Oracle® Application Server 10*g*

Upgrading from Release 1 (1.0.2.2.x) to 10*g* (9.0.4)

10*g* (9.0.4)

Part No. B13674-02

May 2004

ORACLE®

Oracle Application Server 10*g* Upgrading from Release 1 (1.0.2.2.x) to 10*g* (9.0.4), 10*g* (9.0.4)

Part No. B13674-02

Primary Author: Julia Pond

Contributing Authors: John Bellemore, Chung Cheng, Will Chin, Saheli Dey, Mark Gizejewski, Elizabeth Hanks, Nagaraj Hegde, Clara Jaeckel, Maneesh Joshi, Shaun Lin, Jeremy Lizt, Oscar Naim, Andy Page, Vinod Kumar Pandurangan, Udayini Pendyala, Rajesh Ramachandran, Jianping Shi, Nasseer Syed, Brian Wright, Liujin Yu

Contents

3 Upgrading to OracleAS Containers for J2EE (OC4J)

4 Upgrading to OracleAS Web Cache

Send Us Your Comments

Oracle Application Server 10*g* Upgrading from Release 1 (1.0.2.2.x) to 10*g* (9.0.4), 10*g* (9.0.4)

Part No. B13674-02

Oracle welcomes your comments and suggestions on the quality and usefulness of this publication. Your input is an important part of the information used for revision.

- Did you find any errors?
- Is the information clearly presented?
- Do you need more information? If so, where?
- Are the examples correct? Do you need more examples?
- What features did you like most about this manual?

If you find any errors or have any other suggestions for improvement, please indicate the title and part number of the documentation and the chapter, section, and page number (if available). You can send comments to us in the following ways:

- Electronic mail: appserverdocs_us@oracle.com
- FAX: 650.506.7365 Attn: Oracle Application Server
- Postal service:

  Oracle Corporation
  Oracle Application Server Documentation
  500 Oracle Parkway, M/S 1op6
  Redwood Shores, California 94065
  USA

If you would like a reply, please give your name, address, telephone number, and electronic mail address (optional).

If you have problems with the software, please contact your local Oracle Support Services.

Preface

This preface describes *Upgrading from Release 1 (1.0.2.2.x) to 10g* (9.0.4), including the intended audience, structure, and conventions for the document. It also identifies related Oracle documents.

Intended Audience

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

- Oracle Application Server system administration and configuration
- The configuration and expected behavior of the system being upgraded

Documentation Accessibility

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

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

Accessibility of Code Examples in Documentation JAWS, a Windows screen reader, may not always correctly read the code examples in this document. The conventions for writing code require that closing braces should appear on an otherwise empty line; however, JAWS may not always read a line of text that consists solely of a bracket or brace.

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

Structure

Upgrading from Release 1 (1.0.2.2.x) to 10g (9.0.4) contains the following chapters:

Chapter 1, "Overview of the Oracle Application Server Upgrade"

This chapter compares Release 1 (1.0.2.2.x) installation types to 10*g* (9.0.4) installation types, identifying upgrade paths, and highlighting the usage changes in upgraded components.

Chapter 2, "Upgrading the Oracle HTTP Server"

This chapter provides instructions on upgrading the Oracle HTTP Server.

Chapter 3, "Upgrading to OracleAS Containers for J2EE (OC4J)"

This chapter provides instructions on upgrading OC4J, including specific considerations for JSPs and servlet applications.

Chapter 4, "Upgrading to OracleAS Web Cache"

This chapter provides instructions on upgrading to OracleAS Web Cache.

Chapter 5, "Upgrading to OracleAS Wireless"

This chapter provides instructions on upgrading to OracleAS Wireless.

Chapter 6, "Upgrading mod_plsql"

This chapter provides instructions on upgrading the Database Access Descriptor (DAD) configuration file in mod_plsql.

Chapter 7, "Upgrading to OracleAS Discoverer"

This chapter provides instructions on upgrading to OracleAS Discoverer.

Chapter 8, "Upgrading to OracleAS Forms Services"

This chapter provides instructions on upgrading to OracleAS Forms Services.

Chapter 9, "Upgrading to OracleAS Reports Services"

This chapter provides instructions on upgrading to OracleAS Reports Services.

Chapter 10, "Upgrading OracleAS InterConnect"

This chapter provides instructions on upgrading OracleAS InterConnect.

Chapter 11, "Upgrading to Oracle Workflow"

This chapter provides instructions on upgrading Oracle Workflow.

Chapter 12, "Post-Upgrade Considerations"

This chapter discusses post-upgrade testing and decommissioning of the source Oracle home.

Related Documents

The following Oracle publications are referenced in this guide. They contain information that may be useful when performing an upgrade:

- *Oracle Application Server 10g Administrator's Guide*
- *Oracle Application Server 10g Concepts*
- *Oracle Application Server 10g Security Guide*
- *Oracle Application Server 10g High Availability Guide*
- *Oracle HTTP Server Administrator's Guide*
- *Oracle Application Server Web Cache Administrator's Guide*
- *Oracle Application Server Containers for J2EE User's Guide*
- *Oracle Process Manager and Notification Server Administrator's Guide*
- *Oracle Application Server Forms Services Deployment Guide*
- *Oracle Application Server 10g Installation Guide*
- *Oracle Application Server Discoverer Configuration Guide*
- *Oracle Application Server Wireless Developer's Guide*
- *Oracle Application Server Single Sign-On Administrator's Guide*
- *Oracle Internet Directory Administrator's Guide*
- *Distributed Configuration Management Reference Guide*
- *Oracle Application Server 10g Release Notes*
- *Oracle9i Database Administrator's Guide*
- *Step-By-Step Guide to Install Portal 9.0.2.3 in a Custom Database (document ID 238516.1)* This document is located at `http://metalink.oracle.com`.
- *Oracle Application Server Portal Error Messages Guide*
- *Oracle9iAS Portal Configuration Guide*
- *Oracle Application Server Portal User's Guide*
- *Oracle Application Server Web Cache Administrator's Guide*
- *Boosting Performance in Web Cache white paper*
- *How do I add additional Portal and Single Sign-On Targets to the Enterprise Manager Web Site technical note*

Conventions

This guide uses the following conventions:

Convention	Meaning
. . .	Horizontal ellipsis points in statements or commands mean that parts of the statement or command not directly related to the example have been omitted
`monospace text`	File names, path names, command names, code, URLs
`monospace bold text`	Typed user input.
`monospace italic text`	Variables in text or code.
`<source_MT_OH>`	The full path to the Release 1 (1.0.2.2.x) middle tier Oracle home.
`<destination_MT_OH>`	The full path to the 10*g* (9.0.4) middle tier Oracle home.
`<source_Infra_OH>`	The full path to the Release 1 (1.0.2.2.x) Infrastructure Oracle home.
`<destination_Infra_OH>`	The full path to the 10*g* (9.0.4) Infrastructure (Identity Management or Metadata Repository) Oracle home.
`/`	Directory path delimiter. This delimiter is used in UNIX and Windows directory paths.

1

Overview of the Oracle Application Server Upgrade

This chapter introduces the components and functionality involved in the upgrade from Oracle9*i*AS Release 1 (1.0.2.2.x) to Oracle Application Server 10*g* (9.0.4), and highlights some usage changes in components in the new release.

It contains these sections:

1.1 The Oracle9*i*AS Release 1 (1.0.2.2.x) to Oracle Application Server 10*g* (9.0.4) Upgrade Process

This section describes the upgrade process, beginning with the installation of Oracle Application Server 10*g* (9.0.4), and discusses alternatives to upgrading manually.

1.1.1 Steps in the Upgrade Process

Upgrading to Oracle Application Server 10*g* (9.0.4) consists of these steps:

1. If you plan to use Infrastructure services, install the Oracle Application Server 10*g* (9.0.4) Infrastructure installation type.
2. Install the Oracle Application Server 10*g* (9.0.4) middle tier of your choice (J2EE & Web Cache, Portal and Wireless, Business Intelligence and Forms).

   Important considerations for installation and upgrade:

   - Review carefully the *Oracle Application Server 10g Installation Guide*, Chapter 3, "Compatibility with Other Versions", and Chapter 4, "Requirements" to ensure that all requirements are met for your configuration.
   - If you are installing a Portal and Wireless middle tier, and intend to upgrade Oracle9*i*AS Wireless to Oracle Application Server Wireless, perform the installation within the sequence of steps in Section 5.3, "Preliminary OracleAS Wireless Upgrade Tasks" on page 5-2. Those steps recommend backups of the middle tier and Infrastructure installations at certain junctures. *In particular, do not configure the Wireless component during the installation.*

3. Perform the upgrade steps in this guide for the components you want to use in Oracle Application Server 10*g* (9.0.4).

1.1.2 An Alternative to Manual Upgrade

The procedure presented in this guide for upgrading middle tier instances from Oracle9*i*AS Release 1 (1.0.2.2.x) to Oracle Application Server 10*g* (9.0.4) is manual, with the exception of a few scripts for certain components. However, if you are upgrading from an Oracle9*i*AS Release 1 (1.0.2.2.x) Core installation to a Oracle Application Server 10*g* (9.0.4) J2EE and Web Cache installation type, it might be worthwhile to install the intermediate release, Oracle9*i*AS Release 2 (9.0.2), and use the Migration Assistant (provided in Oracle9*i*AS Release 2 (9.0.2)) and the Upgrade Assistant (provided in 10*g* (9.0.4)) to automate part of the upgrade. The majority of the configuration for Oracle HTTP Server, Oracle9*i*AS Containers for J2EE, and Oracle9*i*AS Web Cache is automated by the Assistants.

This alternative is probably worthy of consideration if you have an extensively customized Oracle HTTP Server, since the `httpd.conf` file is large (and thus time-consuming and error-prone to upgrade manually).

All upgrade and installation documentation from prior releases is available on Oracle Technology Network. The upgrade documentation, *Migrating from Oracle9iAS Release 1 (1.0.2.2.x) to Release 2 (9.0.2)* and *Upgrading to 10g (9.0.4)* describes in detail the processing performed by the Migration Assistant and the Upgrade Assistant. Reviewing this documentation should inform your choice of upgrade method.

Before performing any of the installations, review the installation documentation for the release you are installing to ensure that coexistence and other requirements for your particular configuration are met.

The steps in the partially automated upgrade process are as follows:

1. Install the Oracle9*i*AS Release 2 (9.0.2) middle tier on the same computer as the Oracle9*i*AS Release 1 (1.0.2.2.x) middle tier.
2. Follow the instructions in *Migrating from Oracle9iAS Release 1 (1.0.2.2.x) to Release 2 (9.0.2)* in the Release 2 (9.0.2) documentation library. The Oracle9*i*AS Migration Assistant, described in the guide, automates much of the Oracle HTTP Server, Oracle9*i*AS Containers for J2EE, and Oracle9*i*AS Web Cache upgrades.
3. After using the Migration Assistant, perform any manual tasks necessary to complete the upgrade of these components. The tasks are documented in Migrating from Oracle9iAS Release 1 (1.0.2.2.x) to Release 2 (9.0.2)
4. If you plan to use Infrastructure services in 10*g* (9.0.4), install the Oracle Application Server 10*g* (9.0.4) Infrastructure.
5. Install the Oracle Application Server 10*g* (9.0.4) middle tier on the same computer as the Oracle9*i*AS Release 2 (9.0.2) middle tier.
6. Follow the instructions in *Upgrading to 10g (9.0.4)* in the 10*g* (9.0.4) documentation library. The OracleAS Upgrade Assistant, described in the guide, automates much of the Oracle HTTP Server, Oracle9*i*AS Containers for J2EE, and Oracle9*i*AS Web Cache upgrades.
7. After using the Upgrade Assistant, perform any manual upgrade tasks necessary to complete the upgrade of each component. The tasks are documented in *Upgrading to 10g (9.0.4)* in the Oracle Application Server 10*g* (9.0.4) documentation library.

1.2 Component Usage and Functionality Changes Between Releases

Some components are common to Oracle9*i*AS Release 1 (1.0.2.2.x) and the Oracle Application Server 10*g* (9.0.4); others have been deprecated or renamed. As a prelude to the upgrade discussion, it is useful to compare the two releases to identify usage changes and highlight benefits of upgrading. Table 1–1 identifies the Oracle9*i*AS Release 1 (1.0.2.2.x) components and summarizes their disposition in Oracle Application Server 10*g* (9.0.4), providing a reference to the information about the changes to the component in the new release.

Table 1–1 Components in Release 1 (1.0.2.2.x) and 10g (9.0.4)

Component in Release 1 (1.0.2.2.x)	Component in 10*g* (9.0.4)	Description	For more information, see:
Oracle HTTP Server (based on the Apache HTTP Server v. 1.3) Available in Core, Minimal, Standard and Enterprise Editions	Oracle HTTP Server (based on the Apache HTTP Server v. 1.3.28) Available in J2EE & Web Cache, Portal & Wireless, and Business Intelligence and Forms Editions	Oracle HTTP Server is the web server component of the Oracle Application Server that services HTTP and HTTPS requests from the Internet.	Section 1.2.1, "Changes in the Oracle HTTP Server" on page 1-5 and `http://httpd.apache.org/docs/new_features_1_3.html`
Oracle9*i*AS Containers for J2EE (OC4J) Available in Core Edition	Oracle Application Server Containers for J2EE (OC4J) Available in J2EE & Web Cache, Portal & Wireless, and Business Intelligence and Forms Editions	OC4J is the J2EE container for Oracle Application Server. It was introduced in Oracle9iAS Release 1 (1.0.2.2).	Section 1.2.2, "Changes in Oracle Application Server Containers for J2EE (OC4J)" on page 1-6
Oracle9*i*AS Web Cache Available in Core Edition	Oracle Application Server Web Cache Available in J2EE & Web Cache, Portal & Wireless, and Business Intelligence and Forms Editions	Oracle Application Server Web Cache is a content-aware server accelerator, or reverse proxy server, that improves the performance, scalability, and availability of Web sites that run on Oracle Application Server.	Section 1.2.3, "Changes in Oracle Application Server Web Cache" on page 1-6
Oracle9*i*AS Single Sign-On	Oracle Application Server Single Sign-On	OracleAS Single Sign-On is a component of Oracle Application Server that provides a framework for secure single sign-on; allowing users to log in to multiple Web-based applications, such as expense reporting, e-mail, and benefits information, using a single username and password.	Section 1.4, "Accessing OracleAS Single Sign-On Upgrade Documentation" on page 1-10.
Oracle9*i*AS Portal	Oracle Application Server Portal Available in Portal & Wireless and Business Intelligence and Forms Editions	OracleAS Portal is a Web-based application for building and deploying portals. It provides a secure, manageable environment for accessing and interacting with enterprise software services and information resources.	Section 1.5, "Accessing OracleAS Portal Upgrade Documentation" on page 1-11.

Table 1–1 (Cont.) Components in Release 1 (1.0.2.2.x) and 10g (9.0.4)

Component in Release 1 (1.0.2.2.x)	Component in 10*g* (9.0.4)	Description	For more information, see:
Oracle9*i*AS Wireless	Oracle Application Server Wireless Available in Portal & Wireless and Business Intelligence and Forms Editions	OracleAS Wireless is the wireless and voice platform of Oracle Application Server. It enables development and deployment of wireless browsing applications, voice applications, asynchronous applications, J2ME applications, and notifications. Wireless is also the voice and wireless platform for the mobile extensions of the Oracle E-Business Suite, the wireless and voice components of the Oracle Collaboration Suite, and the foundation for custom and partner wireless and voice solutions.	
Oracle Enterprise Manager Client	Oracle Enterprise Manager Application Server Control in Oracle Enterprise Manager	The Oracle Enterprise Manager Client is the first tier in the three-tiered architecture of the Oracle Enterprise Manager. The second tier is the Oracle Management Server, and the third consists of managed nodes containing databases and other services. The Oracle Enterprise Manager Application Server Control in Oracle Enterprise Manager is the graphical interface for managing and monitoring application server components.	
Oracle Enterprise Java Engine	This functionality replaced by Oracle Application Server Containers for J2EE.	Oracle Enterprise Java Engine is a server environment that supports Enterprise Java Beans, CORBA, and database stored procedures.	
Oracle9iAS Discoverer	Oracle Application Server Discoverer Available in Business Intelligence and Forms Editions	OracleAS Discoverer is an ad-hoc query analysis tool. Since Oracle9iAS Release 2 (9.0.2), it has integrated with Oracle Enterprise Manager to perform various functions supported by Enterprise Manager and Oracle Portal. OracleAS Discoverer is available as a deployed application in the OC4J BI FORMS stack. In Oracle Application Server 10*g* (9.0.4), Discoverer services are managed by Oracle Process Manager and Notification Server (OPMN).	Section 1.2.4, "Changes in Oracle Application Server Discoverer" on page 1-6

1.2.1 Changes in the Oracle HTTP Server

The Oracle HTTP Server has undergone significant change since Release 1 (1.0.2.2.x). In addition to the changes in the base product (Apache HTTP Server version 1. 3 to 1.3.28), the configuration files have been restructured to conform to the OHS SMI requirements, the `mod_ssl` module has been replaced with a new Oracle module, `mod_ossl`, new Oracle modules have been added and out-of-date modules removed, and SSL certificates have been replaced with Oracle wallets.

The configuration of OHS is very sophisticated because, in addition to the general server configuration directives, each module has its own set of directives. In addition, virtual hosts can be set up, each of which has its own complete set of configuration directives. All of this is customizable, so upgrade is a complicated task.

Specific changes to the Oracle HTTP Server main configuration file and its widely used modules are summarized in Table 1–2.

Table 1–2 Changes in the Oracle HTTP Server Functionality and Usage

Area of Change	Release 1 (1.0.2.2.x) Usage/Behavior	10*g* (9.0.4) Usage/Behavior
`mod_jserv`, JServ servlet container	These components enabled users to run servlets and Java Server Pages.	OC4J is the J2EE-compliant servlet engine. `mod_jserv`/Jserv is disabled by default. All applications using JServ should be upgraded to OC4J. See: Section 3.1.2, "Upgrading from JServ to the OC4J 10g (9.0.4) Servlet Environment" on page 3-16
mod_oprocmgr	This module managed the start/stop/restart and monitoring of JServ processes.	Oracle Process Manager and Notification Server (OPMN) performs the equivalent functionality in managing OC4J processes. `mod_oprocmgr` is not loaded by the LoadModule directive in `httpd.conf`, and all configuration related to it is ignored in server operations.
mod_ssl	This module is based on the OpenSSL toolkit.	`mod_ssl` is replaced by `mod_ossl`, which is based on the Oracle security stack. Secure Sockets Layer (SSL) certificates must be converted to Oracle wallets.
`mod_onsint`	Not present.	This module communicates with OPMN, facilitating its management of OHS.
`mod_wchandshake`	Not present.	This module performs the handshake between OracleAS Web Cache and the Oracle HTTP Server.
`httpd.conf`	Main server configuration file, with some Oracle-specific customizations.	The following changes have been made to this file: ▪ Configuration related to the Dynamic Monitoring Service (DMS) has been relocated to a separate file, `<destination_MT_OH>/Apache/Apache/conf/dms.conf`. ▪ Configuration related to SSL has been moved to a separate file, `<destination_MT_OH>/Apache/Apache/conf/mod_ssl.conf`. ▪ `mod_rewrite` has been relocated to the end of the `httpd.conf` file. ▪ `mod_ossl` is always loaded; it is no longer surrounded by `IfDefine` directives. ▪ Module-specific directives (other than those defined by standard modules) are surrounded by `IfModule` directives.

1.2.2 Changes in Oracle Application Server Containers for J2EE (OC4J)

The differences between the OC4J container in Oracle Application Server 10*g* (9.0.4) and the OC4J container shipped with Oracle9*i*AS Release 1 (1.0.2.2.x) are summarized in Table 1–3.

Table 1–3 Changes in the OC4J Functionality and Usage

Area of Change	Oracle9*i*AS Release 1 (1.0.2.2.x)	Oracle Application Server 10*g* (9.0.4)

Integration	Minimal ingegration with Oracle9*i*AS; no capability to use Distributed Configuration Management (DCM) to communicate with or manage OC4J.	OC4J is integrated with DCM. Among the many configuration management commands provided by the `dcmctl` utility are the following OC4J-specific commands:
		`deployApplication`
		`redeployApplication`
		`undeployApplication`
		`validateEarFile`
		In addition, OC4J and other components' configuration changes made in the local instance are propagated to the DCM repository with the command:
		`updateConfig`
		If you use Enterprise Manager to make configuration changes, you do not need to issue this command (Enterprise Manager will invoke it). However, if you make configuration changes by editing files directly, you must use this command to keep configurations in sync.
		For more information about DCM, see the *Distributed Configuration Management Reference Guide*.

