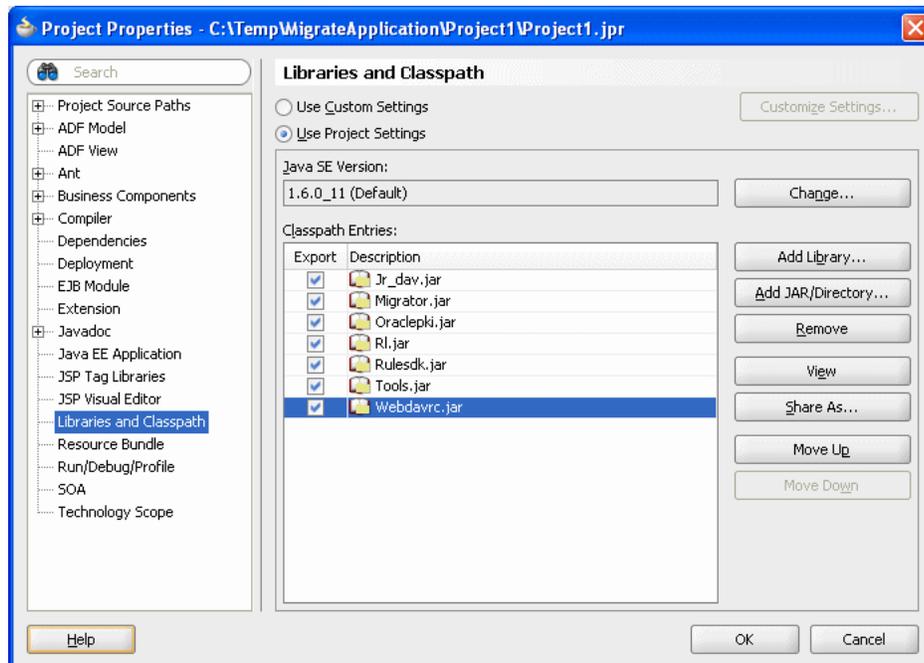
12. Repeat from step 10 for the remaining JAR files.

13. Click **Add Library**.

You must add the following libraries to your project classpath:

- **BC4J Client**

- **Oracle JDBC**

- **Oracle Rules**

- **Oracle XML Parser v2**

- **JAX-RPC Client**

14. In the Add Library dialog, select the required library and click **OK** as Figure B–6 shows.

Figure B–6   Project Properties Dialog: Oracle Rules Library
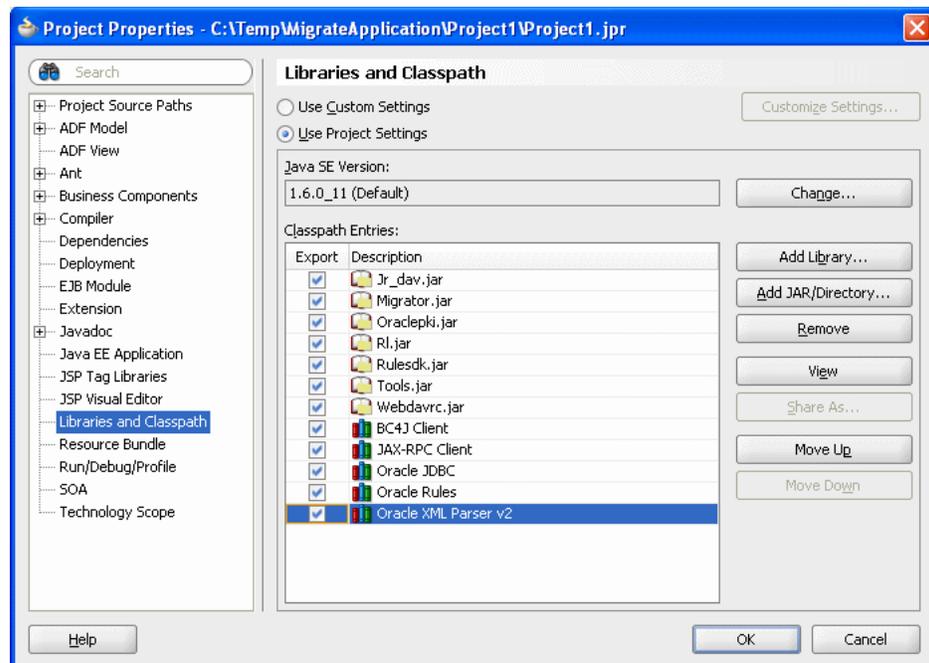
15. Repeat from step 13 for the remaining libraries.