Start-up	Manual or scripted	Started by Oracle Process Manager and Notification Server (OPMN). (Any application-specific start-up commands used in Oracle9*i*AS Release 1 (1.0.2.2.x), and needed in Oracle Application Server 10*g* (9.0.4) will need to be manually upgraded.)
Listener	The OC4J HTTP listener was used.	OHS is the default Web listener.
Load-balancing configuration	The `loadbalancer.jar` file may have been used for load balancing.	`mod_oc4j` in OHS routes requests to OC4J.

1.2.3 Changes in Oracle Application Server Web Cache

The differences between the Oracle9*i*AS Web Cache and OracleAS Web Cache are in the areas of:

- DTDs
- Names of elements in configuration files

1.2.4 Changes in Oracle Application Server Discoverer

The differences between the Oracle9*i*AS Discoverer and OracleAS Discoverer are in the areas of the Discoverer client and the Discoverer C++ server.

- In Oracle9*i*AS Release 1 (1.0.2.2.x), the Discoverer client was plugged into the Oracle HTTP Server configuration files. In Oracle Application Server 10*g* (9.0.4), it is a deployed application in the OC4J Business Intelligence and Forms stack.
- In Oracle9*i*AS Release 1 (1.0.2.2.x), the Discoverer C++ server was self-controllable via Visibroker Gatekeeper. In Oracle Application Server 10*g* (9.0.4), it is OPMN-controllable via Visibroker Gatekeeper.

1.2.5 Changes in OracleAS Forms Services

The differences between the Oracle9*i*AS Forms Services and OracleAS Forms Services are in the areas of servlet container configuration, method of configuration, and deployment methods available.

- In Oracle9*i*AS Release 1 (1.0.2.2.x), In Oracle9*i*AS Release 1 (1.0.2.2.x), Forms Services may have required manual configuration of the Oracle HTTP Server and JServ servlet container files. The Oracle Application Server 10*g* (9.0.4) installer automatically configures the Oracle HTTP Server and the OC4J servlet container for Forms Services. If additional customization of these is required, it can be done with the graphical user interface of Oracle Enterprise Manager. In addition, key OracleAS Forms Services configuration files (`formsweb.cfg`, `default.env`) can also be edited using Oracle Enterprise Manager.
- In Oracle Application Server 10*g* (9.0.4), OracleAS Forms Services uses the Forms Listener Servlet architecture for deployment of Forms applications. The following deployment modes are not available: character mode; client/server; the Forms CGI; and the Forms Server (also known as the Forms Listener).

1.2.6 Changes in OracleAS Reports

The differences between Oracle Reports and OracleAS Reports are itemized in Table 1–4. The changes between the releases are also discussed thoroughly in:

Oracle9i Reports Obsolescense Plan: Oracle9i Reports Statement of Direction at:

`http://otn.oracle.com/products/reports/pdf/movingto9i.pdf`

Statement of Direction: Oracle Application Server Reports Services 10g (9.0.4) at:

`http://otn.oracle.com/products/reports/10g/SOD_Reports.html`

Table 1–4 Changes in the OracleAS Reports Functionality and Usage

Area of Change	Release 1 (1.0.2.2.x) Usage/Behavior	10*g* (9.0.4) Usage/Behavior
Reports Server architecture	See: `http://download.oracle.com/docs/html/A73173_01/output/pbr_arch.htm#1003225`	Scalable, flexible architecture for the distribution and automated management of report generation engines on the same server and across multiple servers. For more information, see: `http://download.oracle.com/docs/html/B10314_01/pbr_arch.htm#1005635`
Web client choices	Servlet and CGI implementation	Servlet, JSP, or CGI implementation. For more information, see: `http://download.oracle.com/docs/html/B10314_01/pbr_arch.htm#1005928`
High availability	No high availability features	Numerous high availability features keep the middle tier running even when particular servers or components fail. For more information, see: `http://download.oracle.com/docs/html/B10314_01/pbr_arch.htm#1006314`
Starting and stopping	Command line or Windows service. See: `http://download.oracle.com/docs/html/A73173_01/output/pbr_case.htm#1017604`	Oracle Process Manager and Notification Server (OPMN). See: `http://download.oracle.com/docs/html/B10314_01/pbr_strt.htm#1005573`

Table 1–4 (Cont.) Changes in the OracleAS Reports Functionality and Usage

Area of Change	Release 1 (1.0.2.2.x) Usage/Behavior	10*g* (9.0.4) Usage/Behavior
Reports Services configuration	Reports server configuration: http://download.oracle.com/docs/html/A73173_01/output/pbr_case.htm#1006332 Reports web client configuration: http://download.oracle.com/docs/html/A73173_01/output/pbr_case.htm#1007896	All configuration files are in XML format. See: http://download.oracle.com/docs/html/B10314_01/pbr_conf.htm#1004903
Reports destination configuration	No destination configuration available	Supports default and custom destinations. See: http://download.oracle.com/docs/html/B10314_01/pbr_dest.htm#1004903
Security	Provides functionality for: ▪ Application security: access control for the reports application from which reports are launched ▪ Resource security: access control for reports servers, printers, calendars, and reports See: http://download.oracle.com/docs/html/A73173_01/output/booktoc.htm	Provides functionality for application server level (single sign-on or non-single-sign-on) access to reports and data: ▪ Application security: access control for the reports application from which reports are launched ▪ Resource security: access control for reports servers, printers, calendars, and reports ▪ Data source security: access control for a particular database See: Securing OracleAS Reports http://download.oracle.com/docs/html/B10314_01/pbr_sec_arch.htm#1004903 Configuring and Administering OracleAS Single Sign-On http://download.oracle.com/docs/html/B10314_01/pbr_sso.htm#1005576 Deploying Reports in OracleAS Portal http://download.oracle.com/docs/html/B10314_01/pbr_portal.htm#1008280
Configuring Reports server clusters	Master-slave clustering architecture. See: http://download.oracle.com/docs/html/A73173_01/output/pbr_clus.htm#1656	Peer-level clustering support. See: http://download.oracle.com/docs/html/B10314_01/pbr_clus.htm#1008605
Deploying and running report requests	Mechanism to deploy and run paper-based reports. See: http://download.oracle.com/docs/html/A73173_01/output/pbr_run.htm#1008829	Mechanism to deploy paper and Web reports. Support is also provided to run these reports through pluggable engines. Usage of key-map file is changed. See: http://download.oracle.com/docs/html/B10314_01/pbr_run.htm#1009713
Batch registration of reports in OracleAS Portal	Unavailable	See: http://download.oracle.com/docs/html/B10314_01/pbr_bat.htm
Oracle Reports Web Service	Unavailable	Web Services interface through which external applications can submit a job. See: http://download.oracle.com/docs/html/B10314_01/pbr_webservice.htm#1010102

Table 1–4 (Cont.) Changes in the OracleAS Reports Functionality and Usage

Area of Change	Release 1 (1.0.2.2.x) Usage/Behavior	10*g* (9.0.4) Usage/Behavior
Advance distributions	Builder distribution	Advance distribution and bursting capabilities, which can also be used with customer destinations. See: Advance Distribution http://download.oracle.com/docs/html/B10314_01/pbr_dist.htm#1007339 Bursting and Distributing a Report http://otn.oracle.com/products/reports/htdocs/getstart/examples/distribution/doc/orbr_dist.pdf
Event-driven publishing	Unavailable	Reports can be run automatically in response to an event in the database, such as the insertion of a record or the change of a value. See: http://download.oracle.com/docs/html/B10314_01/pbr_evnt.htm#1006610
DISPLAY environment variable dependency (UNIX)	DISPLAY must be set on UNIX systems	Not necessary to have the DISPLAY environment variable or a valid printer defined at runtime. See: http://download.oracle.com/docs/html/B10314_01/pbr_conf.htm#1010014
Managing and monitoring OracleAS Reports Services	Unavailable	Oracle Enterprise Manager provides managing and monitoring services for OracleAS Reports Services. See: http://download.oracle.com/docs/html/B10314_01/pbr_oem.htm#1008796
Reports Web Cartridge	See: http://download.oracle.com/docs/html/A73173_01/output/pbr_wcap.htm#630125	Reports Web Cartridge
Configuring and using the pluggable data source	Unavailable	Pluggable data sources are supported. See: JDBC Data Source Example http://download.oracle.com/docs/html/B10314_01/pbr_jdbcpds.htm#1007084 XML Data Source Example http://download.oracle.com/docs/html/B10602_01/orbr_xmlpds.htm#1005598 Text Data Source Example http://download.oracle.com/docs/html/B10602_01/orbr_text.htm#1005598
Report using Oracle Express Data	Unavailable	OLAP-based built-in data source. See: http://otn.oracle.com/products/reports/pluginxchange/plugins/datasource/olap/olap_pds_release_notes.html
Printing on UNIX with Oracle Reports		OracleAS Reports provides a rich set of features out of the box for printing on various platforms. See: http://download.oracle.com/docs/html/B10314_01/pbr_uxprt.htm#1007271

Table 1–4 (Cont.) Changes in the OracleAS Reports Functionality and Usage

Area of Change	Release 1 (1.0.2.2.x) Usage/Behavior	10*g* (9.0.4) Usage/Behavior
PDF in Oracle Reports	Supported	Supports PDF 1.4, and is capable of generating high resolution PDF reports on all platforms. The PDF features supported by Oracle Reports include: ▪ Compression ▪ Font features ▪ Accessibility ▪ Taxonomy ▪ Enhanced graph support See: http://download.oracle.com/docs/html/B10314_01/pbr_pdf.htm#1005713
Oracle9*i* JDeveloper integration	Unavailable	
Command-line options and report-related environment variables	See: http://download.oracle.com/docs/html/A73173_01/output/partpage.htm#435787	See: http://download.oracle.com/docs/html/B10314_01/part5.htm#435787
JSP Web-based reports	Unavailable	See the tutorial at: http://download.oracle.com/docs/html/B10612_01/toc.htm

1.3 Using This Guide

Depending on the Release 1 (1.0.2.2.x) installation type you are upgrading, or the components you will use in the upgraded instance, you may use some or all of the chapters in this guide. Review, understand, and perform the steps in Section 1.1, "The Oracle9iAS Release 1 (1.0.2.2.x) to Oracle Application Server 10g (9.0.4) Upgrade Process" on page 1-1, paying particular attention to installation requirements that apply to the configuration you are upgrading. The *Oracle Application Server 10g Installation Guide*, used in conjunction with this guide, provides complete information on compatibility and installation requirements.

References to other Oracle documentation may be necessary as you use this guide (lengthy configuration instructions for components are not replicated here). Ensure that you have access to all of the books in Related Documents. All of the documents are available on the Oracle Technology Network (OTN).

1.4 Accessing OracleAS Single Sign-On Upgrade Documentation

The upgrade documentation for OracleAS Single Sign-On is not included in this guide. It is provided in the OracleAS Single Sign-On Upgrade Patch, which is available on Oracle MetaLink (you need a MetaLink user ID and password to obtain the patch). Follow these instructions to download the patch:

1. Log in to Oracle MetaLink at:

   http://metalink.oracle.com

2. Locate and download patch 3498669.

3. Follow the instructions in the `README.TXT` file and the *OracleAS Single Sign-On Upgrade Guide* (included in the patch).

1.5 Accessing OracleAS Portal Upgrade Documentation

The upgrade documentation for OracleAS Portal is not included in this guide. It is provided in the Portal Center on Oracle Technology Network. Follow these instructions to access the OracleAS Portal upgrade documentation:

1. Access the URL:

   `http://portalcenter.oracle.com/upgrades`

2. Click the **9.0.4** link in the **Upgrade Targets** section.

   The 9.0.4 information page appears.

3. Read these sections:

   - **Generic Upgrade Information**
   - **Version 3.0.9 to 9.0.4**

2

Upgrading the Oracle HTTP Server

This chapter contains step-by-step instructions for upgrading the Oracle HTTP Server. It contains the following sections:

2.1 Oracle HTTP Server Upgrade Items

An upgrade item is an aspect of configuration that is acted upon during upgrade: a file, executable, or a setting that you must add, change, or delete/replace in the 10g (9.0.4) installation. Oracle HTTP Server upgrade items are described in Table 2–1.

Table 2–1 Oracle HTTP Server Upgrade Items

Upgrade Item	Location in Release 1 (1.0.2.2.x) Oracle home	Location in 10*g* (9.0.4) Oracle home	Notes
`httpd.conf`	`Apache/Apache/conf`	`Apache/Apache/conf`	`httpd.conf` is the main configuration file for OHS. If this file is not located in its default location, `<source_MT_OH>/Apache/Apache/conf` in Oracle9*i*AS Release 1 (1.0.2.2.x) and you want it to be in a non-default location in 9.0.4, you must also modify the `opmn.xml` file (OPMN's configuration file) to reflect the non-default location.
User-defined configuration files			User-defined configuration files are the files defined recursively by `Include` directives in `httpd.conf`. Since `mod_jserv` and the JServ servlet engine have been replaced by `mod_oc4j` and OC4J, upgrading JServ is not discussed here. The upgrade of component configuration files defined in or referenced by `Include` directives in `oracle_apache.conf` is not discussed in this chapter.
User-defined modules			User-defined modules are modules other than those listed in the default set of modules in "Modules Shipped in Release 1 (1.0.2.2)".

Table 2–1 (Cont.) Oracle HTTP Server Upgrade Items

Upgrade Item	Location in Release 1 (1.0.2.2.x) Oracle home	Location in 10*g* (9.0.4) Oracle home	Notes
`mod_ssl`			This module was replaced with mod_ossl in Oracle9iAS Release 2 (9.0.2). Therefore, to upgrade to Oracle Application Server 10*g* (9.0.4), you must replace the SSL certificates with Oracle wallets. If you use SSL connections, you should upgrade SSL certificates to Oracle wallets. For instructions on how to create Oracle wallets, see: *Oracle Application Server 10g Security Guide*
cgi/fastcgi programs			You must locate all programs referenced in the configuration files and copy them to the corresponding locations in the 10*g* (9.0.4) Oracle home.
Static documents			You must locate all static documents and copy them to the corresponding locations in the 10*g* (9.0.4) Oracle home.

Modules Shipped in Release 1 (1.0.2.2)

- mmap_static_module
- vhost_alias_module
- env_module
- agent_log_module
- referer_log_module
- mime_magic_module
- mime_module
- negotiation_module
- status_module
- info_module
- includes_module
- autoindex_module
- dir_module
- cgi_module
- asis_module
- imap_module
- action_module
- speling_module
- userdir_module
- alias_module
- rewrite_module
- access_module
- auth_module

- anon_auth_module
- dbm_auth_module
- digest_module
- proxy_module
- cern_meta_module
- expires_module
- headers_module
- usertrack_module
- example_module
- unique_id_module
- setenvif_module
- define_module
- dms_module
- perl_module
- fastcgi_module
- ssl_module
- oprocmgr_module

2.2 Components Related to the Oracle HTTP Server

This section discusses the Oracle HTTP Server relationship to other components and describes the configuration tasks necessary to make the components work together.

2.2.1 Oracle Process Management and Notification

The Oracle HTTP Server is managed by the Oracle Process Management and Notification (OPMN) server. You must place the entry shown below in bold text in the `opmn.xml` file:

```
<ias-component id="HTTP_Server" status="enabled" id-matching="false">
 <process-type id="HTTP_Server" module-id="OHS">
    <process-set id="HTTP_Server" restart-on-death="true" numprocs="1">
      <module-data>
       <category id="start-parameters">
          <data id="config-file" value="/myconfs/httpd.conf"/>
          <data id="start-mode" value="ssl-enabled"/>
        </category>
...
```

The `"config-file"` module-data directive in the first line of the OPMN entry specifies the location of the `httpd.conf` file. If is not in the default location after upgrade, enter its full path as the value for the `"config-file"` directive. If the Oracle HTTP Server uses an SSL connection, set the `"start-mode"` module-data directive value to `"ssl-enabled"`.

2.2.2 Oracle Application Server Web Cache

If Oracle Application Server Web Cache is configured as the first listener, ensure that the Oracle HTTP Server directives listed in Table 2–2 have the same values as the corresponding OracleAS Web Cache elements. In particular, note that the Oracle HTTP Server Port directive specifies the port number of a front-end load balancer or reverse proxy. Thus, if Oracle Application Server Web Cache is used, then the Oracle HTTP Server Port directive should have the value of the port on which OracleAS Web Cache is listening.

Table 2–2 Oracle HTTP Server and Oracle Application Server Web Cache Port Settings

Oracle HTTP Server Directive	Oracle Application Server Web Cache Element
VirtualHost	Site definitions
Listen	Origin server ports
VirtualHost, Listen	Site-to-server mappings
Port	Listen

2.3 Oracle HTTP Server Upgrade Tasks

This section provides instructions for upgrading each item identified in Section 2.1, "Oracle HTTP Server Upgrade Items". It is divided into the following subsections:

2.3.1 Upgrading the httpd.conf File

Since the file is quite large, it is useful to begin with an exhaustive listing of the changes to the file since installation, and a general idea of the customizations represented. Use a file comparison utility to compare:

`<source_MT_OH>/Apache/Apache/conf/httpd.conf`

to

`<source_MT_OH>/Apache/Apache/conf/httpd.conf.default`

The results of the comparison will identify all changes made to the file since installation. These changes to the file represent two types of customizations:

- Changes made to default directive settings throughout the file (which were initially set by the installer). To upgrade these, you must modify the directives throughout `<destination_MT_OH>/Apache/Apache/conf/httpd.conf` to reflect the same settings. Table 2–3 shows the changes to default settings.
- Directives added (typically, appended to the end of the file). To upgrade these, you must copy them to the end of the `<destination_MT_`

OH>/Apache/Apache/conf/httpd.conf file, replacing any path values to reflect the new Oracle home location.

Table 2–3 Changes to Default Oracle HTTP Server Directives and Settings

Upgrade Item	Release 1 (1.0.2.2.x) Setting	10*g* (9.0.4) Setting	Status	Remarks
oprocmgr (process manager for JServ)	LoadModule oprocmgr_module libexec/liboprocmgr.so	None	Removed	OPMN is the process manager for OC4J; JServ no longer used
SSL	<IfDefine SSL> LoadModule ssl_module libexec/mod_ssl.so </IfDefine>	LoadModule ossl_module libexec/mod_ossl.so	Changed	mod_ossl replaced mod_ssl. In 10*g* (9.0.4), mod_ossl is always loaded.
SSL Configuration			Relocated and changed	
Dynamic Monitoring Service (DMS)	LoadModule dms_module libexec/mod_dms.so	Not applicable	Relocated	In 10*g* (9.0.4), DMS has its own configuration file, dms.conf. The LoadModule directive for mod_dms has been moved to that file.
mod_rewrite	LoadModule rewrite_module libexec/mod_rewrite.so	Same	Relocated	To avoid the conflict of mod_osso and mod_oc4j to process the request, this module's location has to be moved to the end.
MaxSpareServers directive	10	20	Value changed	On UNIX systems, the maximum number of child processes that are not handling a request. This directive has no effect on Windows systems.
ErrorLog				If this directive specifies a new location for the log file, ensure that the specified directory structure is present. If it is not, Oracle HTTP Server will not start.
CustomLog				If this directive specifies a new location for the log file, ensure that the specified directory structure is present. If it is not, Oracle HTTP Server will not start.
<Directory/>	Options directive in this container does not have MultiViews set	Options directive in this container has MultiViews set		
Alias /javacachedocs	This directive not present		Added	
Alias /perl/	Not in an <IfModule> container	<IfModule mod_perl.c> Alias /perl/ </IfModule>	Changed	
Alias /soapdocs/	This directive is used	This directive is not used	Removed	
<Location /server-status> </Location>			The Allow directive in this <Location> container has changed.	
SetEnv Perl			Relocated	

Table 2–3 (Cont.) Changes to Default Oracle HTTP Server Directives and Settings

Upgrade Item	Release 1 (1.0.2.2.x) Setting	10*g* (9.0.4) Setting	Status	Remarks
`DirectoryMatch` for WEB_INF directory		`<DirectoryMatch /WEB_INF/>`	Added	

`IfModule mod_fastcgi.c>`
...
...
SSLOptions
+StdEnvVars
...
...
`</IfModule>`
	`<IfDefine SSL>`
	...
	SSLOptions
	...
	`</IfDefine>`
		`<IfModule mod_ossl.c>`
		...
		SSLOptions
		...
		`</IfModule>`
			Changed	

`PassEnv` directives	No `PassEnv` directives used	`PassEnv ORACLE_HOME`
		`PassEnv LIBRARY_PATH`
		`PassEnv NLS_LANG`
		`PassEnv TNS_ADMIN`
			Added	

2.3.2 Upgrading Port and Listen Directives

In a production environment, you will not want port assignments to change as part of the upgrade. The installer assigns a new set of ports for the Port and Listen directives. When you upgrade, you want to retain the ports from the previous installation, discarding those assigned during installation of 10*g* (9.0.4).