16. Click **OK**.

17. In Oracle JDeveloper, select the project named **MigrateProject**.

18. Right-click and from the drop-down list select **New**.

19. In the New Gallery, in the **Categories** area, select **General**.

20. In the New Gallery, in the **Items** area, select **Java Class**.

21. Click **OK**.

22. In the Create Java Class window, configure the following properties as shown in Figure B–7:

- Enter the **Name** value `Migrate`.

- Enter the **Package** value `com.example`.

- Check the following check boxes:

  - **Public**

  - **<None>**

  - **Constructors from Superclass**

  - **Main Method**

Figure B–7   Creating the Migrate.java Class
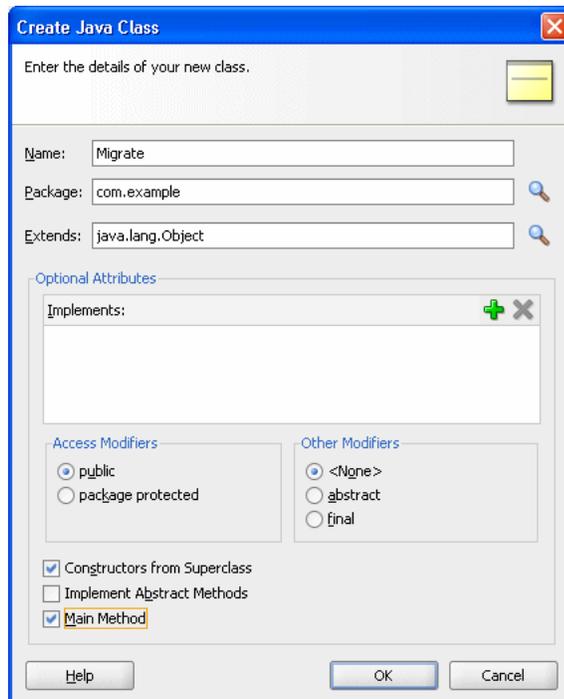
23. Click **OK**.

Oracle JDeveloper displays the Java Class, as shown in Example B–1.

Example B–1   Code Created for New Migrate.java Class

```
package com.example;

public class Migrate {
    public Migrate() {
    }

    public static void main(String[] args) {
        Migrate migrate = new Migrate();
    }
}
```

24. Use the `MigrateRuleRepository` API to add code like that shown in Example B–2.

For more information about the `MigrateRuleRepository` API, see Section B.4.3, "What You May Need to Know About Manual Migration".

Example B–2   Migrate.java Completed

```
package com.example;

import java.io.PrintWriter;

import oracle.rules.tools.migrator.MigrateRuleRepository;

public class Migrate {
    public Migrate() {
        try {

            // Create a migrator instance
```

```
                  MigrateRuleRepository conv = new MigrateRuleRepository();

                  // Set the output buffer
                  conv.setOutLog(new PrintWriter(System.out));

                  // Set input properties (SDK format)
                  conv.setOriginLocation("C:\\Temp\\how-to-rules-java\\dict\\CarRepository");
                  conv.setOriginType(MigrateRuleRepository.FILE_REPO);
                  conv.setOriginDictionaryName(MigrateRuleRepository.MIGRATE_ALL);
                  conv.setOriginVersionName(MigrateRuleRepository.MIGRATE_ALL);
                  conv.setImportedClassPath("C:\\Temp\\how-to-rules-java\\lib\\car.jar");

                  // Set output properties
                  conv.setDestinationLocation("C:\\Temp\\how-to-rules-java\\dict\\CarRepositorySDK2");

                  conv.migrate();
                      String results = conv.getTotalStats();
                      System.out.print(results);
                  }
                  catch (Exception e)
                  {
                      System.out.print(e);
                  }
              }

          public static void main(String[] args) {
              Migrate migrate = new Migrate();
          }
      }
```

25. If the repository you are migrating contains XML Fact Types, you need to specify the location of the schema file that defines these XML Fact Types, and also the output directory that JAXB should use when converting schema files into Java classes.

    Since there are no XML Fact Types in this demo, these lines of code are not needed:

    ```
    conv.setXSDLocation("C:\\Temp\\how-to-rules-java\\schemas"); // or other
            directory, as appropriate
    conv.setXSDOutputDirectory("C:\\Temp\\how-to-rules-java\\jaxbOutput"); // or
            other directory, as appropriate
    ```

26. In the Application Navigator, right-click `Migrate.java` and select **Make**.

27. In the Application Navigator, right-click `Migrate.java` and select **Run**.

28. Examine the **Messages - Log** window for any errors.

    In some cases, you may need to manually edit rules to complete the migration.

29. Retrieve the migrated dictionary from the destination location.

    In this example, from
    `C:\Temp\how-to-rules-java\dict\CarRepositorySDK2`.

B.3.2 What You May Need to Know About the MigrateRuleRepository API

For complete details on the `MigrateRuleRepository` API, see the *Oracle Fusion Middleware Java API Reference for Oracle Business Rules*.

As shown in Example B–3, a typical migration program uses the `MigrateRuleRepository` API to set:

- Input Properties
- Output Properties

B.3.2.1 Input Properties

Input properties define characteristics of the dictionary that you want to migrate. Example B–3 shows a migration program setting typical input properties.

Example B–3   Input Properties

```
// Set input properties (SDK format)
conv.setOriginLocation( "C:\\scratch\\SDK1_Repository" );
conv.setOriginType(MigrateRuleRepository.FILE_REPO);
conv.setOriginDictionaryName(MigrateRuleRepository.MIGRATE_ALL);
conv.setOriginVersionName(MigrateRuleRepository.MIGRATE_ALL);
```

You must set the following mandatory input properties using `MigrateRuleRepository` method:

- `setOriginLocation` to specify the fully qualified path to the rules dictionary to migrate.

- `setOriginType` to specify the type of repository that contains the rules dictionary. You can choose any of the following constants:

  - `MigrateRuleRepository.FILE_REPO`: the rules dictionary is stored in the host file system as a simple file.

  - `MigrateRuleRepository.WEBDAV_REPO`: the rules dictionary is stored in a WebDAV-managed repository.

- `setOriginDictionaryName` to specify the name of the dictionary contained by the specified rules dictionary to migrate. Specify either the dictionary name as a `String` or `MigrateRuleRepository.MIGRATE_ALL` to migrate all dictionaries contained by the specified dictionary.

- `setOriginVersionName` to specify the version of the specified rules dictionary to migrate. Specify either the dictionary version as a `String` or `MigrateRuleRepository.MIGRATE_ALL` to migrate all versions of the specified dictionary.

You may use other `MigrateRuleRepository` methods to define optional input properties as required.

B.3.2.2 Output Properties

Output properties define characteristics of the Oracle Fusion Middleware 11*g* Release 1 (11.1.1) Oracle Business Rules dictionary that the `MigrateRuleRepository` class migrates to, given the input properties you specify (see Section B.3.2.1, "Input Properties"). Example B–4 shows a migration program setting typical input properties.

Example B–4   Output Properties

```
// Set output properties (SDK2 format)
conv.setDestinationLocation( "C:\\scratch\\SDK2_Migrated_Dictionaries" );
```

You must set the following mandatory output properties using `MigrateRuleRepository` method:

- `setDestinationLocation` to specify the fully qualified file name of the new Oracle Fusion Middleware 11*g* Release 1 (11.1.1) Oracle Business Rules based rules dictionary.

You may use other `MigrateRuleRepository` methods to define optional output properties as required.

For example, by default, the new Oracle Fusion Middleware 11*g* Release 1 (11.1.1) Oracle Business Rules dictionary uses the same package name as that used in the original dictionary. To override this behavior, you can specify a new package name using `MigrateRuleRepository` method `setDestinationPackageName`.

B.4 Oracle Business Rules Manual Migration Tasks

Before you can use an a dictionary created with Oracle Business Rules Release 10.1.3.x, you must migrate it to the format required for Oracle Fusion Middleware 11*g* Release 1 (11.1.1). The easiest way to migrate a dictionary that is part of a Decision Service in a BPEL process is to use JDeveloper. Alternatively, you can migrate using the `MigrateRuleRepository` command-line tool. Certain migration scenarios require you to perform manual migration tasks.