Beginning with 10*g* (9.0.4), the installer no longer assigns port 80 for http requests and port 443 for https requests, since it is common to use a load balancer such as OracleAS Web Cache in front of the Oracle HTTP Server. So, in practice, Oracle HTTP Server rarely listens on ports 80 and 443.

Configure the port directives as required by your environment, with the following points in mind:

- If you choose to install OracleAS Web Cache, the Oracle HTTP Server `Port` directive and the OracleAS Web Cache `Listen` element will be different. `Port` is the port that OracleAS Web Cache listens on, and `Listen` is the port Oracle HTTP Server listens on. By default, the values differ by 1.
- If you have port-based virtual hosts, you might want to put `Listen` near the virtual host section in the `httpd.conf` file. The port value for the virtual host should differ from the Listen port value by 1 if OracleAS Web Cache is installed.

2.3.3 Upgrading Module-Specific Directives

In 10*g* (9.0.4), a new convention for module-specific directives is observed in the `httpd.conf` file: they are placed inside of an `<IfModule>` container. Directives related to `mod_dms`, `mod_ossl`, and `mod_oc4j` now observe this convention.

For example, in Release 1 (1.0.2.2.x), module-specific directives may appear as name-value pairs, such as:

```
AA aa
BB bb
```

In 10*g* (9.0.4), you should configure these directives in the following manner:

```
<IfModule mod_mymod.c>
   AA aa
   BB bb
</IfModule>
```

In addition, extract all configuration related to DMS from the `httpd.conf` file in Release 1 (1.0.2.2.x) into the `dms.conf` file in the 10*g* (9.0.4) installation. Do the same for all SSL-related configuration, placing it into `ssl.conf`. Remove all `mod_oprocmgr` configuration and the `mod_jserv.conf` file, since these are deprecated.

2.3.4 Upgrading User-defined Configuration Files

If user-defined configuration files are part of your system, follow these steps to upgrade them.

1. Find these files by searching <*source_MT_OH*>`/Apache/Apache/conf/httpd.conf` recursively for `Include` directives.
2. Copy the files from <*source_MT_OH*> to <*destination_MT_OH*>. If the files are not located in <*source_MT_OH*>, save a backup copy of the files and make the necessary changes (Steps 3 and 4) in-place, in the original files.
3. Edit the files to reflect the new Oracle home path.
4. Restructure the files, if necessary.

2.3.5 Upgrading User-Defined Dynamic Module Libraries

If you have added dynamic module libraries to your system (on UNIX, these files have an `.so` extension; on Windows, a `.dll` extension), follow these steps to upgrade them.

1. Find these files by searching <*source_MT_OH*>`/Apache/Apache/conf/httpd.conf` recursively for `LoadModule` directives.
2. Copy the files from <*source_MT_OH*> to <*destination_MT_OH*>. (If the files are not located in <*source_MT_OH*>, and <*destination_MT_OH*> is on the same computer, no action is necessary.)

2.3.6 Upgrading CGI and FastCGI Programs

CGI scripts can be enabled in two ways: `ScriptAlias` or `ScriptAliasMatch` directives enable a URL inside the directory specified by the `DocumentRoot` directive to be mapped onto a valid directory name, identifying that directory as one containing executable scripts. The other way is to use the `ExecCGI` option, specified in an `Options` directive.

To upgrade the CGI scripts, identify them using the directory or file references in the directives in <*source_MT_OH*>`/Apache/Apache/conf/httpd.conf` (and other configuration files, if applicable) and copy them to <*destination_MT_OH*>. (If the files are not located in <*source_MT_OH*>, and <*destination_MT_OH*> is on the same computer, no action is necessary.)

2.3.7 Upgrading Static Documents

The `DocumentRoot` directive specifies the location of static documents and their related directories. The base server and each virtual host can define a document root; you must copy all static documents in each location in <*source_MT_OH*> to the corresponding location in <*destination_MT_OH*>.

2.3.8 Upgrading SSL Certificates

Upgrading from Release 1 (1.0.2.2.x) to 10*g* (9.0.4) introduces a significant change in the security configuration of the Oracle HTTP Server: `mod_ossl` replaces `mod_ssl`, which requires that SSL certificates be replaced with Oracle wallets.

If you have not replaced the SSL certificate since installation, and did not add new SSL certificates, then you do not need to perform these steps. However, if you replaced the installer-created SSL certificate with a certificate of your own, use the `osslconvert` utility to generate the Oracle wallet.

Issue these commands:

1. Issue these commands:

   export ORACLE_HOME=<*destination_MT_OH*>

   export LD_LIBRARY_PATH=$ORACLE_HOME/lib:$LD_LIBRARY_PATH

2. Execute the `osslconvert` utility with this command:

   run <*destination_MT_OH*>/Apache/Apache/bin/osslconvert -cert <*source_MT_OH certificate file*> -key <*source_MT_OH private key file*>

   where:

   - <*source_MT_OH certificate file*> is the path and filename of the certificate file in the Release 1 (1.0.2.2.x) Oracle home.
   - <*source_MT_OH private key file*> is the path and filename of the private key file in the Release 1 (1.0.2.2.x) Oracle home.

   The utility takes other parameters, described below:

   - `-chain` <*source_MT_OH certificate chain file*>
   - `-capath` <*source_MT_OH certificate authority path*>
   - `-cafile` <*source_MT_OH certificate authority file*>
   - `-wallet` <*destination_MT_OH wallet full path*>
   - `-certpass` <*password for source_MT_OH certificate*>
   - `-wltpass` <*password for destination_MT_OH wallet*>
   - `-ssowallet <yes | no>`
   - `-validate <yes | no>`

2.3.8.1 Modifying SSL-Related Directives

Upgrading from Release 1 (1.0.2.2.x) to 10*g* (9.0.4) introduces a significant change in the security configuration of the Oracle HTTP Server: `mod_ossl` replaces `mod_ssl`, so SSL-related directives must be changed.

The mod_ossl module does not use the directives listed below. The Oracle HTTP Server will not start if they are used. The following directives are replaced by SSLWallet in 10*g* (9.0.4):

- SSLCertificateFile
- SSLCertificateKeyFile
- SSLCertificateChainFile
- SSLCACertificatePath
- SSLCACertificateFile
- SSLRandomSeed
- SSLVerifyDepth

Example 2–1 shows the SSL directives used in Release 1 (1.0.2.2.x), and Example 2–2 shows the directives used in 10*g* (9.0.4).

Example 2–1 SSL Directives in Release 1 (1.0.2.2.x)

```
<IfDefine SSL>
   <VirtualHost _default_:443>
     SSLCertficateFile certificate location
     SSLCertificateKeyFile key location
     SSLCertificateChainFile chain location
     SSLVerifyClient optional_no_ca
     SSLProtocol TLSv1
   </VirtualHost>
</IfDefine>
```

Example 2–2 SSL Directives in 10g (9.0.4)

```
<IfModule mod_ossl.c>
   <VirtualHost _default_:4443>
     SSLWallet <wallet location>
     SSLVerifyClient optional
     SSLProtocol all
   </VirtualHost>
</IfModule>
```

Two directive settings may require change:

- SSLVerifyClient no longer takes optional_no_ca as a valid value. If it is set, change the setting to optional.
- If Set SSLProtocol was set to TLSv1, change it to all.

> **Tip:** Refer to the *Oracle9i Application Server Security Guide* in the Release 2 (9.0.2) documentation library, Chapter 4, "Configuring HTTP Server Security", particularly the section titled "Using Secure Sockets Layer (SSL) to Authenticate Users". All SSL configuration directives are described in detail there. Chapter 5, "using Oracle Wallet Manager", explains how to create and manage Oracle wallets.

2.4 Validating the Oracle HTTP Server Upgrade

After you complete the upgrade tasks, ensure that the upgraded version of the Oracle HTTP Server is working as expected. Steps 1 through 5 below validate that the server will start in SSL mode, and is correctly configured to be managed by Oracle Process

Management and Notification (OPMN). You must devise and perform specific tests for applications and configuration elements that are unique to your site.

Example 2–3 shows a portion of the *<destination_MT_OH>*/opmn/conf/opmn.xml file, the configuration file for OPMN.

Example 2–3 Oracle HTTP Server Entries in OPMN Configuration File

```
<ias-component id="HTTP_Server" status="enabled" id-matching="false">
  <process-type id="HTTP_Server" module-id="OHS">
   <process-set id="HTTP_Server" restart-on-death="true" numprocs="1">
       <module-data>
     <category id="start-parameters">
          <data id="config-file" value="/myconfs/httpd.conf"/>
      <data id="start-mode" value="ssl-enabled"/>
 </category>
...
```

1. If the Oracle HTTP Server configuration file is not in the default location *<destination_MT_OH>*/Apache/Apache/conf/httpd.conf after upgrade, you must specify the location to OPMN. Edit the line as shown below to include the location and filename (replacing httpd.conf with the filename, if it is different):

   ```
   <data id="config-file" value="/<path to file>/httpd.conf"/>
   ```

2. If an SSL connection is used, set the start mode to SSL, as shown:

   ```
   <data id="start-mode" value="ssl-enabled"/>
   ```

3. If OPMN is not running, start it with this command:

   ```
   opmnctl start
   ```

4. Use OPMN to start the Oracle HTTP Server by issuing this command:

   ```
   opmnctl startproc ias-component=HTTP_Server
   ```

5. Test the SSL connection by accessing `http://` and `https://` URLs.
6. Invoke CGI and FastCGI programs and verify that they function as expected.
7. Perform a test for each application configured in your site.

3

Upgrading to OracleAS Containers for J2EE (OC4J)

This chapter contains step-by-step instructions for upgrading OC4J. It contains the following major sections:

3.1 Upgrade Considerations for OC4J Applications

When you upgrade applications from Oracle9*i*AS Release 1 (1.0.2.2.x) to Oracle Application Server, certain components of the applications may require manual adjustments, or may have characteristics of which you should be aware.

The key consideration in upgrading from Oracle9*i*AS Release 1 (1.0.2.2.x) to Oracle Application Server 10*g* (9.0.4) is the adaptation from an Apache JServ servlet 2.0 environment to the OC4J servlet 2.3 environment.

This section discusses relevant upgrade considerations, covering the following topics:

Note: Whenever XML configuration files are mentioned, be aware that in an Oracle Application Server 10*g* (9.0.4) environment, as opposed to an Oracle9*i*AS Release 1 (1.0.2.2.x) or OC4J standalone environment, you should generally not edit configuration files directly. Use Oracle Enterprise Manager for configuration. If you must edit configuration files, then use the `dcmctl` utility to notify Enterprise Manager that there have been changes, as follows:

```
dcmctl updateconfig -ct oc4j
```

See the *Oracle Application Server Containers for J2EE User's Guide* for general information about using Enterprise Manager and `dcmctl` with OC4J.

3.1.1 Upgrading to the OC4J 10*g* (9.0.4) JSP Environment

This section covers the key considerations for Oracle Application Server customers in upgrading JSP applications from Oracle9*i*AS Release 1 (1.0.2.2.x) to Oracle Application Server 10*g* (9.0.4). In addition to differences between the servlet 2.0 and servlet 2.3 environments, there are differences in the JSP container.

The following topics are covered:

Note: See Section 3.1.2, "Upgrading from JServ to the OC4J 10g (9.0.4) Servlet Environment" on page 3-16" for relevant information about application environment, servlet context and servlet path mapping, application root configuration, OC4J configuration, and deployment.

3.1.1.1 OracleJSP Container Versus Orion JSP Container in Oracle9*i*AS Release 1 (1.0.2.2.x)

In Oracle9*i*AS Release 1 (1.0.2.2.x), the first release to include OC4J, there were two JSP containers:

- The container developed by Oracle and formerly known as "OracleJSP"
- The container licensed from Ironflare AB and formerly known as the "Orion JSP container"

The OracleJSP container offered a number of advantages, including useful value-added features and enhancements such as for globalization and SQLJ support. The Orion container also offered advantages, including superior speed, but had disadvantages as well. It did not always exhibit standard behavior when compared to the JSP 1.1 reference implementation (Tomcat), and its support for internationalization and globalization was not as complete.

In Oracle9*i*AS Release 2 (9.0.2) and higher, these two containers are integrated into a single JSP container, referred to as the "OC4J JSP container". This container offers the best features of both original versions and runs efficiently as a servlet in the OC4J servlet container. The integrated container primarily consists of the OracleJSP translator and the Orion JSP runtime, running with a simplified dispatcher and the OC4J core runtime classes.

3.1.1.2 Key Issues for JSP Pages: The Servlet 2.3 Versus Servlet 2.0 Environment

In Oracle9*i*AS Release 1 (1.0.2.2.x), JServ was the primary servlet environment. There are significant differences between the OC4J 9.0.4 servlet environment, which is a servlet 2.3 implementation, and the JServ servlet environment, which is a servlet 2.0 implementation. The following is a summary of highlights, particularly with respect to JSP implementations:

- Standard application environment versus `globals.jsa`: A well-defined concept of a Web application exists in the servlet 2.3 definition, but did not exist in the servlet 2.0 definition. The servlet standard and OC4J implementation now define the concept of the document root of a Web application, and how to package an application. See Section 3.1.1.4, "The Application Environment: Upgrading from OracleJSP globals.jsa in Release 1 (1.0.2.2.x)" on page 3-5 for more information. For JServ, the OracleJSP implementation (but not the Orion JSP implementation) in Oracle9*i*AS Release 1 (1.0.2.2.x) emulated the application framework through the `globals.jsa` mechanism. Use of a `globals.jsa` file is not necessary in Oracle Application Server 10*g* (9.0.4) and is not supported in OC4J.
- Request dispatcher: The concept of the request dispatcher was introduced in the servlet 2.1 specification. This mechanism allows a JSP page or servlet to include content from another page or servlet or to forward execution to another page or servlet. For JServ, the OracleJSP implementation (but not the Orion JSP implementation) in Oracle9*i*AS Release 1 (1.0.2.2.x) emulated request dispatcher functionality. For OC4J in Oracle Application Server 10*g* (9.0.4), this emulation is no longer necessary.
- Attribute storage: Beginning with the servlet 2.1 specification, request-level and application-level attribute storage is possible. Developers can use HTTP request objects and servlet context (application-level) objects to store and retrieve state information. For JServ, the OracleJSP implementation (but not the Orion JSP implementation) in Oracle9*i*AS Release 1 (1.0.2.2.x) emulated this functionality. For OC4J in Oracle Application Server 10*g* (9.0.4), this emulation is not necessary.

- Servlet filtering: The concept of servlet filtering was introduced in the servlet 2.3 specification. This mechanism allows verification and modification of HTTP request and response objects by developers. This might be used, for example, for common headers and footers or customized authentication or authorization. This functionality was not available in Oracle9*i*AS Release 1 (1.0.2.2.x) but is available in Oracle Application Server 10*g* (9.0.4).
- Globalization: The servlet 2.3 specification provides globalization support for HTTP parameters through the standard `setCharacterEncoding()` method of the HTTP request object. The OracleJSP implementation (but not the Orion JSP implementation) in Oracle9*i*AS Release 1 (1.0.2.2.x) supported globalization through the `translate_params` configuration parameter. Later OC4J JSP implementations also supported globalization through the `setReqCharacterEncoding()` method of a public utility class. You should now migrate your applications to `setCharacterEncoding()`. See Section 3.1.1.12.1, "The setCharacterEncoding() Method" on page 3-14.

3.1.1.3 Issues for Upgrading from the Release 1 (1.0.2.2.x) Orion JSP Container

The Orion container was the default JSP container in Oracle9*i*AS Release 1 (1.0.2.2.x). If that is the container you used, there are a number of considerations when upgrading to the OC4J JSP container in Oracle Application Server 10*g* (9.0.4). Following is a summary.

3.1.1.3.1 tatglib-location Setting Consider the following tag library definition in the `web.xml` file:

```
<taglib>
   <taglib-uri>/hello</taglib-uri>
   <taglib-location>WEB-INF/lib/taglib.tld</taglib-location>
</taglib>
```

Note there is no opening "/" in the `taglib-location` setting. The OC4J 9.0.4 implementation of the JSP container resolves this to the following:

```
/WEB-INF/WEB-INF/lib/taglib.tld
```

This is compliant with the JSP specification. The Orion JSP container resolves it to the following (which was presumably the intent of the developer):

```
/WEB-INF/lib/taglib.tld
```

It is advisable to change the `taglib-location` setting to the following, adding the opening "/":

```
<taglib-location>/WEB-INF/lib/taglib.tld</taglib-location>
```

3.1.1.3.2 HTML Comments The Orion JSP container sometimes ignored the content of HTML comments, denoted by `<!-- ... -->` (as opposed to `<%-- ... --%>` for JSP comments). As a result, the content within the HTML comment was not rendered in the browser.

The OC4J JSP container does not interpret HTML comments, but also does not ignore them. The container passes them through to the browser, which is behavior that complies with the JSP specification. This makes it feasible for developers to add JavaScript to an HTML comment.

3.1.1.3.3 Use of an Include Directive to Include a Page with an Unclosed Tag The JSP specification does not specify whether an `include` directive should accept a page header without a proper closing tag. The Orion JSP container would accept such a

header, but the OC4J JSP engine and the Tomcat reference implementation do not. Consider the following example:

```
-------------------------
a.jsp
<jsp:useBean id="b" class="pkgA.BeanB" >
<% // init the bean %>
-------------------------
-------------------------
b.jsp
<%@ include file="a.jsp" %>
</jsp:useBean>
<%= new java.util.Date() %>
-------------------------
```

This would be accepted by the Orion JSP container. To upgrade this to the OC4J JSP container, modify the syntax as follows:

```
-------------------------
a.jsp
<jsp:useBean id="b" class="pkgA.BeanB" >
<% // init the bean %>
</jsp:useBean>
-------------------------

-------------------------
b.jsp
<%@ include file="a.jsp" %>
<%= new java.util.Date() %>
-------------------------
```

3.1.1.3.4 Include Directive Syntax The Orion JSP container accepted the following incorrect syntax:

```
<%@ include file="value" />
```

This does not follow the specification and is not accepted by the OC4J JSP container. The correct `include` directive syntax is:

```
<%@ include file="value" %>
```

3.1.1.3.5 Quotes in Tag Attribute Settings According to the JSP 1.2 specification, tag attribute settings must always be quoted. Quotes within a setting must use an escape character. This was not clarified in previous JSP specifications, and the Orion JSP container accepted settings that were not properly quoted.

The following is incorrect, but was accepted by the Orion container:

```
<jsp:tag prop=<%=bean.getProperty("name")%> />
```

The following is correct and is now required by the OC4J JSP container (note the additional quotes and escapes):

```
<jsp:tag prop="<%=bean.getProperty(\"name\")%>"/>
```

3.1.1.4 The Application Environment: Upgrading from OracleJSP globals.jsa in Release 1 (1.0.2.2.x)

The servlet 2.0 specification did not have a clearly defined concept of a Web application and there was no defined relationship between servlet contexts and applications, as there is in later servlet specifications. In a servlet 2.0 environment such

as JServ, there is only one servlet context object per JVM. A servlet 2.0 environment also has only one session object.

The OracleJSP 1.0.2.2 implementation, however, offered the use of `globals.jsa` files (a non-standard Oracle extension) to provide support for multiple applications and multiple sessions in a Web server, particularly for use in the servlet 2.0 environment. Where a distinct servlet context object would not otherwise be available for each application, the presence of a `globals.jsa` file for an application allowed the OracleJSP container to provide the application with a distinct ServletContext object.

Because OC4J in Oracle Application Server 10*g* (9.0.4) offers a servlet 2.3 environment with standard application support, use of globals.jsa is no longer supported. If an existing application uses globals.jsa, you should migrate away from this usage. The following substitutions for globals.jsa functionality are recommended:

- Instead of using globals.jsa as an application marker, use standard WAR packaging to denote the application structure.
- Instead of using globals.jsa start-session, end-session, start-application, and end-application events, use standard servlet 2.3 listener functionality. For example, equivalent capabilities are offered through the standard javax.servlet.ServletContextListener and javax.servlet.http.HttpSessionListener interfaces.
- Instead of using globals.jsa for global variable declarations, put the declarations in a single source file and use "global include" functionality of the OC4J JSP engine, introduced in release 9.0.2. See Section 3.1.1.7, "JSP Global Includes" on page 3-8

3.1.1.5 JspScopeListener Issues for Upgrading from OracleJSP in Release 1 (1.0.2.2.x)

The OC4J JspScopeListener interface tracks page-scope, request-scope, session-scope, and application-scope events. To conform with servlet 2.3 standards, however, there are changes from how this mechanism was used in Oracle9*i*AS Release 1 (1.0.2.2.x).

For page-scope objects, no special steps or configuration are necessary if you use the OC4J JSP container. There is an Oracle-specific runtime implementation to support page scope.

If you use JspScopeListener for session-scope events, you should now implement the standard `HttpSessionBindingListener` interface as well as the JspScopeListener interface. This is necessary because the servlet 2.3 standard uses the servlet container instead of the JSP container to provide notification for session-based events. Delegate the `valueUnbound()` method of `HttpSessionBindingListener` to a common method shared by the `outOfScope()` method of the JspScopeListener interface.

`JspScopeListener` now supports request-scope objects through a servlet filter. The filtering applies to any servlets matching a specified URL pattern. Event-handling for request-scope objects requires an entry such as the following in the web.xml file for your application. To ensure proper operation of the `JspScopeListener` functionality, this setting must be after any other filter settings.

```
<filter>
   <filter-name>Request Filter</filter-name>
   <filter-class>oracle.jsp.event.impl.RequestScopeFilter</filter-class>
</filter>
<!-- Define filter mappings for the defined filters -->
<filter-mapping>
   <filter-name>Request Filter</filter-name>
   <url-pattern>/jsp/*</url-pattern>
</filter-mapping>
```

JspScopeListener now supports application-scope objects through a servlet context listener implementation class, in compliance with the servlet 2.3 specification. Event-handling for application-scope objects requires an entry such as the following in the web.xml file for your application. To ensure proper operation of the JspScopeListener functionality, this setting must be after any other listener settings.

```
<listener>
   <listener-class>oracle.jsp.event.impl.AppScopeListener</listener-class>
</listener>
```

Be aware that you should directly manipulate configuration files only in an OC4J standalone environment. In Oracle Application Server, use Oracle Enterprise Manager for configuration.

For additional information and examples, see the *Oracle Application Server Containers for J2EE JSP Tag Libraries and Utilities Reference*.

3.1.1.6 JSP Classpath Functionality

The OC4J JSP container in Oracle Application Server 10*g* (9.0.4) uses standard locations on the Web server in searching for translated JSP pages and any .class files and .jar files that they require. The container will find files in these locations without any Web server classpath configuration, and has the ability to automatically reload classes in these locations (depending on configuration settings).

The locations for dependency classes are as follows and are relative to the application root:

```
/WEB-INF/classes/...
```

```
/WEB-INF/lib
```

The location for JSP page implementation classes (translated pages) is as follows:

```
.../_pages/...
```

The /WEB-INF/classes directory is for individual Java .class files. These classes should be stored in subdirectories under the classes directory according to Java package naming conventions. For example, consider a JavaBean called LottoBean whose code defines it to be in the oracle.jsp.sample.lottery package. The JSP container will look for LottoBean.class in the following location relative to the application root:

```
/WEB-INF/classes/oracle/jsp/sample/lottery/LottoBean.class
```

The lib directory is for .jar files. Because the Java package structure is specified in the .jar file structure, the .jar files are all placed directly in the /WEB-INF/lib directory, not in subdirectories. As an example, LottoBean.class might be stored in lottery.jar, located as follows relative to the application root:

```
/WEB-INF/lib/lottery.jar
```

By default, the _pages directory is under the following directory in OC4J:

/j2ee/home/application-deployments/<*app name*>/<*web app name*>/persistence

The *app name* is determined through the <application> element in the OC4J server.xml file; the *web app name*, which corresponds to the WAR file name, is mapped to the *app name* through the <web-app> element in the OC4J

default-web-site.xml file (or http-web-site.xml in OC4J standalone). See the *Oracle Application Server Containers for J2EE Servlet Developer's Guide* for information.

Generated page implementation classes for translated JSP pages are placed in subdirectories under the _pages directory according to the locations of the original .jsp files.

> **Note:** Implementation details, such as the location of the _pages directory, are subject to change in future releases.

3.1.1.7 JSP Global Includes

In Oracle9*i*AS Release 2 (9.0.2) and later, the OC4J JSP container provides a feature called global includes. You can use this feature to specify one or more files to statically include into JSP pages in (or under) a specified directory, through virtual JSP include directives. During translation, the JSP container looks for a configuration file, /WEB-INF/ojsp-global-include.xml, that specifies the included files and the directories for the pages.

This enhancement is particularly useful in upgrading applications that had used globals.jsa or translate_params functionality in Oracle9*i*AS Release 1 (1.0.2.2.x).

Globally included files can be used for the following, for example:

- global bean declarations (formerly supported through globals.jsa)
- common page headers or footers
- translate_params equivalent code (for globalization)

The ojsp-global-include.xml File

The ojsp-global-include.xml file specifies the names of files to include, whether they should be included at the tops or bottoms of JSP pages, and the locations of JSP pages to which the global includes should apply. This section describes the elements of ojsp-global-include.xml.

3.1.1.7.1 <ojsp-global-include> This is the root element of the ojsp-global-include.xml file. It has no attributes.

Subelements:

<include>

3.1.1.7.2 <include...> Use this subelement of <ojsp-global-include> to specify a file to be included, and whether it should be included at the top or bottom of JSP pages.

Subelements:

<into>

Attributes:

- file: Specify the file to be included, such as /header.html or /WEB-INF/globalbeandeclarations.jsph. The file name must start with a slash ("/"). In other words, it must be context-relative, not page-relative.
- position: Specify whether the file is to be included at the top or bottom of JSP pages. Supported values are "top" (default) and "bottom".

3.1.1.7.3 <into...> Use this subelement of `<include>` to specify a location (a directory, and possibly subdirectories) of JSP pages into which the specified file is to be included. This element has no subelements.

Attributes:

- `directory`: Specify a directory. Any JSP pages in this directory, and optionally its subdirectories, will statically include the file specified in the `file` attribute of the `<include>` element. The directory setting must start with a slash ("/"), such as `/dir1`. The setting can also include a slash after the directory name, such as `/dir1/`, or a slash will be appended internally during translation.
- `subdir`: Use this to specify whether JSP pages in all subdirectories of the directory should also have the file statically include. Supported values are "true" (default) and "false".

3.1.1.7.4 Global Include Examples This section provides examples of global includes.

Example 3–1 Header/Footer: ojsp-global-include.xml File

```
<?xml version="1.0" standalone='yes'?>
<!DOCTYPE ojsp-global-include SYSTEM 'ojsp-global-include.dtd'>

<ojsp-global-include>
  <include file="/header.html">
     <into directory="/dir1" />
  </include>
  <include file="/footer1.html" position="bottom">
     <into directory="/dir1" subdir="false" />
     <into directory="/dir1/part1/" subdir="false" />
  <include file="/footer2.html" position="bottom">
  </include>
     <into directory="/dir1/part2/" subdir="false" />
  </include>
</ojsp-global-include>
```

This example accomplishes three objectives:

- The `header.html` file is included at the top of any JSP page in or under the `dir1` directory. The result would be the same as if each `.jsp` file in or under this directory had the following `include` directive at the top of the page:

  ```
  <%@ include file="/header.html" %>
  ```

- The `footer1.html` file is included at the bottom of any JSP page in the `dir1` directory or its `part1` subdirectory. The result would be the same as if each `.jsp` file in those directories had the following `include` directive at the bottom of the page:

  ```
  <%@ include file="/footer1.html" %>
  ```

- The `footer2.html` file is included at the bottom of any JSP page in the `part2` subdirectory of `dir1`. The result would be the same as if each `.jsp` file in that directory had the following `include` directive at the bottom of the page:

  ```
  <%@ include file="/footer2.html" %>
  ```

> **Note:** If multiple header or multiple footer files are included into a single JSP page, the order of inclusion is according to the order of `<include>` elements in the `ojsp-global-include.xml` file.

Example 3–2 translate_params Equivalent Code: ojsp-global-include.xml File

```
<?xml version="1.0" standalone='yes'?>
<!DOCTYPE ojsp-global-include SYSTEM 'ojsp-global-include.dtd'>

<ojsp-global-include>
  <include file="/WEB-INF/nls/params.jspf">
     <into directory="/" />
  </include>
</ojsp-global-include>
```

Further assume that the `params.jspf` file contains the following:

```
<% request.setCharacterEncoding(response.getCharacterEncoding();
%>
```

The `params.jspf` file is included at the top of any JSP page in or under the application root directory. In other words, it is included in any JSP page in the application. The result would be the same as if each `.jsp` file in or under this directory had the following `include` directive at the top of the page:

```
<%@ include file="/WEB-INF/nls/parms.jspf" %>
```

Alternatively, the globally included file could consist of scriptlet code performing one of the following actions:

- Hardcode the request character set:

  ```
  <% request.setCharacterEncoding("desired_charset"); %>
  ```

- Use the character set of the response as the character set of the request, where the character set of the response is determined dynamically by Java logic:

  ```
  <% String yourCharSet = yourLogicToDetermineCharset();
     response.setContentType("text/html; charset="+yourCharSet);
     request.setCharacterEncoding(response.getCharacterEncoding());
     // NOTE: The relative ordering of response.setContentType()
     // and request.setCharacterEncoding() is important.
  %>
  ```

  Also see Section 3.1.1.12.1, "The setCharacterEncoding() Method" on page 3-14.

3.1.1.8 JSP Configuration: Upgrading from the OracleJSP Implementation in Release 1 (1.0.2.2.x)

In Oracle9*i*AS Release 2 (9.0.2) and later, the `oracle.jsp.runtimev2.JspServlet` front-end servlet is used instead of the `oracle.jsp.JspServlet` front-end servlet that was used in the 1.0.2.2 OracleJSP implementation.

Mapping of this class as the JSP servlet is handled automatically in the OC4J `global-web-application.xml` file, as in the following entry:

```
<servlet>
   <servlet-name>jsp</servlet-name>
   <servlet-class>oracle.jsp.runtimev2.JspServlet</servlet-class>
   ...
      init params
   ...
</servlet>
```

This file also includes `<servlet-mapping>` elements where file name extensions (`.jsp`, `.JSP`, `.sqljsp`, `.SQLJSP`, and `.jspx`) are mapped to this front-end servlet.

Configuration parameters ("init params", see above) can also be specified in `global-web-application.xml`, as in the following example:

```
<init-param>
   <param-name>precompile_check</param-name>
   <param-value>true</param-value>
</init-param>
```

3.1.1.8.1 Support for Previous OracleJSP Configuration Parameters Table 3–1 summarizes JSP configuration parameters supported in the front-end servlet for the JServ environment, and notes which are still relevant and supported.

Table 3–1 Configuration Parameter Support: JServ to OC4J

Supported Config Params in JServ	Supported Config Params in OC4J	Comments
`alias_translation`		not necessary in OC4J
`bypass_source`		migrated to main_mode parameter
`classpath`		not necessary in OC4J
`debug_mode`	`debug_mode`	no change
`developer_mode`		migrated to main_mode parameter
`emit_debuginfo`	`emit_debuginfo`	no change
`external_resource`	`external_resource`	no change
`javaccmd`	`javaccmd`	no change
`send_error`		unnecessary in OC4J
`session_sharing`		unnecessary in OC4J
`sqljcmd`	`sqljcmd`	no change
`translate_params`		Unnecessary in OC4J; use standard servlet request `setCharacterEncoding()` method. Note: See "translate_params Equivalent Code: ojsp-global-include.xml File" on page 3-10.
`unsafe_reload`		unnecessary in OC4J

3.1.1.8.2 JSP Configuration Parameters in Release 2 (9.0.2) and Higher The following JSP configuration parameters were added in Oracle9*i*AS Release 2 (9.0.2):

- `main_mode`: This determines whether classes are automatically reloaded or JSP pages are automatically recompiled, in case of changes. Possible settings are `justrun`, `reload`, and `recompile`.
- `old_include_from_top`: Set this boolean to `true` for page locations in nested `include` directives to be relative to the top-level page, for backward compatibility with OracleJSP behavior in Oracle9*i*AS Release 1 (1.0.2.2.x).
- `precompile_check`: Set this boolean to `true` to check the HTTP request for a standard `jsp_precompile` setting.
- `reduce_tag_code`: Set this boolean to `true` for further reduction in the size of generated code for custom tag usage.

- `req_time_introspection`: Set this boolean to `true` to enable request-time JavaBean introspection if compile-time introspection is not possible. If compile-time introspection is possible and succeeds, this parameter is ignored and there is no request-time introspection.
- `static_text_in_chars`: Set this boolean to `true` to instruct the JSP translator to generate static text in JSP pages as characters instead of bytes. Also see Section 3.1.1.12.2, "Static Text as Characters" on page 3-15.
- `tags_reuse_default`: This specifies a default setting for JSP tag handler pooling (`true` to enable by default; `false` to disable by default). This default setting can be overridden for any particular JSP page. Also see Section 3.1.1.12.4, "Tag Handler Reuse" on page 3-15.
- `xml_validate`: Set this boolean to specify whether XML validation is to be performed on the `web.xml` file and TLD files.

See *Oracle Application Server Containers for J2EE Support for JavaServer Pages Developer's Guide* for additional information.

3.1.1.8.3 JSP Configuration Parameters in Release 2 (9.0.3) and Higher for Backward Compatibility

When upgrading to Oracle Application Server 10*g* (9.0.4) and using JSP pages, use appropriate settings for the following important JSP configuration parameters:

- `check_page_scope`
- `forgive_dup_dir_attr`

These are set as initialization parameters for the JSP front-end servlet, either in the `global-web-application.xml` file or in the application `web.xml` file. Here is an example:

```
<servlet>
   <servlet-name>jsp</servlet-name>
   <servlet-class>oracle.jsp.runtimev2.JspServlet</servlet-class>
   <init-param>
      <param-name>check_page_scope</param-name>
      <param-value>true</param-value>
   </init-param>
   ...
</servlet>
```

Manipulate configuration files directly only if you are in an OC4J standalone environment. In Oracle Application Server use Oracle Enterprise Manager for configuration.

See the *Oracle Application Server Containers for J2EE Support for JavaServer Pages Developer's Guide* for more information about JSP configuration parameters.

- `check_page_scope` (boolean; default: `false`): This parameter is supported in Oracle9*i*AS Release 2 (9.0.3) and higher. For OC4J environments, set it to `true` to enable Oracle-specific page-scope checking by the `JspScopeListener` utility.

  This parameter is not relevant for non-OC4J environments. For JServ, Oracle-specific page-scope checking is always enabled. See the *Oracle Application Server Containers for J2EE JSP Tag Libraries and Utilities Reference* for information about `JspScopeListener`.
- `forgive_dup_dir_attr` (boolean; default: `false`): This parameter is supported in Oracle9*i*AS Release 2 (9.0.3) and higher. Set it to `true` to avoid translation errors in a JSP 1.2 environment such as OC4J if you have duplicate settings for the same

directive attribute within a single JSP translation unit (a JSP page plus anything it includes through `include` directives).

The JSP 1.2 specification directs that a JSP container must verify that directive attributes, with the exception of the page directive import attribute, are not set more than once each within a single JSP translation unit.

The JSP 1.1 specification did not specify such a limitation. OC4J offers the `forgive_dup_dir_attr` parameter for backward compatibility.

3.1.1.9 Potential Issues with the ojspc Utility

There are a few relatively minor upgrade considerations regarding the `ojspc` pre-translation utility in Oracle Application Server 10*g* (9.0.4).

3.1.1.9.1 Running ojspc for the OC4J Environment The `ojspc` front-end script that sets up the classpath for pre-translation has been modified since Oracle9*i*AS Release 1 (1.0.2.2.x). Most users running `ojspc` for OC4J should not encounter problems using the different `ojspc` defaults; however, there are two potential issues to consider:

- If your application relies on the pre-JSP 1.2 behavior of the `include` directive, you can set `-oldIncludeFromTop=true` for compatibility with this behavior. This `ojspc` option has the same functionality as the `old_include_from_top` JSP configuration parameter.
- In OC4J, static text is now generated in bytes by default, whereas it was generated in characters in Oracle9*i*AS Release 1 (1.0.2.2.x), in the JServ environment. You can set `-staticTextInChars=true` if you want the old behavior. This ojspc option has the same functionality as the `static_text_in_chars` JSP configuration parameter.

See Section 3.1.1.8.2, "JSP Configuration Parameters in Release 2 (9.0.2) and Higher" on page 3-11 for information about the `old_include_from_top` and `static_text_in_chars` parameters.

3.1.1.9.2 Running ojspc_jserv for the JServ Environment If you want to use `ojspc` for pre-translation for the JServ environment in Oracle Application Server 10*g* (9.0.4), use the `ojspc_jserv` script (`ojspc_jserv.bat` on Windows) instead of the `ojspc` (or `ojspc.bat`) script.

3.1.1.10 Packaging and Deployment

In Oracle Application Server 10*g* (9.0.4), deployment is through the standard J2EE EAR (Enterprise archive) file, with the WAR (Web archive) file included inside the EAR file.

For OC4J, deploy each application through a standard EAR file. The name of the application and the name and location of the EAR file are specified through an `<application>` element in the OC4J `/j2ee/home/config/server.xml` file. In a standalone environment, you can accomplish this through the `admin.jar` utility. In an Oracle Application Server environment, use Enterprise Manager.

The EAR file includes the following:

- a standard `application.xml` configuration file, in `/META-INF`
- (optionally) an `orion-application.xml` configuration file, in `/META-INF`
- a standard WAR file

The WAR file includes the following:

- a standard `web.xml` configuration file, in `/WEB-INF`

- (optionally) an `orion-web.xml` configuration file, in `/WEB-INF`
- all JSP pages and Java classes (servlets, JavaBeans, and other classes) necessary to run the application, under `/WEB-INF/classes` and in JAR files in `/WEB-INF/lib`

For details about deployment in Oracle Application Server 10*g*, refer to the *Oracle Application Server Containers for J2EE User's Guide* or Oracle Application Server Containers for J2EE Stand Alone User's Guide, as applicable.

3.1.1.11 Enabling Extra Imports

In Oracle9*i*AS Release 2 (9.0.3) and higher, the OC4J JSP container by default imports the packages listed below into any JSP page, in compliance with the JSP specification. No page directive import settings are required.

```
javax.servlet.*

javax.servlet.http.*

javax.servlet.jsp.*
```

In earlier releases, the following packages were also imported by default:

```
java.io.*

java.util.*

java.lang.reflect.*

java.beans.*
```

For backward compatibility, you can use the JSP `extra_imports` configuration parameter as a workaround. Alternatively, you can add desired imports through page directives or global includes (see Section 3.1.1.7, "JSP Global Includes" on page 3-8). See the *Oracle Application Server Containers for J2EE Support for JavaServer Pages Developer's Guide* for information about these topics.

3.1.1.12 Other JSP Considerations

This section covers additional JSP upgrade considerations.

3.1.1.12.1 The setCharacterEncoding() Method Effective with the servlet 2.3 specification, the `setCharacterEncoding()` method is available in the `javax.servlet.ServletRequest` class as the standard mechanism for specifying a non-default character encoding for reading HTTP requests. The signature of this method is as follows:

```
void setCharacterEncoding(java.lang.String enc)
            throws java.io.UnsupportedEncodingException
```

The `enc` parameter is a string specifying the name of the desired character encoding, overriding the default character encoding. Call this method before reading request parameters or reading input through the `getReader()` method (also of the `ServletRequest` class).

There is also a corresponding getter method:

```
String getCharacterEncoding()
```

In OC4J implementations using pre-2.3 servlet environments, including the implementation in Oracle9*i*AS Release 1 (1.0.2.2.x), the `setCharacterEncoding()` method was not available. For such environments, particularly the JServ servlet 2.0 environment, the OracleJSP implementation provided alternative mechanisms as

non-standard extensions: the `oracle.jsp.util.PublicUtil.setReqCharacterEncoding()` method and the `translate_params` configuration parameter.

In using the application server 9.0.4 implementation, you should upgrade to the `setCharacterEncoding()` method wherever possible. In summary:

- Use `setCharacterEncoding()` for IANA encodings.
- Use `setReqCharacterEncoding()` for non-IANA encodings. (The `setCharacterEncoding()` method does not support such encodings.)
- Avoid using the `translate_params` configuration parameter. For information about global includes, which you can use to replace `translate_params` functionality `with setCharacterEncoding()` calls (among many other uses), see Section 3.1.1.7, "JSP Global Includes" on page 3-8.