B.4.1 How to Migrate JAXB 1.0 to JAXB 2.0

To take advantage of the significant usability improvements present in JAXB 2.0, you must perform the following migration tasks manually.

The most significant difference between JAXB 1.0 and JAXB 2.0 is the classes which are generated. JAXB 1.0 was created before Java 1.5 introduced annotations, so it relies on a series of interfaces and abstract classes. The default in Oracle Business Rules Release 10.1.3.x was to use JAXB 1.0. In Oracle Fusion Middleware 11*g* Release 1 (11.1.1), the default is to use JAXB 2.0. Rules Designer only allows XML schemas to be imported using the JAXB 2.0. Migrated dictionaries continue to use JAXB 1.0, but the schema cannot be modified or reimported as JAXB 1.0.

A migrated dictionary which uses JAXB 1.0 objects still works with Oracle Fusion Middleware 11*g* Release 1 (11.1.1). However, JAXB 2.0 provides significant new features and many improvements for working with Oracle Business Rules.

Consider the XSD fragment that Example B–5 shows.

Example B–5   XSD Fragment

```
<element name="foo">
  <complexType>
     <sequence>
       <element name="bar" type="string"/>
       <element name="baz" type="string"/>
     </sequence>
  </complexType>
  </element>
```

This element uses an anonymous complex type to define a data structure. JAXB 1.0 generates the following classes:

- `FooType`

- `Foo extends FooType, javax.xml.bind.Element`

- `FooImpl extends FooTypeImpl implements Foo`

- `FooTypeImpl extends oracle.xml.jaxb.JaxbNode implements FooType`

The only way to create instances of these classes is the static factory methods in `ObjectFactory`.

By contrast, JAXB 2.0 generates a single annotated class, `Foo`, which can be instantiated using either `new` or an `ObjectFactory` method.

In the previous release, rules were written using the `FooType` fact type, whereas in 11*g* Release 1 (11.1.1.6.0), they are written using the `Foo` fact type.

To migrate from `FooType` to `Foo` fact types, perform the following manual migration task.

To migrate JAXB 1.0 to JAXB 2.0:

1. Use Rule Migrator to migrate.

   For more information, see Section B.2, "Using Rule Migrator Tool to Migrate an Oracle Business Rules Dictionary".

2. Delete all classes in `SCA-INF/classes`.

3. Record the aliases assigned to all of the fact types used in your rules.

   For example, `FooType` fact alias "My Foo Type".

4. Delete from the data model the XML Fact Types you want to upgrade to JAXB 2.0.

   This operation can result in a large number of validation warnings, one for every reference to the now-deleted fact types.

   However, the SDK stores references to both the ID and `String` value of all referenced elements.

   If the reference ID is set and it exists, then the `String` value is updated to reflect its current alias. So, if the alias of a fact type is changed, all of the places where that fact type is referenced get the new value because of the reference id.

   If the element with the ID is removed from the dictionary, the `String` value retains the last known name of the referenced element. This allows you to, for example:

   1. Delete a fact type with alias "FactType1"

   2. Import a new fact type.

   3. Set the alias of the new fact type to "FactType1" and all of the references to the deleted fact type automatically change to the new fact type.

5. Import the schema.

   In 11*g* Release 1 (11.1.1.6.0), JAXB 2.0 is the default.

6. Set the alias of the fact type to the previous alias.

   For example, change the alias of `Foo` to `FooType`.

7. Run validation on the dictionary.

8. If desired, set the alias of the fact type back to the original alias.

   For example, change the alias `FooType` back to `Foo`.

9. Confirm that all of the references are now be updated.

B.4.2 How to Migrate RL Functions

In the previous release, the only option for creating a Function was to write free form RL. These functions relied on generated RL names, which have changed in 11*g* Release 1 (11.1.1.6.0): the RL names are now automatically computed from the alias so they are RL-compliant, instead of relying on the user to provide a valid name.

You must re-write any function which accesses globals to comply with this change by using Actions. It is easier to write functions using Actions because they are validated at design-time.