3.1.1.12.2 Static Text as Characters In JServ, which was the primary servlet environment under Oracle9*i*AS Release 1 (1.0.2.2.x), static text is output in character format. In OC4J, static text is output as bytes by default, for faster throughput.

Some globalization functionality and flexibility is unavailable if static text is generated as bytes. For this reason, the OC4J JSP container supports the configuration parameter `static_text_in_chars` if you want to revert to character format for any reason.

Enable this flag, for example, if your application requires the ability to change the character encoding dynamically during runtime, such as in the following example:

```
<% response.setContentType("text/html; charset=UTF-8"); %>
```

3.1.1.12.3 Proper Handling of jsp:param Settings Consider the following `jsp:include` tag and nested `jsp:param` tag:

```
<jsp:include page="..." >
   <jsp:param name="..." value="..." />
</jsp:include>
```

With the OC4J JSP container in Oracle Application Server 10*g* (9.0.4), there is no need to manually encode the `name` and `value` settings for the `jsp:param` tag. Just use the appropriate Java strings; encoding is handled automatically. This was not the case with the Orion JSP container in Oracle9*i*AS Release 1 (1.0.2.2.x), where manual encoding was required.

3.1.1.12.4 Tag Handler Reuse In Oracle Application Server 10*g* (9.0.4), you can specify whether JSP tag handler instances are pooled in a particular JSP page (always in the application scope) by setting the `oracle.jsp.tags.reuse` attribute in the JSP page context. Set it to `true` to enable pooling, or to `false` to disable pooling. For example:

```
pageContext.setAttribute("oracle.jsp.tags.reuse", new Boolean(true));
```

You can use separate settings in different pages, or even in different sections of the same page.

The default is according to the setting of the `tags_reuse_default` JSP configuration parameter. This default is `true` in OC4J but `false` in JServ.

3.1.1.12.5 Session Key Seed Generation OC4J in Oracle Application Server 10*g* (9.0.4) has a more secure session key seed generation process. When the first `HttpSession` object is created in an OC4J instance, there are a number of threads created to generate the session key seed. Therefore, users experience a longer delay during the first compilation or serving of a JSP page that uses session objects, compared to when using

the OC4J in Oracle9*i*AS Release 1 (1.0.2.2.x). After the seed is generated, the process of compiling and serving JSP pages is as fast as before.

3.1.2 Upgrading from JServ to the OC4J 10*g* (9.0.4) Servlet Environment

This section covers the major considerations for upgrading servlet applications from Oracle9*i*AS Release 1 (1.0.2.2.x) to Oracle Application Server 10*g* (9.0.4). This involves adapting from a servlet 2.0 environment, such as Apache JServ in the 1.0.2.2 implementation, to a servlet 2.3 environment, such as Oracle Application Server Containers for J2EE in the 9.0.4 implementation.

Included is information about application environment and servlet zones, class and file locations, mount points, servlet aliases, initialization parameters, environment and JVM settings, prestarting, class loading, logging, servlet sessions, load balancing, and fault tolerance. The section contains the following topics:

Most sections present information for JServ as a reference, followed by the corresponding information for OC4J.

Note: The material in this section assumes that the reader has knowledge of the Sun Microsystems Java Servlet Specification, versions 2.2 and 2.3, including WAR files, EAR files, and the `web.xml` file is assumed. Prior knowledge of OC4J configuration files is also helpful.

For more information about the topics discussed here, refer to the *Oracle Application Server Containers for J2EE Servlet Developer's Guide* and the *Oracle Application Server Containers for J2EE User's Guide*.

3.1.2.1 The Web Application Environment

In Oracle9*i*AS Release 1 (1.0.2.2.x), Apache JServ, a servlet 2.0 environment, was the primary servlet environment. The servlet 2.0 specification did not have a clearly defined concept of a Web application and there was no defined relationship between servlet contexts and Web applications, as there is in later servlet specifications. In a servlet 2.0 environment such as JServ, there is only one servlet context object per JVM. A servlet 2.0 environment also has only one session object.

In Oracle Application Server 10*g* (9.0.4), OC4J is the primary application environment. OC4J includes a servlet 2.3 container with standard Web application support.

This section discusses the Web application environment, contrasting servlet 2.0 features and servlet 2.3 features.

3.1.2.1.1 Servlet Context and Servlet Path Mapping The servlet 2.2 and 2.3 specifications provide for each Web application to have its own servlet context, unlike in the servlet 2.0 JServ environment. This section reviews the servlet 2.2 and 2.3 functionality.

Each servlet context is associated with a directory path, the application root, in the server file system that is the base path for modules of the Web application. Each Web application has its own application root. For a Web application in a servlet 2.2 or 2.3 environment, servlets, JSP pages, and static files such as HTML files all share this application root. By contrast, in servlet 2.0 environments the application root for servlets and JSP pages is distinct from the doc root for static files.

Note that a URL for a servlet is typically of the form:

`http://<host>[:<port>]/<context path>/<servlet path>`

When a servlet context is created, a mapping is specified between the application root and the context path portion of a URL. The servlet path is defined in the application `web.xml` file—the `<servlet>` element within `web.xml` associates a servlet class with a servlet name, and the `<servlet-mapping>` element within `web.xml` associates a URL pattern with a servlet name. When a request reaches a Web application, the servlet container will compare the path in the request with known URL patterns defined in `web.xml`, and invoke the servlet according to a matched URL pattern. See the *Oracle Application Server Containers for J2EE Servlet Developer's Guide* for more information.

3.1.2.1.2 Servlet Zones Versus Web Applications JServ has the concept of servlet zones, somewhat comparable to the Web application concept in servlet 2.2 and higher specifications. This section compares basic zone setup in JServ to basic application setup in OC4J.

Basics of JServ Zone Specification A servlet zone is somewhat similar in concept to a Web application. The use of zones helps developers separate the overall JServ environment into separate groups of servlets, according to conditions such as work load, usage, and security privileges. In JServ, servlets are grouped and managed based on servlet zones, not based on the servlet container itself. It is mandatory to have at least one servlet zone.

Servlet zones are specified in the `jserv.properties` file. In addition, each zone has its own configuration file, known as a "zone properties file", typically with a naming convention such as `zonexxx.properties`.

Here is an example of zone settings in jserv.properties:

```
zones = zone1,zone2
```

The locations of the corresponding zone properties files are also specified in `jserv.properties`, as in the following example:

```
zone1.properties =/servlet/zone1/zone1.properties

zone2.properties =/servlet2/zone2/zone2.properties
```

Basics of OC4J Application Specification In OC4J, Web applications can be considered equivalent to zones. It is also true that in OC4J, servlets are grouped and managed on a per-application basis. Additionally, OC4J has the concept of a default

global application that is the parent of all applications, and also defines a default Web application. Following is an overview of key OC4J configuration files for Web applications.

These are global files for all OC4J applications, typically in the OC4J `/j2ee/home/config` directory:

- `default-web-site.xml` (or `http-web-site.xml` for OC4J standalone): This includes a `<web-app>` element for each Web application for the default Web site, mapping the application name to the "Web application name". The Web application name corresponds to the WAR deployment file name. Additional Web site XML files, as specified for additional Web sites in the `server.xml` file, have the same functionality.
- `global-web-application.xml`: This is a global configuration file for OC4J Web applications, establishing default configurations, and including setup and configuration of the JSP front-end servlet, `JspServlet`.
- `application.xml`: This is a configuration file for the global OC4J J2EE application.
- `data-sources.xml`: This specifies data sources for database connections.

> **Note:** n an Oracle Application Server environment, use Oracle Enterprise Manager for configuration. If you must update configuration files directly, use the `dcmctl` utility to make Enterprise Manager aware of the changes, as described at the beginning of Section 3.1, "Upgrade Considerations for OC4J Applications" on page 3-1.

In addition to the global `application.xml` file, there is a standard `application.xml` file, and optionally an `orion-application.xml` file, for each application. These files are in the application EAR file.

Also, in an application WAR file, which is inside the application EAR file, there is a standard `web.xml` file and optionally an `orion-web.xml` file. These are for application-specific and deployment-specific configuration settings, overriding `global-web-application.xml` settings or providing additional settings as appropriate. The `global-web-application.xml` and `orion-web.xml` files support the same elements, a superset of those supported by the `web.xml` file.

If the `orion-application.xml` and `orion-web.xml` files are not present in the archive files, they will be generated during initial deployment according to settings in the global `application.xml` file or the `global-web-application.xml` file, respectively.

For more information about the use of these files, see the *Oracle Application Server Containers for J2EE User's Guide* and the *Oracle Application Server Containers for J2EE Servlet Developer's Guide*.

3.1.2.2 Enterprise Applications Versus Web Applications

According to the servlet 2.0 specification, and therefore in the JServ environment of the 1.0.2.2 application server implementation, Web applications stood on their own and were deployed through standalone WAR files. According to the J2EE 1.3 and servlet 2.3 specifications, and therefore in the OC4J environment of the 9.0.4 application server implementation, a Web application is typically part of a J2EE enterprise application, which may also include other components, such as EJBs. Reflecting this structure, the

WAR file for a Web application is contained within the EAR file for the enterprise application of which the Web application is a part. The enterprise application and all its components, including Web applications, are deployed through the EAR file.

Enterprise applications and EAR files are discussed in the *Java2 Platform Enterprise Edition Specification*, version 1.3.

3.1.2.3 Application Structure and File Location

This section discusses the typical OC4J Web application structure and how to configure file locations in OC4J compared to JServ.

3.1.2.3.1 JServ File Repositories The locations, or repositories, of servlets under a servlet zone are specified through `repositories` commands in the zone properties file. (See Section 3.1.2.1.2, "Servlet Zones Versus Web Applications", sub-topic "Basics of JServ Zone Specification" on page 3-17, for information about zone properties files.) JServ loads classes from locations specified in repository entries. Here are some examples:

```
repositories=/private/mydir/net.jar

repositories=/private/mydir/tmp/net.zip

repositories=/private/mydir/applications
```

As shown, class files can be read directly from directories or from archive files (ZIP or JAR). See Section 3.1.2.7.1, "Class Loading in JServ" on page 3-26 for more information about repositories and class loading.

3.1.2.3.2 OC4J Application Structure and File Locations This section shows the typical (and recommended) Web application structure under the application root directory. Much of it applies generally to servlet 2.2 and 2.3 environments:

```
WEB-INF/
   web.xml
   orion-web.xml
   classes/
      package/
         xxx.class
   lib/
      xxx.jar
xxx.jsp
xxx.html
```

In OC4J, the root directory is *app name*`/`*web app name*. The application name is defined in the `server.xml` file and mapped to a Web application name in the `default-web-site.xml` file or other Web site XML file.

According to this structure, servlet classes go under the *app name*`/`*web app name*`/WEB-INF/classes` directory, in subdirectories according to package names as appropriate. For example, if you have `HelloWorldServlet` in the examples package, then the class file should be located as follows:

app name`/`*web app name*`/WEB-INF/classes/examples/HelloWorldServlet`

Place HTML files, JSP pages, and other resource files in the application root directory.

Place required library files, such as JAR files, in the following directory:

app name`/`*web app name*`/WEB-INF/lib`

Note that in OC4J as well as in JServ, class files can be read either directly from a directory or from an archive file (ZIP or JAR).

You can also specify a codebase, placing additional required files in a desired location, then adding that location to your classpath through the `<classpath>` subelement of the `<orion-web-app>` element in the `orion-web.xml` file, as in the following example:

```
<classpath path="/private/test/test.jar" />
```

In an Oracle Application Server environment, configure this using Oracle Enterprise Manager.

3.1.2.4 OC4J Deployment

This section covers the basics of deployment in the OC4J 9.0.4 implementation.

In an Oracle Application Server environment, you configure and deploy applications using Oracle Enterprise Manager. Also note that you will typically use an Enterprise archive (EAR) file for OC4J deployment.

Consider the following example, the procedure for creating an EAR file for a simple application with a servlet `HelloServlet` and a JSP page `Hello.jsp`:

1. Create and configure the `web.xml` file to specify application-specific settings. Also configure the `orion-web.xml` file, if desired. Use Enterprise Manager if you are in an Oracle Application Server environment.
2. Create a Web archive (WAR) file, `helloapp.war`.

   The `helloapp.war` file contains the following:

   ```
   META-INF/MANIFEST.MF
   WEB-INF/
   WEB-INF/classes/
   WEB-INF/classes/HelloServlet.class
   WEB-INF/orion-web.xml
   WEB-INF/web.xml
   Hello.jsp
   ```

(The `META-INF/MANIFST.MF` file is created by the JAR utility. There is no need to modify it.)

3. Create an EAR file, `helloapp.ear`, for deploying the application. The EAR file contains:

   ```
   META-INF/
   META-INF/MANIFEST.MF
   WEB-INF/application.xml
   WEB-INF/orion-applcation.xml
   helloapp.war
   ```

The `application.xml` file is required when deploying an EAR file, but the `orion-application.xml` file is optional.

Using this `helloapp.ear` file, you can deploy the Web application using Oracle Enterprise Manager.

3.1.2.5 Servlet and Environment Setup

This section covers a variety of servlet setup issues, reviewing configuration steps for JServ and providing the equivalent steps for OC4J.

3.1.2.5.1 JVM Parameters and Environment Variables This section describes how to set JVM and environment variables for OC4J and JServ.

Setting Environment Variables in JServ In `jserv.properties`, you can set environment variables for the JVM to use, including Oracle environment settings.

The following examples are for Oracle settings:

```
wrapper.env=LD_LIBRARY_PATH=...
```

```
wrapper.env=ORACLE_HOME=...
```

```
wrapper.env=ORACLE_SID=...
```

There are also settings for the Java and system environment:

- Use the following to set the classpath:

  ```
  wrapper.classpath=...
  ```

- Use the following to set the file path:

  ```
  wrapper.path=...
  ```

- Use the following to set the full path of the Java interpreter (if it is not visible in the path):

  ```
  wrapper.bin=...
  ```

- Use `wrapper.bin.parameters` for JVM loading parameters, such as for heap or stack size. Following is an example:

  ```
  wrapper.bin.parameters=-Xms64m
  ```

- Use `wrapper.env.copy` or `wrapper.env.copyall` to copy environment parameters from the caller to the JVM. The following examples copy all environment parameters, and a particular environment parameter (`myparam`), respectively:

  ```
  wrapper.env.copyall
  ```

  ```
  wrapper.env.copy=myparam
  ```

In addition, JServ has the following security parameters:

`security.maxConnections`, `security.allowedaddresses`, and `security.authentication`.

Setting Environment Variables in OC4J The method used to set environment variables in OC4J depends on how it is started.

In particular, if OC4J is started from Oracle Enterprise Manager (that is, via the process management module OPMN), then parameters are specified through the `<`*`ORACLE_HOME`*`>/opmn/conf/opmn.xml` configuration file. In particular, the following subelements of the `<oc4j>` element are relevant:

- The `<java-bin>` subelement specifies a path to the Java executable. If this element is not specified, `<`*`ORACLE_HOME`*`>/jdk/bin/java` is used by default.
- The `<java-option>` subelement specifies the command line parameters required by the JVM.

The `<java-bin>` subelement is equivalent to `wrapper.bin` in JServ. The `<java-option>` subelement is equivalent to `wrapper.bin.parameters` in JServ.

You should generally use Enterprise Manager to configure these parameters. If you must update the configuration file directly, use the `dcmctl` utility to make Enterprise Manager aware of the changes, as described at the beginning of Section 3.1, "Upgrade Considerations for OC4J Applications" on page 3-1.

Here is an example (showing only the relevant portions of <oc4j> element syntax):

```
<oc4j numProcs="1"  maxRetry="4" ... >
      <java-bin path="/private/my-sun/jdk/bin/java" />
     <java-option value="-Xmls32m -Xmx64m -Xss128K
                         -Doracle.ons.oraclehome=/private/oracle" />
...
</oc4j>
```

The opmn.xml file also specifies Oracle environment variables, through the <environment> element, such as in the following example:

```
<environment>
   <prop name="PATH" value="/private/home/ias/lib"/>
   <prop name="CLASSPATH" value="/private/home/ias/bin" />
   <prop name="LD_LIBRARY_PATH" value="/private/home/lib" />
</environment>
```

These settings specify the environment for the new process when it is spawned.

See the *Oracle Application Server 10g Administrator's Guide* for information about OPMN.

In an OC4J standalone environment, you can set parameters through the oc4j.properties file and use the -properties option in the Java command line to point to the location of the file.

3.1.2.5.2 Mount Settings JServ (mod_jserv) and OC4J (mod_oc4j) each have "mount" commands to establish application root locations, or "mount points".

JServ Mount Settings In JServ, mount points are used to define the root locations for different servlet zones. If the protocol, host, or port are not specified, then they are picked from default entries in the jserv.conf file, which is included into the httpd.conf file. Consider the following sample mount commands:

```
ApJServMount /servlets /root
ApJServMount /servlets/admin ajpv11://myhost:9009/admin
```

These commands have the following effect:

The URL below, as a result of the first ApJServMount command, requests the servlet HelloWorldServlet in the servlet zone root:

```
http://myhost.mycompany.com/servlets/HelloWorldServlet
```

The URL below, as a result of the second ApJServMount command, is handled through port 9009, using Apache JServ Protocol (AJP) version 1.1.

```
http://myhost.mycompany.com/servlets/admin/HelloWorldServlet
```

OC4J Mount Settings In the OC4J and Oracle HTTP Server environment, you can specify mount points through the <*ORACLE_HOME*>/Apache/modoc4j/conf/mod_oc4j.conf file. You can specify additional relevant Java-side settings, such as host and port, through default-web-site.xml settings. While the default-web-site.xml file is mentioned in this section, be aware that in an Oracle Application Server environment, you should perform the configuration using Oracle Enterprise Manager rather than by editing this file.

Refer to "JServ Mount Settings" on page 3-22 for comparisons. Consider the following JServ example again:

```
ApJServMount /servlets/admin ajpv11://myhost:9009/admin
```

You can make equivalent protocol and host settings through the <web-site> element in default-web-site.xml as follows:

```
<web-site port="9009" protocol="ajp13" ...>
...
</web-site>
```

Also note that within the <web-site> element, you can use <web-app> subelements to specify information about individual applications on the site, such as the corresponding Web application name and application root. Here is an example:

```
<web-app application="ojspdemos" name="ojspdemos-web"
           root="/ojspdemos" />
```

On the target host, myhost, OC4J would find the Web application according to application settings in default-web-site.xml.

Based on the preceding Oc4jMount command, the following request would be routed to an OC4J process that listens at myhost on port 9009 using Apache JServ Protocol (AJP) version 1.3:

```
http://myhost.mycompany.com/servlets/admin/HelloWorld
```

OC4J would find the application according to application settings in default-web-site.xml on myhost.

> **Note:** The host and port specified in the Oc4jMount command should be the same as the host and port specified for ajp13 protocol in the default-web-site.xml file.

If Oracle Enterprise Manager (that is, the OPMN process) starts OC4J, then in mod_oc4j.conf you can add the following, in which case OPMN will scan all possible ports for a suitable and available AJP port to use. Requests of the form /servlets/admin/* will be directed to one of the OC4J JVMs in the default "home" OC4J instance.

```
Oc4jMount /servlets/admin/*
```

This is in conjunction with the following default settings in the default-web-site.xml file:

```
<web-site port="0" protocol="ajp13" ...>
...
</web-site>
```

You can also specify a particular OC4J instance, or load balancing between clusters, or load balancing between Oracle Application Server instances, as in the following examples:

```
Oc4jMount /servlets/admin/* oc4j_inst1

Oc4jMount /servlets/admin/*  cluster://ias_cluster_1:home,ias_cluster_2:home

Oc4jMount /servlets/admin/* instance://ias_inst_1:home_1,ias_inst_2:home_2
```

For more information about Oc4jMount commands, refer to the *Oracle HTTP Server Administrator's Guide*. For more information about default-web-site.xml and

other Web site XML files, see the *Oracle Application Server Containers for J2EE Servlet Developer's Guide*

3.1.2.5.3 Servlet Aliases and URL Mapping

This section compares the methods of specifying servlet aliases and URL mapping in JServ and in OC4J.

Aliases and URL Mapping in JServ In JServ, servlet aliases are specified in the appropriate zone properties file. For example, for a servlet class `example.extensionmapping.InputServlet`, you can specify the alias `inputservlet` to avoid having to specify the full path when you invoke the servlet. This is done as follows:

```
servlet.inputservlet.code =
example.extensionmapping.InputServlet
```

URL extension mappings, like mount points, are defined in the `jserv.conf` file. Following is an example:

```
ApJServAction .inp
/servlets/example.extensionmapping.InputServlet
```

This results in URL patterns ending in ".inp" being mapped to InputServlet.