In Oracle Business Rules Release 10.1.3.x dictionaries that contain free-form RL functions, it was a common practice to refer to Java classes which were not in the data model. For example, many dictionaries include functions which use ObjectFactory to create instances of JAXB classes. In Oracle Fusion Middleware 11*g* Release 1 (11.1.1), to include this class in the data model you cannot simply reimport the schema and import ObjectFactory because it is imported as a JAXB 2.0 class. The solution in this case is to use the ObjectFactory class which is in the "classes" directory in your project from when the dictionary was migrated. This ObjectFactory class should be imported as a Java class and can then be referenced in Oracle Fusion Middleware 11*g* Release 1 (11.1.1) functions.

For more information, see Section B.4.3, "What You May Need to Know About Manual Migration".

To migrate RL functions:

1. For each free form RL function, re-write the RL to use Actions.

   In 11*g* Release 1 (11.1.1.6.0) you may use Actions both in Function bodies and in rules. There are several new Action forms which in most cases eliminate the need for free form function bodies. Action forms available include:

   - `Call`
   - `Assert` and `Assert new`
   - `Retract`
   - `Assign` and `Assign New`
   - `Modify`
   - `Synchronized`
   - `Expression`
   - `Assert`
   - `If`, `Else If`, and `Else`
   - `For` and `While`
   - `Return`, `Throw`, `Try`, `Catch`, and `Finally`
   - `RL`

2. Validate your dictionary.

   For more information, see "Understanding Decision Table Validation" in the *Oracle Fusion Middleware User's Guide for Oracle Business Rules*.

B.4.3 What You May Need to Know About Manual Migration

In the previous release, the only option for XML schema import was JAXB 1.0. In 11*g* Release 1 (11.1.1.6.0), Oracle Business Rules provides support for JAXB 1.0, but JAXB 2.0 is the default. If you choose to continue to use JAXB 1.0, generated RL may throw `MultipleInheritanceException`. For more information, see "JAXB 1.0 Dictionaries and RL MultipleInheritanceException" in the *Oracle Fusion Middleware User's Guide for Oracle Business Rules*.

The SDK does not do any validity checking on free form RL Function bodies. As a result, any errors in the functions are not revealed until the RL is executed. Example B–6 shows what such an error looks like in a BPEL process:

Example B–6   Invalid RL Function Runtime Error

```
[2008-10-01T04:52:15.043-07:00] [server_soa] [ERROR] [] [oracle.soa.services.rules] [tid: 35]
[ecid:0000HmsG4oAEsHQ6ubV4EH18sYBt0000JA,0:2:100000051] [composite_name:ThreePattern]
[component_name: CreditRatingRules] [component_instance_
id:90558a4f-8781-4ac7-821c-49f46dd2f8fc] [composite_instance_id: 90033] <.>
Error caching the Decision Services metadata.[[
Error caching the decision services metadata for path
default/ThreePattern!1.0*c04f9ced-55ff-42fc-a6ee-f64a2055baf2/CreditRatingRules.
Check the underlying exception and correct the error. This is most likely due
to a rule modeling isssue. Validate the rule dictionary in rule designer and
fix any errors and warnings. Contact oracle support if error is not fixable.
ORABPEL-36109 Error caching the Decision Services metadata.
Error caching the decision services metadata for path
default/ThreePattern!1.0*c04f9ced-55ff-42fc-a6ee-f64a2055baf2/CreditRatingRules.
Check the underlying exception and correct the error. This is most likely due to a rule
modeling isssue. Validate the rule dictionary in rule designer and fix any errors and
warnings. Contact oracle support if error is not fixable.
        at
oracle.bpel.services.rules.impl.DecisionServiceCache.cacheDecisionServiceMetadata(DecisionSer
viceCache.java:1039)
        at
oracle.bpel.services.rules.impl.DecisionServiceCache.prepare(DecisionServiceCache.java:343)
        at
oracle.bpel.services.rules.impl.DecisionServiceImpl.preProcess(DecisionServiceImpl.java:1164)
...

Caused by: oracle.rules.rl.exceptions.UndefinedException: symbol'CreditRatingRules.rating_
param' is undefined at line 25 column 8 in main
        at
oracle.rules.rl.exceptions.ExceptionFactory.createUndefinedException(ExceptionFactory.java:38
6)
        at oracle.rules.rl.analyze.Expr.qname(Expr.java:1742)
        at oracle.rules.rl.analyze.Expr.primary(Expr.java:814)
        at oracle.rules.rl.analyze.Expr.expr(Expr.java:195)
...
```

Index