Also, for example, assume the following mount command:

```
ApJServMount /servlets /root
```

The following URL, because of this `ApJServMount` command and the `ApJServAction` command, is passed to `example.extensionmapping.InputServlet` in the servlet zone root.

```
http://myhost.mycompany.com/EmployeeInput.inp
```

Aliases and Extension Mapping in OC4J In OC4J, servlet aliases and URL mappings are specified through entries such as the following in the `global-web-application.xml` file:

```
<servlet>
   <servlet-name>inputservlet</servlet-name>
   <servlet-class>example.extensionmapping.InputServlet</servlet-class>
</servlet>
...
<servlet-mapping>
   <servlet-name>inputservlet</servlet-name>
   <url-pattern>/*.inp</url-pattern>
</servlet-mapping>
```

The servlet name (alias) can be anything—it simply serves as a reference name to associate the servlet class with the URL extension that is specified in the `<servlet-mapping>` element.

In an Oracle Application Server environment, do not edit `global-web-application.xml` directly. Perform configuration through Oracle Enterprise Manager.

3.1.2.5.4 Initialization Parameters

This section details the differences between JServ and OC4J in setting servlet initialization parameters.

Setting Initialization Parameters in JServ JServ supports servlet-based initialization parameter settings ("initArgs") as well as zone-wide default parameter settings. These settings are specified in the appropriate zone properties file.

For example, for a servlet `foo1`, you would define a servlet-based setting for the name parameter as follows:

```
servlet.foo1.initArgs=name=scott
```

A zone-wide default setting, shared by all servlets in the zone, is specified as in the following example. This specifies a default setting for the `company` parameter:

```
servlets.default.initArgs=company=oracle
```

A servlet-based setting overrides a zone-wide (default) setting for the same named parameter.

Setting Initialization Parameters in OC4J In OC4J, servlet-based initialization parameter settings are specified through subelements of the `<servlet>` element in the standard `web.xml` file. The following example is equivalent to the example for `foo1` in the preceding section (but also specifies the servlet class):

```
<servlet>
   <servlet-name>foo</servlet-name>
   <servlet-class>FooServlet</servlet-class>
   <init-param>
        <param-name>name</param-name>
        <param-value>scott</param-value>
   </init-param>
</servlet>
```

No equivalent for zone-wide settings in JServ exists in OC4J—there is no mechanism for application-wide default settings. However, `<context-param>` settings are conceptually similar. For each Web application, there is a servlet context. Attributes for the context can be set in the application `web.xml` file, as in the following example.

```
<context-param>
    <param-name>company</param-name>
    <param-value>oracle</param-value>
</context-param>
```

Also similar, in a servlet 2.3 environment, is the use of servlet filters. You can use filters to enforce a global behavior across groupings of servlets based on URL patterns. This is described in the Java Servlet Specification, version 2.3.

3.1.2.6 Servlet Pre-Starting

This section compares servlet pre-start features between the OC4J 9.0.4 implementation and JServ.

3.1.2.6.1 Pre-Started Servlets In both JServ and OC4J, servlets can be pre-started. Rather than having servlet instances created only after the first request arrives, they can be created in advance and pre-started by the servlet container when the container starts up. This reduces the time taken for servicing the first request.

3.1.2.6.2 Pre-Start and Timeout Settings in JServ In JServ, servlets to pre-start are specified in the appropriate zone properties file, as in the following example:

```
servlets.startup=oracle.sample.test1.HelloWorld,foo1
```

This prestarts `HelloWorld` and `foo1`.

Alternatively, you can use an alias name instead of the complete name.

JServ zone properties files also support the following parameters, which have no equivalents in OC4J:

- Parameters to specify the timeout period for initialization (after which the servlet container will stop trying to initialize) and the timeout period after which a servlet is destroyed:

  `init.timeout`

  `destroy.timeout`

- Parameters for the single-threaded model, to specify the number of servlet instances to be created if the servlet implements the `javax.servlet.SingleThreadModel` interface:

  `SingleThreadModelServlet.initialCapacity`

  `SingleThreadModelServlet.incrementCapacity`

  `SingleThreadModelServlet.maximumCapacity`

3.1.2.6.3 Pre-Start Settings in OC4J In a servlet 2.3 environment such as OC4J, servlets can be pre-started according to the `<load-on-startup>` subelement of the `<servlet>` element in the application `web.xml` file, as in the following example:

```
<servlet>
   <servlet-name>HelloWorld</servlet-name>
   <servlet-class>oracle.sample.test1.HelloWorld</servlet-class>
   <load-on-startup/>
</servlet>
```

In OC4J, there must also be a setting of `auto-start="true"` in the appropriate `<application>` element in the `server.xml` file, and a setting of `load-on-startup="true"` in the relevant `<web-app>` subelement of the `<web-site>` element of the appropriate Web site XML file. This is described in the *Oracle Application Server Containers for J2EE Servlet Developer's Guide*.

3.1.2.7 Class Loaders and Automatic Class Reloading

This section discusses servlet class loaders, and class reloading during servlet execution.

3.1.2.7.1 Class Loading in JServ This section discusses class loaders and class reloading in the JServ environment. There is a separate class loader for each of the following:

- system classes: classes found on the system classpath

  For automatic JServ startup, the system classpath is determined by the `wrapper.classpath` setting in the `jserv.properties` file. For manual JServ startup, the system classpath is determined by the `CLASSPATH` setting for the particular JServ instance.

  Classes loaded from the system classpath (including servlet classes) cannot be automatically reloaded without restarting the server.

- zone classes: classes found on the zone classpath

  The zone classpath is specified through the `repositories` parameter in the zone properties file.

Be aware of the following:

- Classes loaded from the system classpath are shared across all zones in the same JVM.
- Each servlet zone has its own instance of a custom class loader, for classes in the zone classpath.

- Classes loaded from the zone classpath are not shared between zones. Furthermore, for a class that is available in different zones, its static variables cannot be shared across zones, even when the zones are in the same JVM. For static variables to be sharable, the class must be in the system classpath, and therefore loaded by the system class loader.

Classes in the zone classpath, and therefore loaded by the zone class loader, can be automatically reloaded if they are modified. This is useful if you are in the process of developing your application, and is determined by the following setting in the zone properties file:

```
autoreload.classes=true
```

3.1.2.7.2 Class Loading in OC4J OC4J classpath and class loader configuration that is equivalent to that of JServ is determined as follows:

- For server-wide class loading, classpath settings are specified through the `<library>` element of the OC4J global `j2ee/home/config/application.xml` file, as in the following example:

  ```
  <library path="/tmp/net.jar" />
  ```

  Relative or absolute paths to directories can be specified, and these directories are scanned for JAR or ZIP files to include in the classpath at startup. By default in the OC4J 9.0.4 implementation, the `application.xml` file specifies the inclusion of files from the `j2ee/home/applib` directory. (A `<library>` element exists for this path in the default global `application.xml` file).

  > **Note:** Do not confuse the OC4J global `application.xml` file with the standard `application.xml` file for each application.

- For application-based class loading, the default classpath consists of the directories `WEB-INF/classes` (for class files, in subdirectories according to package names) and `WEB-INF/lib` (for JAR and ZIP libraries).

  Additional classpath information can be specified through the `<classpath>` element of the `orion-web.xml` file, as in the following example:

  ```
  <classpath path="/private/test/test.jar" />
  ```

However, do not update configuration files directly in an Oracle Application Server environment. Use Oracle Enterprise Manager for configuration.

In an OC4J standalone development environment, you can specify automatic recompilation and reloading of servlets in OC4J by setting the `development` attribute to `true` in the `<orion-web-app>` element of the `j2ee/home/config/global-web-application.xml` file or the application `orion-web.xml` file (which takes precedence over `global-web-application.xml` for a particular application).

In this case, classes in the target directory are automatically reloaded whenever they are modified, or whenever an application-level XML file is modified. This is useful when you are in the process of developing your application.

By default, when automatically reloading, source files (`.java`) are picked up from the target directory `WEB-INF/src` if it exists. If the `src` directory does not exist, then source files are picked up from the `WEB-INF/classes` directory instead. You can specify an alternative target directory through the `source-directory` attribute of

the <orion-web-app> element in the application orion-web.xml file, as in the following abbreviated example:

```
<orion-web-app ... source-directory="/private/scott/myservletsource" ...>
   ...
</orion-web-app>
```

In this case, files are picked up from the specified directory only, not from the src or classes directory.

3.1.2.8 Session Tracking and Behavior

This section discusses session behavior and related configuration in JServ and OC4J.

3.1.2.8.1 Session Tracking and Behavior in JServ JServ provides the following parameters in the zone properties file for specifying session behavior:

- session.useCookies — Specifies whether to use cookies for sessions (default is true). If false, then the encodeUrl() method of the response object is the only means of session-tracking.
- session.timeout — Specifies the number of milliseconds to wait before invalidating a session (default is 1800000, which is 30 minutes).
- session.checkFrequency — Specifies how frequently (in seconds) to check for timed-out sessions. (The default is 30.)

3.1.2.8.2 Session Tracking and Behavior in OC4J For session tracking in OC4J, the servlet container will first attempt to accomplish tracking through cookies. If cookies are disabled, session tracking can be maintained only by using the encodeURL() method of the response object explicitly in the servlet. (The encodeURL() method replaces the servlet 2.0 encodeUrl() method, which has been deprecated.)

It is also possible to specify the number of minutes for OC4J to wait before a session is invalidated. (The default is 20.) The <session-timeout> subelement of the <session-config> element in the application web.xml file can specify this, as follows:

```
<session-config>
   <session-timeout>30</session-timeout>
</session-config>
```

Additionally, the following setting in the global-web-application.xml or orion-web.xml file disables the use of session cookies:

```
<session-tracking cookies="disabled" ... >
   ...
</session-tracking>
```

However, do not update configuration files manually in an Oracle Application Server environment—use Oracle Enterprise Manager for configuration.

Cookies are enabled by default. This is equivalent to the functionality of session.useCookies in JServ. As in the JServ case, with cookies disabled you must explicitly use the encodeURL() method of the response object for session-tracking.

3.1.2.9 Message and Error Logging

This section compares the JServ and OC4J mechanisms for setting up log files.

3.1.2.9.1 Message Logging for JServ Message logging parameters such as the log file name, timestamp format, and other settings are specified through the `jserv.properties` file. This can help with debugging.

Following are the key parameters:

- `log` — Specifies whether or not to log messages (`log=true` by default).
- `log.file` — Specifies the file where log messages are written. (An absolute path name is recommended.)
- `log.timestamp` — Specifies that messages are to be preceded by a timestamp (`log.timestamp=true` by default).
- `log.dateFormat` — Specifies the date format in the timestamps. The setting is `dd/MM/yyyy HH:mm:ss:SSS.zz` by default.
- `log.queue.maxage` — Specifies the maximum time (in milliseconds) that a message can be in the queue.
- `log.queue.maxsize` — Specifies the maximum number of messages in the queue.

In addition, you can use the following parameters to enable different levels of logging:

- `log.channel`
- `log.channel.info`
- `log.channel.ServletException`
- `log.channel.jservException`
- `log.channel.warning`
- `log.channel.servletLog`
- `log.channel.critical`
- `log.channel.debug`

3.1.2.9.2 Message Logging for OC4J Several logs are available in OC4J in Oracle Application Server 10*g*. Because they are not specific to servlets, they are documented elsewhere, but this section provides a summary list and appropriate cross-references. For each log, you have the option of using text-based logging or Oracle Diagnostic Logging (ODL). For ODL, log file names always take the form `logN.xml`, where `N` is an integer. For text-based logging, you must specify the log file names.

For each log there is a configuration element in the appropriate OC4J configuration file to enable text-based logging, and a separate element to enable ODL logging. The presence of a logging configuration element enables the associated type of logging.

OC4J supports the following logs:

- Application log: There is a log for each application deployed, as configured in `orion-application.xml`. For text-based logging, a typical name is `application.log`.
- Global application log: There is a log for global logging for all applications, including the default application, as configured in the OC4J global `application.xml` file. For text-based logging, a typical name is `global-application.log`.
- JMS log: There is a log for Java Message Service (JMS) functionality, as configured in `jms.xml`. For text-based logging, a typical name is `jms.log`.

- RMI log: There is a log for remote method invocation functionality, as configured in `rmi.xml`. For text-based logging, a typical name is `rmi.log`.
- Server log: There is a server-wide log, as configured in `server.xml`. For text-based logging, a typical name is `server.log`.
- Web site access log: There is a Web site access log (one log file for each Web site to log all accesses of the site), as configured in `default-web-site.xml`. For text-based logging, a typical name is `http-access.log`.

> **Note:** For Web site access logging, you can use only one type of logging, not both.

Configuration of the Web access log is covered in the *Oracle Application Server Containers for J2EE Servlet Developer's Guide*. The *Oracle Application Server Containers for J2EE User's Guide* has information about how to enable logging to the other OC4J files.

In addition to the OC4J log files discussed previously, Oracle Application Server supports the following log files:

- OPMN log file (one log file for each OC4J instance, for Oracle Process Management and Notification functionality)
- `ons.log` (OPMN notification system log, configured in `opmn.xml`)
- `ipm.log` (OPMN process management log, configured in `opmn.xml`)

OPMN manages Oracle HTTP Server and OC4J processes within an application server instance. For information about this, refer to the *Oracle Application Server 10g Administrator's Guide*.

3.1.2.10 Load Balancing and Fault Tolerance

This section describes issues and configuration for load balancing and fault tolerance in JServ and OC4J.

3.1.2.10.1 Request Routing and Load Balancing in JServ JServ supports request routing by appending the appropriate JServ instance ID to the session ID when an `HttpSession` object is used.

When an HTTP request is received before a session is started, an arbitrary JServ instance is chosen from the available instances to service the request, and a cookie with the JServ instance ID is sent back to the Web browser (or other HTTP client). Later, when the next request comes from the same session, it is forwarded to the same JServ instance by matching the JServ ID. (If the original JServ instance is down, the request will be forwarded automatically to an alternative instance.)

Given this functionality, `HttpSession` objects in JServ are non-distributable—the session object cannot be distributed between different JServ instances. As a result, a long-lived HTTP session in JServ decreases the flexibility of load balancing. Also, the session data is lost if the corresponding JServ JVM crashes. Fault tolerance is low, because there is no session failover functionality.

3.1.2.10.2 Load Balancing and Fault Tolerance in OC4J OC4J supports clusters of OC4J instances, and clusters can be customized to the specific needs of the users. Through OC4J load balancing, more user traffic can be handled by distributing the request workload to multiple servers within the cluster. OC4J can be configured to replicate the state of each individual node to the cluster. (The state information is not saved to any persistent storage, but is in memory.) Through OC4J fault tolerance, in case of the

failure of a server, a client can automatically be redirected to an alternative server in the cluster. Oracle HTTP Server performs this failover.

In OC4J, HTTP sessions are replicated to other OC4J JVM instances within a load-balanced cluster island. This preserves session state in case of JVM failure, and is a feature that is not available in JServ.

Assuming proper configuration, when the request is routed through another JVM in the cluster island, the session state is available in the other JVM as well. Furthermore, the session state is still available even in individual JVM failure scenarios. The Web application proceeds smoothly.

For this functionality, the Web application must be marked as "distributable" through the `<distributable>` element of the application `web.xml` file. Objects in a distributable `HttpSession` instance must be serializable or remoteable for the replication to work properly.

See the *Oracle Application Server 10g Performance Guide* for more information about OC4J load balancing and fault tolerance.

3.1.2.10.3 Clarification on loadbalancer.jar It is advisable to use `loadbalancer.jar` as a utility to test distribution of load in development and test environments only. It was not designed for production environments and should not be used in production environments.

In production environments, use Oracle HTTP Server, OPMN, and `mod_oc4j` for load balancing.

3.1.2.11 Example: JServ to OC4J Upgrade

This section provides an example of upgrading a Web application from Apache JServ to OC4J, noting both the original JServ configuration settings and the OC4J configuration settings. This example does not necessarily reflect a typical or optimal scenario; it is merely for illustrative purposes.

The example includes two servlets (source files `HelloWorldServlet.java` and `SessionServlet.java`), two JSP pages (`Hello.jsp` and `snoop.jsp`), and accompanying `.gif` and `index.html` files.

3.1.2.11.1 JServ Configuration This section describes the pre-upgrade JServ directory structure and configuration files.

JServ Directory Structure The JServ directory structure is shown below:

```
/private/scott-sun/upgrade-example/
                  index.html
                  classes/
                           HelloWorldServlet.java
                           SessionServlet.java
                  jsps/
                           Hello.jsp
                           snoop.jsp
                  examples/
                           index.html
                           images/
                                  blk_line_bullet_35.gif
                                  red_arrow_bullet_35.gif
```

JServ Configuration Files The following JServ configuration files would have related entries. (The entries for each file are described in the subsections immediately following.)

```
APACHE_HOME/apache/conf/jserv/jserv.conf
                            jserv.properties
                            zone.properties
```

Entries in the jserv.conf File In this example, the `jserv.conf` file includes the following entries:

```
Alias  /migdemos /private/scott-sun/upgrade-example
ApJServMount /servlet  /root
```

Entries in the jserv.properties File In this example, the `jserv.properties` file includes the following entries:

```
zones=root

# Configuration file for each servlet zone (one per servlet zone)
# Syntax: [servlet zone name as on the zones list].properties=
#                                     [full path to configFile] (String)
# Default: NONE
# Note: if the file could not be opened, try using absolute paths.
root.properties=/private/scott-sun/apache/conf/jserv/zone.properties
```

Entries in the zone.properties File In this example, the `zone.properties` file includes the following entries:

```
# List of Repositories
########################

# The list of servlet repositories controlled by this servlet zone
# Syntax: repositories=[repository],[repository]...
# Default: NONE
# Note: The classes you want to be reloaded upon modification should be put
# here.
repositories=/private/scott-sun/upgrade-example/classes
```

URLs for Invocation in JServ Assuming the preceding configuration, you could use the following URLs (specifying the appropriate port) to directly invoke the various pages.

To invoke the servlets:

```
http://scott-sun:port/servlet/HelloWorldServlet
http://scott-sun:port/servlet/SessionServlet
```

To invoke the JSP pages:

```
http://scott-sun:port/migdemos/jsps/Hello.jsp
http://scott-sun:port/migdemos/jsps/snoop.jsp
```

To invoke the index HTML pages:

```
http://scott-sun:port/migdemos/index.html
http://scott-sun:port/migdemos/examples/index.html
```

3.1.2.11.2 OC4J Configuration This section describes the OC4J directory structure and configuration files. For this discussion, assume an OC4J standalone environment for initial development.

Initially, of course, the EAR file (`upgrade-example.ear`) and WAR file (`upgrade-example.war`) would not yet exist. You would create them after the rest of the directory structure had been established. They are shown here within the directory structure, with contents nested beneath them, for illustrative purposes.

OC4J Directory Structure The OC4J directory structure is shown below:

```
/private/scott-sun/upgrade-example/
                    upgrade-example.ear

                      META-INF/
                               application.xml
                      upgrade-example.war
                          index.html
                          WEB-INF/
                                  web.xml
                                  orion-web.xml

                                  classes/
                                           HelloWorldServlet.java
                                           SessionServlet.java

                                  jsps/
                                           Hello.jsp
                                           snoop.jsp

                                  examples/
                                            index.html

                                            images/
                                                   blk_line_bullet_35.gif
                                                   red_arrow_bullet_35.gif
```

The WAR file has the following structure:

```
META-INF/
META-INF/MANIFEST.MF
WEB-INF/
WEB-INF/classes/
WEB-INF/classes/HelloWorldServlet.java
WEB-INF/classes/SessionServlet.java
WEB-INF/web.xml
WEB-INF/orion-web.xml
jsps/
jsps/Hello.jsp
jsps/snoop.jsp
examples/
examples/index.html
examples/images/
examples/images/blk_line_bullet_35.gif
examples/images/red_arrow_bullet_35.gif
index.html
```

The EAR file has the following structure:

```
META-INF/
META-INF/MANIFEST.MF
```

```
META-INF/application.xml
upgrade-example.war
```

Deploy the EAR file according to OC4J deployment conventions. See the *Oracle Application Server Containers for J2EE User's Guide* and the *Oracle Application Server Containers for J2EE Servlet Developer's Guide* for more information.

Some points to consider for OC4J upgrade and deployment:

- OC4J does not require that servlets be recompiled, but it might be useful to recompile in order to discover any deprecated servlet 2.0 methods in use. (The servelts with deprecated methods would still run in OC4J.)
- In a servlet 2.0 environment, there are no separate servlet contexts—all servlets are grouped together. In a servlet 2.2 or 2.3 environment, you are deploying into a particular servlet context, per the WAR file. Interdependencies between servlets that work effectively together in JServ, but are in different contexts in OC4J, will cause runtime problems in OC4J.
- Place application-specific library or utility JAR files in the `WEB-INF/lib` directory.
- In Oracle Application Server 10*g* (9.0.4) in particular, place system-wide library or utility JAR files in the `j2ee/home/applib` directory, where they will be accessible to all applications.

System-wide path settings, such as for class file and JAR file locations, are specified in the global `application.xml` file. This is where, for example, `j2ee/home/applib` is set as a server-wide library location.

Entries in the application.xml File In this example, the `application.xml` file includes the following entries:

```
<?xml version="1.0"?>
<!DOCTYPE application PUBLIC "-//Sun Microsystems, Inc.//DTD J2EE Application 1.
2//EN" "http://java.sun.com/j2ee/dtds/application_1_2.dtd">
<application>
  <display-name>OC4J demo application</display-name>
  <description>
     Collection of servlet samples.
  </description>
  <module>
    <web>
      <web-uri>upgrade-example.war</web-uri>
      <context-root>/migdemos</context-root>
    </web>
  </module>
</application>
```

> **Note:** Each application has its own standard `application.xml` file containing application-specific settings. Do not confuse this with the OC4J global `application.xml` file mentioned earlier, which contains server-wide settings.

Entries in the orion-web.xml File In this example, the `orion-web.xml` file includes the following entries:

```
<?xml version="1.0"?>
<!DOCTYPE orion-web-app PUBLIC "-//Evermind//DTD Web Application 2.3//EN"
```

```
"http://xmlns.oracle.com/ias/dtds/orion-web.dtd">

<orion-web-app
        deployment-version="OracleAS 9.0.4"
        servlet-webdir="/servlet"
>
</orion-web-app>
```

Entries in the web.xml File In this example, the `web.xml` file includes the following entries:

```
<?xml version="1.0"?>
<!DOCTYPE web-app PUBLIC "-//Sun Microsystems, Inc.//DTD Web Application 2.2//EN
" "http://java.sun.com/j2ee/dtds/web-app_2_2.dtd">

<web-app>
        <!-- A demo servlet, add servlets below -->
        <servlet>
                <servlet-name>HelloServlet</servlet-name>
                <servlet-class>HelloWorldServlet</servlet-class>
        </servlet>
        <welcome-file-list>
                <welcome-file>index.html</welcome-file>
        </welcome-file-list>
</web-app>
```

URLs for Invocation in OC4J Assuming the preceding configuration, you could use the following URLs (specifying the appropriate port) to directly invoke the various pages.

To invoke the servlets:

```
http://scott-sun:port/migdemos/servlet/HelloWorldServlet
http://scott-sun:port/migdemos/servlet/SessionServlet
```

(The `servlet-webdir` setting in `orion-web.xml` is relevant here.)

To invoke the JSP pages:

```
http://scott-sun:port/migdemos/jsps/Hello.jsp
http://scott-sun:port/migdemos/jsps/snoop.jsp
```

To invoke the index HTML pages:

```
http://scott-sun:port/migdemos/index.html
http://scott-sun:port/migdemos/examples/index.html
```

3.1.3 Oracle Data Source Considerations

There are a number of considerations in upgrading from Oracle9*i*AS Release 1 (1.0.2.2.x) to Oracle Application Server 10*g* (9.0.4) regarding data sources, including features supported in the 9.0.4 implementation that were not supported in the 1.0.2.2 implementation. Consider the points that follow.

> **Note:** Refer to the 10*g* (9.0.4) release of the *Oracle Application Server Containers for J2EE Services Guide* for detailed information about current data source features.

- For emulated data sources, always use `ejb-location` to specify the JNDI name. Also be aware that although `location` and `xa-location` are ignored, they still must be present in the `<data-source>` element of `data-sources.xml`, for reasons relating to backward compatibility. (The requirement of specifying `xa-location` was not clarified in the Oracle9*i*AS Release 1 (1.0.2.2.x) documentation.)
- Non-emulated data sources and two-phase commit, which were not supported in the 1.0.2.2 implementation, are supported in the 10*g* (9.0.4) implementation. Specify only `location`; do not use `ejb-location`, `pooled-location`, or `xa-location` for non-emulated data sources.
- Native data sources are also supported in the 10*g* (9.0.4) implementation.
- For EJB CMP functionality, you can now associate a database schema with a data source. This is configured in a database schema XML file—referenced through the schema attribute of the `<data-source>` element—that you can customize for the database you are using, as desired. (When using CMP, the EJB container is responsible for creating the database schema necessary to persist a bean. Associating a data source with a database schema XML file allows you to influence the SQL that is ultimately generated by the container. This can help you solve problems such as accommodating additional data types supported in your application but not in your database.)
- The data source implementation now supports Real Application Clusters (RAC), through the `rac-enabled` attribute of the `<data-source>` element.
- The data source implementation now supports High Availability, through features described in the Oracle Application Server 10*g* (9.0.4) documentation.
- Password specification in `data-sources.xml` can now be obfuscated.
- All attributes of the `<data-source>` element are supported in the 9.0.4 implementation.
- The Merant drivers mentioned in the Oracle9*i*AS Release 1 (1.0.2.2.x) documentation have been renamed "DataDirect".

Note: In an Oracle Application Server environment, use Enterprise Manager to configure data sources. If you modify any XML configuration files directly, including `data-sources.xml`, you must use the `dcmctl` utility to ensure that Enterprise Manager is aware of any changes. This is described at the beginning of Section 3.1, "Upgrade Considerations for OC4J Applications" on page 3-1.

3.1.4 OracleAS JAAS Provider Considerations

This section discusses considerations for customers who will use Oracle Application Server Java Authentication and Authorization Service in Oracle Application Server 10*g* (9.0.4).

3.1.4.1 Upgrading the principals.xml File to OracleAS JAAS Provider

In Oracle9*i*AS Release 1 (1.0.2.2.x) OC4J security services, the `principals.xml` file defined users and groups for mapping to roles defined in application deployment descriptors.

In Oracle Application Server 10*g* (9.0.4), security services are provided through the Java Authentication and Authorization Service (JAAS). For information about JAAS,

see the *Oracle Application Server 10g Security Guide*. The `principals.xml` file is still supported for backward compatibility.

The JAAS Admintool is provided for security administrators to manage users, realms, roles and policies. It has a command switch that allows you to migrate `principals.xml` to a JAAS realm, which is generally advisable. The syntax is:

```
java -jar jazn.jar -convert <filename> <realm>
```

where

`filename` is the name and location of the OC4J `principals.xml` file and *`realm`* is the realm defined in the JAAS. For example:

```
java -jar jazn.jar -convert /home/config/principals.xml
MyCompanyRealm
```

All permissions granted to a `principals.xml` group are granted to the respective JAAS role. Users that were deactivated are not upgraded. See Chapter 5 of the *Oracle Application Server 10g Security Guide* for additional usage information.

3.1.5 Using the Compatibility Test Suite (CTS) Flag for Backward Compatibility

In Oracle Application Server 10*g* (9.0.4), OC4J by default complies with the J2EE 1.3 specification. In some cases, this results in behavior that differs from that seen with previous OC4J implementations. To allow for backward compatibility, OC4J supports a CTS compliance flag that you can set to false to revert to previous OC4J behavior in the following components:

- Oracle JMS
- Oracle JDBC
- Oracle XML parser for JAXP/XDK

The compliance behavior of OC4J is determined by the flag `oracle.cts.useCtsFlags`, with a default value of `true`. If any of the upgrade issues are critical in a particular application, you can disable CTS compliance and revert to old behavior for an OC4J instance by setting the flag value to `false` in an OC4J properties file and providing the location of the properties file to OC4J.

For example, the file <*`destination_MT_OH`*`/j2ee/home/config/oc4j.properties` might contain the setting:

```
oracle.cts.useCtsFlags=false
```

Supply the name and location of a properties file to OC4J through an `<oc4j-option>` element in the <*`destination_MT_OH`*`/opmn/conf/opmn.xml` file, as in the following example:

```
<oc4j>
...
 <oc4j-option value="-p
 destination_MT_OH/j2ee/home/config/oc4j.properties"/>...
</oc4j>
```

This is equivalent to starting OC4J as follows in standalone mode with the following command:

```
java -jar oc4j.jar -p <destination_MT_
OH>/j2ee/home/config/oc4j.properties
```

Note: In an Oracle Application Server environment, use Enterprise Manager to make any configuration changes. If you must update any configuration files directly, use the `dcmctl` utility so that Enterprise Manager is aware of the changes, as described at the beginning of Section 3.1, "Upgrade Considerations for OC4J Applications" on page 3-1.

3.1.5.1 CTS Compatibility and Oracle JMS (OJMS)

In the Oracle Application Server 10*g* (9.0.4) implementation of (OJMS), which complies with J2EE 1.3, some behavior differs from OJMS behavior in Oracle9*i*AS Release 1 (1.0.2.2.x). (There are no such upgrade considerations between Oracle9*i*AS Release 2 (9.0.2) and Oracle Application Server 10*g* (9.0.4).) The differences are identified in Table 3–2.

Table 3–2 OJMS Behavior in Oracle9iAS Release 1 (1.0.2.2.x) and Oracle Application Server 10g (9.0.4)

Item	Behavior in Oracle9*i*AS Release 1 (1.0.2.2.x) OJMS Implementation	Behavior in Oracle Application Server 10*g* (9.0.4) OJMS Implementation
`JMSExpiration`	The `JMSExpiration` header value in a dequeued message is the duration until expiration of the message, in milliseconds. If a message never expires, the value is -1.	The `JMSExpiration` header value in a dequeued message is the sum of the JMS timestamp when the message was enqueued, and the time-to-live. This value is expressed in milliseconds from midnight, January 1, 1970 to the current Greenwich Mean Time. If a message never expires, the value is 0.
`JMSPriority`	In this implementation, `java.lang.Integer.MIN_VALUE` is the highest priority, `Integer.MAX_VALUE` is the lowest priority, and 1 is the default priority.	In this implementation, 9 is the highest priority, 0 is the lowest priority, and 4 is the default priority.
Durable Subscribers	Durable Topic Subscribers with the same name are allowed if they are subscribed to different topics.	Durable Topic Subscribers with the same name are not allowed under any circumstances.
Strongly typed JMS selectors	Not subject to the restrictions listed for 10*g* (9.0.4), or to the limited subset of SQL92 syntax for selector expression syntax.	In compliance with the JMS 1.02b specification and J2EE 1.3 compliance requirements, the OJMS 9.0.4 implementation uses only a certain subset of SQL92 syntax for selector expression syntax, with the following mandated restrictions: ▪ Selector expressions are strongly typed, meaning operators and operands in arithmetic comparisons must be of the same type. Automatic type conversions for the purpose of comparison, such as converting the string "1" to the integer 1, are prohibited. ▪ String and boolean comparisons are restricted to "=", "<", and ">". Two strings are equal only if they contain the exact same sequence of characters. ▪ The "!=" operator is prohibited.

3.1.5.2 CTS Compatibility and JDBC

In the Oracle Application Server 10*g* (9.0.4) implementation of Oracle JDBC, which complies with J2EE 1.3, some behavior differs from JDBC behavior in previous releases. The differences are identified in Table 3–3.

Table 3–3 JDBC Behavior in Oracle9iAS Release 1 (1.0.2.2.x) and Oracle Application Server 10g (9.0.4)

Item	Behavior in Oracle9*i*AS Release 1 (1.0.2.2.x) JDBC Implementation	Behavior in Oracle Application Server 10*g* (9.0.4) JDBC Implementation
Java types for `NUMBER` columns	`getObject()` returns a `BigDecimal` value for any `NUMBER` column. (This is also true for Release 2 (9.0.2).)	The `getObject()` method of a result set (`java.sql.ResultSet` instance) returns a `java.lang.Double` value for a `NUMBER` column with precision, or a `java.math.BigDecimal` value for a `NUMBER` column without precision.
Metadata for `NUMBER` columns	`getColumnTypeName()` returns "`NUMBER`" for any `NUMBER` column, and `getColumnType()` returns `Types.NUMBER` for any `NUMBER` column. (This is also true for Release 2 (9.0.2.)	The `getColumnTypeName()` method of a result set metadata object (`java.sql.ResultSetMetaData` instance) returns "`FLOAT`" for a `NUMBER` column with precision, or "`NUMBER`" for a `NUMBER` column without precision. The `getColumnType()` method returns `java.sql.Types.FLOAT` for a `NUMBER` column with precision, or `Types.NUMBER` for a `NUMBER` column without precision.
Java types for `DATE` and `TIMESTAMP` columns	`getObject()` returns a `java.sql.Timestamp` value for a `DATE` column. (`TIMESTAMP` columns were not supported.)	The `getObject()` method of a result set returns a `java.sql.Date` value for a `DATE` column, and a `java.sql.Timestamp` value for a `TIMESTAMP` column.
Exceptions for inappropriate SQL statements	Inappropriate SQL statements do not result in exceptions.	If an `executeQuery()` call in a statement object contains anything but a `SELECT` statement (such as if it instead contains an `INSERT` or `UPDATE` statement), the JDBC driver properly throws an exception. Similarly, if an `executeUpdate()` call contains a `SELECT` statement, the driver properly throws an exception. (An `UPDATE`, `INSERT`, or `DELETE` statement is expected.)

3.1.5.3 CTS Compatibility and the JAXP/XDK XML Parser

In the Oracle Application Server 10*g* (9.0.4) implementation of the XML parser for JAXP/XDK, which complies with J2EE 1.3, some behavior differs from XML parser behavior in previous releases. The key differences are described in Table 3–4:

Table 3–4 JAX/XDK XML Parser Behavior in Oracle9iAS Release 1 (1.0.2.2.x) and Oracle9iAS Release 2 (9.0.2) versus Oracle Application Server 10g (9.0.4)

Item	Behavior in Oracle9*i*AS Release 1 (1.0.2.2.x) and Oracle9*i*AS Release 2 (9.0.2) XML Parser Implementation	Behavior in Oracle Application Server 10*g* (9.0.4) XML Parser Implementation
`getNamespaceURI()` null return values	The `getNamespaceURI()` method returns `""""` if the namespace is not defined for an element or attribute.	The `getNamespaceURI()` method returns `null` if the namespace is not defined for an element or attribute.
`getLocalName()` null return values	The `getLocalName()` method returns `""""` if the element or attribute was created using a DOM level 1 API call to `createElement()` or `createAttribute()`.	The `getLocalName()` method returns `null` if the element or attribute was created using a DOM level 1 API call to `createElement()` or `createAttribute()`.
`getPrefix()` null return values	The `getPrefix()` method returns `""""` if the element or attribute was created using a DOM level 1 API call to `createElement()` or `createAttribute()`.	The `getPrefix()` method returns `null` if the element or attribute was created using a DOM level 1 API call to `createElement()` or `createAttribute()`. **Note:** The `getNamespaceURI()`, `getLocalName()`, and `getPrefix()` methods exist with the above changes in the `XMLElement` and `XMLAttr` classes of the `oracle.xml.parser.v2` package.

Table 3–4 (Cont.) JAX/XDK XML Parser Behavior in Oracle9iAS Release 1 (1.0.2.2.x) and Oracle9iAS Release 2 (9.0.2) versus Oracle Application Server 10g (9.0.4)

Item	Behavior in Oracle9*i*AS Release 1 (1.0.2.2.x) and Oracle9*i*AS Release 2 (9.0.2) XML Parser Implementation	Behavior in Oracle Application Server 10*g* (9.0.4) XML Parser Implementation
SAX exceptions	Error handlers throw an `XMLParseException` in error conditions.	Registered error handlers throw a `SAXException` or `SAXParseException` in error conditions.
I/O exceptions	An `IOException` is wrapped in an `XMLParseException`.	An `IOException` is thrown as is in I/O error conditions.
Insignificant white space	Insignificant white space was preserved only by explicit request, by the method call `XMLParser.setPreserveWhitespace(true)`.	The default behavior of the XML parser regarding the preservation of insignificant white space depends on the presence of a DTD in the XML document being parsed. If a DTD is present, then insignificant white space is not preserved. To ensure the same behavior with the 9.0.4 implementation as with the 1.0.2.2 or 9.0.2 implementation, add the `XMLParser.setPreserveWhitespace()` call to the code.

3.1.6 Upgrade Considerations for Enterprise Java Beans

In Oracle Application Server 10*g* (9.0.4), OC4J complies with the J2EE 1.3 specification and implements the Enterprise Java Beans (EJB) 2.0 specification in its entirety. Therefore, if you are upgrading to the OC4J 9.0.4 implementation, applications using EJB features in the areas of container-managed persistence and container-managed relationships will require modification.

See *Oracle Application Server Containers for J2EE Enterprise JavaBeans Developer's Guide*, Appendix C.

3.1.7 JDK 1.4 Issues: Cannot Invoke Classes Not In Packages

Among the upgrade considerations in moving to a Sun Microsystems JDK 1.4 environment, which is the environment that is shipped with Oracle Application Server 10*g* (9.0.4), there is one of particular importance to servlet and JSP developers.

As stated by Sun Microsystems, "The compiler now rejects import statements that import a type from the unnamed namespace." (This was to address security concerns and ambiguities with previous JDK versions.) Essentially, this means that you cannot invoke a class (a method of a class) that is not within a package. Any attempt to do so will result in a fatal error at compilation time.

This especially affects JSP developers who invoke JavaBeans from their JSP pages, as such beans are often outside of any package (although the JSP 2.0 specification now requires beans to be within packages, in order to satisfy the new compiler requirements). Where JavaBeans outside of packages are invoked, JSP applications that were built and executed in an earlier environment will no longer work in an OC4J 9.0.4 / JDK 1.4 environment.

Until you update your application so that all JavaBeans and other invoked classes are within packages, you have the alternative of reverting back to a JDK 1.3.1 environment to avoid this issue.

Notes:

- The `javac -source` compiler option is intended to allow JDK 1.3.1 code to be processed seamlessly by the JDK 1.4 compiler, but this option does not account for the "classes not in packages" issue.
- Only the JDK 1.3.1 and JDK 1.4 compilers are supported and certified by OC4J. It is possible to specify an alternative compiler through a `<java-compiler>` element in the `server.xml` file, and this might provide a workaround for the "classes not in packages" issue, but no other compilers are certified or supported by Oracle for use with OC4J. (Furthermore, do not update the `server.xml` file directly in an Oracle Application Server environment. Use Oracle Enterprise Manager.)

For more information about the "classes not in packages" issue and other JDK 1.4 compatibility issues, refer to the following Web site:

`http://java.sun.com/j2se/1.4/compatibility.html`

The link "Incompatibilities Between Java 2 Platform, Standard Edition, v1.4.0 and v1.3" specifically addresses this issue.

3.2 OC4J Upgrade Items

An upgrade item is an aspect of configuration that is acted upon during upgrade: a file, executable, or a setting that you must add, change, or delete/replace in the 10g (9.0.4) installation. OC4J upgrade items are described in Table 3–5.

Table 3–5 OC4J Upgrade Items

Upgrade Item	Location in Release 1 (1.0.2.2.x) Oracle home	Location in 10*g* (9.0.4) Oracle home	Notes
`data-sources.xml`	`j2ee/home/config`	`j2ee/home/config`	Contains data sources for database connections.
`principals.xml`	`j2ee/home/config`	`j2ee/home/config`	The `principals.xml` file defines users and groups for mapping to roles that are defined in application deployment descriptors. It is still supported for backward compatibility, so you may continue using it; upgrade it as described in Section 3.3, "OC4J Upgrade Tasks" on page 3-43. Otherwise, to use the OracleAS JAAS Provider instead, follow the instructions in Section 3.1.4.1, "Upgrading the principals.xml File to OracleAS JAAS Provider" on page 3-36.
J2EE applications (`*.ear` files) See Section 3.1, "Upgrade Considerations for OC4J Applications" on page 3-1 for details on upgrading applications, including the structure and deployment of EAR files.	`j2ee/home/applications`	`j2ee/home/applications`	The `server.xml` file identifies the applications defined in the OC4J home instance. To upgrade these applications, you deploy them into Oracle Application Server 10*g* (9.0.4).
`jms.xml` file queues, topics, and connection factories	`j2ee/home/config/jms.xml`	`j2ee/home/config/jms.xml`	
`application.xml` file entries (e.g., library paths, Java options, OC4J options)	`j2ee/<name of OC4J instance>/config/application.xml`	`j2ee/<name of OC4J instance>/config/application.xml`	
`opmn.xml` file customizations made for OC4J instances that were created by the installer	`opmn/conf/opmn.xml`	`opmn/conf/opmn.xml`	Examples of such instances are `home`, `OC4J_WIRELESS`, `OC4J_DEMOS`, `OC4J_PORTAL`, and `OC4J_BI_FORMS`.

3.3 OC4J Upgrade Tasks

After you have read Section 3.1, "Upgrade Considerations for OC4J Applications" beginning on page 3-1, addressing any issues that apply to the applications you will upgrade, and have created new application EAR files, you are ready to perform the OC4J upgrade.

1. Copy <*source_MT_OH*>/j2ee/home/config/principals.xml to <*destination_MT_OH*>/j2ee/home/config.
2. Copy <*source_MT_OH*>/j2ee/home/config/data-sources.xml to <*destination_MT_OH*>/j2ee/home/config.
3. Examine the <*source_MT_OH*>/j2ee/home/config/server.xml file, noting the paths to each application EAR file you will upgrade.
4. Deploy each application to the corresponding OC4J home instance, using Oracle Enterprise Manager or the dcmctl deployApplication command.
   - For instructions on using Oracle Enterprise Manager to deploy an application, see *Oracle Application Server Containers for J2EE User's Guide*
   - For instructions on using the dcmctl utility to deploy an application, see *Distributed Configuration Management Reference Guide*.
5. Copy any queues, topics, and connection factories from <*source_MT_OH*>j2ee/home/config/jms.xml to <*destination_MT_OH*>j2ee/home/config/jms.xml.
6. Copy any library paths, Java options, and OC4J options from <*source_MT_OH*>/j2ee/<*name of OC4J instance*>/config/application.xml to <*destination_MT_OH*>j2ee/<*name of OC4J instance*>/config/application.xml.
7. Copy any customizations that were made to OC4J instances created by the installer from <*source_MT_OH*>/ompn/conf/opmn.xml to <*destination_MT_OH*>/opmn/conf/opmn.xml.
8. Ensure that the context root defined in <*source_MT_OH*>/j2ee/home/config/default-web-site.xml is the same as that defined in <*destination_MT_OH*>/j2ee/home/config/default-web-site.xml. If the context root values are different, then you can do one of the following:
   - Edit <*destination_MT_OH*>/j2ee/home/config/default-web-site.xml to use the same value defined in <*source_MT_OH*>/j2ee/home/config/default-web-site.xml. In addition, edit <*destination_MT_OH*>/Apache/Apache/conf/mod_oc4j.conf to use the context root defined in default-web-site.xml. The problem arises because the original context root defined in the EAR file is different from that used in <*source_MT_OH*>/j2ee/home/config/default-web-site.xml. You must also use the dcmctl updateconfig command so that the change is reflected in the repository.
   - Use the context root defined in <*destination_MT_OH*>/j2ee/home/config/default-web-site.xml (which is the original context root defined in the EAR file) to access the application. This means that you will have to change the URL used in Oracle9*i*AS Release 1 (1.0.2.2.x) to access the application. For example, to access the PetStore demo in Oracle9*i*AS Release 1 (1.0.2.2.x), the URL is:

`http://<host>:<port>/petstore`

After upgrading to Oracle Application Server 10*g* (9.0.4), you would use this URL to access the application:

`http://<host>:<port>/estore`

3.4 Validating the OC4J Upgrade

Follow the steps below to verify that the OC4J applications that were upgraded are functioning correctly.

1. Start the Oracle Application Server with these commands:

   a. **`<destination_MT_OH>/opmn/bin/opmnctl startall`**

      Oracle Process Management and Notification and all of the processes it manages are started (i.e., Distributed Configuration Management, Oracle HTTP Server, OC4J instances, Oracle Application Server Web Cache, Oracle Application Server Forms Services, and Oracle Application Server Reports Services).

   b. **`<destination_MT_OH>/bin/emctl start iasconsole`**

      The Oracle Enterprise Manager Application Server Control is started.

2. In a browser, access the Oracle Enterprise Manager Application Server Control in the 10g (9.0.4) middle tier Oracle home by entering its URL. Ensure that you provide the correct host name and port number. For example:

   http://midtierhost.mycompany.com:1810

   The **Oracle Enterprise Manager** page appears. A link for the middle tier instance appears in the **Standalone Instances** section.

3. Click the link.

   The **System Components** page appears.

4. Verify that the components are running (a green upward arrow appears in the **Status** column).

5. Verify that the configuration information for the components in use is reflected in the 10*g* (9.0.4) Oracle home.

6. Access each application URL, and verify that the application can perform all of the functions specified in its configuration (you should have a prepared system test for each application).

4

Upgrading to OracleAS Web Cache

This chapter contains step-by-step instructions for upgrading Oracle9*i*AS Web Cache. It contains the following sections:

4.1 OracleAS Web Cache Upgrade Items

An upgrade item is an aspect of configuration that is acted upon during upgrade: a file, executable, or a setting that you must add, change, or delete/replace in the 10g (9.0.4) installation. Oracle Application Server Web Cache upgrade items are described in Table 4–1.

Table 4–1 OracleAS Web Cache Upgrade Items

Upgrade Item	Location in Release 1 (1.0.2.2.x) Oracle home	Location in 10*g* (9.0.4) Oracle home	Notes
`webcache.xml`	`webcache/`	`webcache/`	To upgrade, it is necessary to edit the `webcache.xml` file in the new location, due to differences in the names of configuration file elements. Do not copy the file from the source Oracle home to the destination Oracle home. You will use the Administration user interface to edit the `webcache.xml` file. For more information, see the *Oracle Application Server Web Cache Administrator's Guide* in the Oracle Application Server documentation Library.

4.2 Components Related to OracleAS Web Cache

OracleAS Web Cache works in concert with Oracle HTTP Server and Oracle Application Server Portal to receive and process requests to the application server. These components must be provided with certain host name and port values in order to function together. Specifically:

- OracleAS Web Cache needs to access the origin servers. To do this, it must be provided with the Oracle HTTP Server host name and port.
- OracleAS Portal must be provided with the OracleAS Web Cache host name, listening port, and invalidation port.

For information about the relationship between Oracle HTTP Server port values and OracleAS Web Cache port values, see Section 2.2.2, "Oracle Application Server Web Cache" on page 2-4.

4.3 OracleAS Web Cache Upgrade Tasks

This section provides instructions for upgrading each item identified in Section 4.1, "OracleAS Web Cache Upgrade Items". It is divided into the following subsections:

4.3.1 Configuring OracleAS Web Cache

The OracleAS Web Cache component is not automatically configured by the installation unless the Web Cache component was selected during installation.

If Web Cache was not selected during installation and the configuration steps below were not performed after the installation, Web Cache will not appear in the list of components in the **System Components** page Oracle Enterprise Manager. In that case, you must configure it before upgrading by performing the following steps:

1. Start the Oracle Application Server 10*g* (9.0.4) instance and access the Oracle Enterprise Manager Application Server Control home page. Use the Oracle Enterprise Manager Application Server Control to configure OracleAS Web Cache:
   a. On the **System Components** page, click **Configure Component**.
   b. On the resulting page, select **Web Cache** from the drop-down list and click **Continue**.

      A progress message appears as OracleAS Web Cache is configured, then the message "Component Web Cache has been configured."
   c. Click **OK**.

      The **System Components** page reappears. Web Cache now appears in the list of components. The default configuration, set during installation, is in effect.

4.3.2 Accessing the Web Cache Manager

From the The Oracle Enterprise Manager Application Server Control, you can access the **Web Cache Manager**, the graphical user interface provided for editing the configuration stored in the `webcache.xml` file. Follow these steps to access the **Web Cache Manager** from the System Components page:

1. In the **Administration** section, click the **Web Cache Administration** link.

The Web Cache Administrator password dialog appears.

2. Enter the `ias_admin` password.

   The **Web Cache Manager** page appears. A scrollable frame on the left side of the window contains groups of configuration elements. To access an element, click on its link. The content area of the page is then populated with the values for that element.

4.3.3 Upgrading Ports

Follow these steps to upgrade the administration, invalidation, statistics, and listen ports.

1. Open the `webcache.xml` file in the Oracle9*i*AS Release 2 (1.0.2.2.x) Oracle home.
2. Locate the values for the administration, invalidation, and statistics ports in the Release 2 (1.0.2.2.x) `webcache.xml` file.
3. In the Web Cache Manager page frame, locate the **Ports** section in the frame and click the **Operations Ports** link.

   The content area is populated with the available caches.
4. Select the radio button for the cache you want and click **Edit Selected**.

   The **Edit Operations Ports** window opens.
5. Set the **ADMINISTRATION** port number to the value specified in the Oracle9*i*AS Release 2 (1.0.2.2.x) `webcache.xml` file.
6. Set the **INVALIDATION** port number to the value specified in the Oracle9*i*AS Release 2 (1.0.2.2.x) `webcache.xml` file.
7. Set the **STATISTICS** port number to the value specified in the Oracle9*i*AS Release 2 (1.0.2.2.x) `webcache.xml` file.
8. Click **Submit**.

   The **Web Cache Manager** page reappears.
9. Click **Apply Changes**.

   The page refreshes with a message to restart Web Cache.
10. Click **Restart**.
11. Locate the **Ports** section in the frame and click **Listen Ports**.

    The **Listen Ports** page appears.
12. Select the radio button for the cache you want and click **Edit Selected**.

    The **Edit/Add Listen Port** window opens.
13. Set the **LISTEN** port number to the value specified in the Oracle9*i*AS Release 2 (1.0.2.2.x) `webcache.xml` file.
14. Click **Submit**.

    The **Web Cache Manager** page reappears.
15. Click **Apply Changes**.

    The page refreshes with a message to restart Web Cache.

16. Click **Restart**.

    A dialog box opens with a message that the cache was stopped and started.

17. Click **OK**.

4.3.4 Upgrading Rules

Follow these steps to upgrade the rules for cacheability, session caching, expiration, and multi-version cookies.

1. Open the `webcache.xml` file in the Oracle9*i*AS Release 2 (1.0.2.2.x) Oracle home.
2. Locate the rules in the Release 2 (1.0.2.2.x) `webcache.xml` file.
3. In the **Web Cache Manager** page frame, locate the **Rules for Caching, Personalization, and Compression** section in the frame and click the **Caching, Personalization, and Compression Rules** link.

   The content area is populated with the available caching rules.

4. Select the radio button for the cache you want and click **Edit Selected**.

   The **Edit/Add Caching, Personalization, and Compression Rule** window opens.

5. Enter the values and click **Submit**.

   The **Web Cache Manager** page reappears.

6. Click **Apply Changes**.

   The page refreshes with a message to restart Web Cache.

7. Click **Restart**.
8. In the **Web Cache Manager** page frame, locate the **Rules for Caching, Personalization, and Compression** section in the frame and click the **Session Definitions** link.

   The content area is populated with the session definitions for the current site.

9. Select the site you want to configure from the drop-down list.
10. Select the radio button for the session you want and click **Edit Selected**, or click **Add** to add a session.

    The **Edit/Add Session Definition** window opens.

11. Edit or add the values for the session and click **Submit**.

    The **Web Cache Manager** page reappears.

12. Click **Apply Changes**.

    The page refreshes with a message to restart Web Cache.

13. Click **Restart**.

    A dialog box opens with a message that the cache was stopped and started.

14. Click **OK**.
15. To upgrade additional sessions, click the **Session Definitions** link to populate the content area with caches again, and repeat Steps 4 through 13.

4.3.5 Upgrading Application Server Hosts

Follow these steps to create hosts for the application web servers used in Oracle9*i*AS Release 2 (1.0.2.2.x).

1. Open the `webcache.xml` file in the Oracle9*i*AS Release 2 (1.0.2.2.x) Oracle home.
2. Locate the host definitions in the Release 2 (1.0.2.2.x) `webcache.xml` file.
3. In the **Web Cache Manager** page frame, locate the **Origin Servers, Sites and Load Balancing** section in the frame and click the **Origin Servers** link.

   The content area is populated with the current application server hosts.
4. Click **Add**.
5. The **Add Application Web Server** window opens.
6. Enter the values and click **Submit**.

   The **Web Cache Manager** page reappears.
7. Click **Apply Changes**.

   The page refreshes with a message to restart Web Cache.
8. Click **Restart**.
9. To add additional hosts, click the **Origin Servers** link and repeat Steps 4 through 9.

4.3.6 Upgrading Site-to-Server Mappings

Follow these steps to create site-to-server mappings for forwarding of requests to application server hosts.

1. Open the `webcache.xml` file in the Oracle9*i*AS Release 2 (1.0.2.2.x) Oracle home.
2. Locate the site-to-server mappings in the Release 2 (1.0.2.2.x) `webcache.xml` file.
3. In the **Web Cache Manager** page frame, locate the **Origin Servers, Sites and Load Balancing** section in the frame and click the **Site-to-Server Mapping** link.

   The content area is populated with the current site and origin server pairs.
4. Select the radio button for the site and server you want and click **Edit Selected**, or click **Add** to add a session.

   The **Edit/Add Site-to-Server Mapping** window opens.
5. Edit or add the values for the session and click **Submit**.

   The **Web Cache Manager** page reappears.
6. Click **Apply Changes**.

   The page refreshes with a message to restart Web Cache.
7. Click **Restart**.

   A dialog box opens with a message that the cache was stopped and started.
8. Click **OK**.
9. To upgrade additional mappings, click the **Site-toServer Mapping** link and repeat Steps 4 through 8.

4.3.7 Upgrading Resource Limits

Follow these steps to upgrade cache sizes and connection limits.

1. Open the `webcache.xml` file in the Oracle9*i*AS Release 2 (1.0.2.2.x) Oracle home.
2. Locate the cache size and connection limit values in the Release 2 (1.0.2.2.x) `webcache.xml` file.
3. In the **Web Cache Manager** page frame, locate the **Properties** section in the frame and click the **Resource Limits** link.

   The content area is populated with the current caches.
4. Select the radio button for the cache you want and click **Edit Selected**.

   The **Edit Resource Limits** window opens.
5. Edit or values for the cache and click **Submit**.

   The **Web Cache Manager** page reappears.
6. Click **Apply Changes**.

   The page refreshes with a message to restart Web Cache.
7. Click **Restart**.

   A dialog box opens with a message that the cache was stopped and started.
8. Click **OK**.
9. If applicable, change the Maximum Cached Object Size by following these steps:
   a. In the **Web Cache Manager** page frame, locate the **Properties** section in the frame and click the **Resource Limits** link.
   b. Click **Edit**.

      The Edit Maximum Cached Object Size window opens.
   c. Do one of the following:

      Click the **Don't set maximum cache size** radio button

      Click the **Set maximum cached object size:** radio button and enter a value for the size.
   d. Click **Submit**.

      The **Web Cache Manager** page reappears.
   e. Click **Apply Changes**.

      The page refreshes with a message to restart Web Cache.
   f. Click **Restart**.

      A dialog box opens with a message that the cache was stopped and started.
   g. Click **OK**.

4.3.8 Upgrading Trusted Subnets

Follow these steps to upgrade trusted subnet selections.

1. Open the `webcache.xml` file in the Oracle9*i*AS Release 2 (1.0.2.2.x) Oracle home.
2. Locate the trusted subnet definitions in the Release 2 (1.0.2.2.x) `webcache.xml` file.
3. In the **Web Cache Manager** page frame, locate the **Properties** section in the frame and click the **Security** link.

   The content area is populated with security settings.
4. Click **Change Trusted Subnets**.

   The **Change Trusted Subnets** window opens.
5. Click the radio button to select the subnets you want (entering IP addresses if applicable) and click **Submit**.

   The **Web Cache Manager** page reappears.
6. Click **Apply Changes**.

   The page refreshes with a message to restart Web Cache.
7. Click **Restart**.

   A dialog box opens with a message that the cache was stopped and started.
8. Click **OK**.

4.3.9 Upgrading Error Pages

Follow these steps to upgrade error page selections.

1. Open the `webcache.xml` file in the Oracle9*i*AS Release 2 (1.0.2.2.x) Oracle home.
2. Locate the error page definitions in the Release 2 (1.0.2.2.x) `webcache.xml` file.
3. In the **Web Cache Manager** page frame, locate the **Origin Servers, Sites and Load Balancing** section in the frame and click the **Error Pages** link.

   The content area is populated with error page information.
4. Click the radio button to select the site name or default pages and click **Edit Selected**.

   The **Edit Error Pages** window opens.
5. Enter file names for the error pages and click **Submit**.

   The **Web Cache Manager** page reappears.
6. Click **Apply Changes**.

   The page refreshes with a message to restart Web Cache.
7. Click **Restart**.

   A dialog box opens with a message that the cache was stopped and started.
8. Click **OK**.

4.3.10 Running OracleAS Web Cache with Root Privilege

On UNIX systems, OracleAS Web Cache requires root privilege when:

- It is configured to run on ports lower than 1024.
- It is configured to run as a user other than the user that installed it.
- It is configured to support more than 1024 connections (including connections to origin servers and OracleAS Web Cache cluster members.

To grant OracleAS Web Cache root privilege, execute the following command:

```
<destination_MT_OH>/webcache/bin/webcache_setuser.sh
```

See the *Oracle Application Server Web Cache Administrator's Guide* in the Oracle Application Server documentation Library for more information on the `setuser.sh` script.

4.4 Validating the OracleAS Web Cache Upgrade

Perform the following steps to ensure that OracleAS Web Cache is functioning correctly after the upgrade:

1. Try to access the normal listening ports.
2. Send HTTP requests to OracleAS Web Cache and check the responses.
3. Use the Administration user interface (which uses the Administration port) to display statistics.

5

Upgrading to OracleAS Wireless

This chapter contains step-by-step instructions for upgrading Oracle9*i*AS Wireless. It contains the following sections:

5.1 OracleAS Wireless Upgrade Items

An upgrade item is an aspect of configuration that is acted upon during upgrade: a file, executable, or a setting that you must add, change, or delete/replace in the 10g (9.0.4) installation. OracleAS Wireless upgrade items are described in Table 5–1.

Table 5–1 OracleAS Wireless Upgrade Items

Upgrade Item	Location in Release 1 (1.0.2.2.x)	Location in 10*g* (9.0.4)	Notes
Database schema			The database schema has been modified and enhanced significantly in Oracle Application Server 10*g* (9.0.4). The wireless schema must be upgraded.
Model objects			Model objects are modified because new model objects have been introduced and schema changes have been made. Model objects in the following groups are affected: ▪ group ▪ role ▪ user ▪ service list ▪ transformer ▪ logical device ▪ adapters
User Agent property files	Properties file in middle tier	Database	Previously, the HTTP user agent header to logical device mapping was stored as a plain properties file. In this release, the mapping is stored in the database. This information must be moved from the Java properties file to the database.
Transformers			Because persistent representation between the transformer and logical device has been modified since Oracle9*i*AS Release 1 (1.0.2.2), a separate Java program is now used to upgrade to the new table schema.
Site and Node configuration property	Properties files in middle tier	Database	
User information in the PanamaUser Table	Wireless Repository	`orclUserV2` user attributes in Oracle Internet Directory	

5.2 Components Related to OracleAS Wireless

Oracle Application Server Wireless can be integrated with Oracle Internet Directory, OracleAS Web Cache, and OracleAS Portal. In Oracle Application Server 10*g* (9.0.4), user information is stored centrally in OID. The SSO (Single Sign-On) server uses an OID repository to authenticate users.

5.3 Preliminary OracleAS Wireless Upgrade Tasks

Before you begin upgrading, ensure that the tasks below have been performed. The necessity of installing the Oracle Application Server 10*g* (9.0.4) Infrastructure and Portal and Wireless middle tier is discussed in Section 1.1, "The Oracle9iAS Release 1 (1.0.2.2.x) to Oracle Application Server 10g (9.0.4) Upgrade Process" on page 1-1, but it is repeated here with some additional steps specific to the OracleAS Wireless component (creation of backups at certain points in the process).

1. Back up the Oracle9*i*AS Release 1 (1.0.2.2.x) database and middle tier.
2. Go to the patch download page by accessing this link:

   `http://updates.oracle.com/ARULink/PatchSearch/process_form?aru=5993750`
3. Click **Download**.

   The **connect to updates.oracle.com** dialog box appears.
4. Enter your MetaLink user name and password.

   The **Patch Download** page appears.
5. Click **Download** and save the `p3509336_9040_GENERIC.zip` file.
6. Unzip the file by issuing this command:

   `unzip p3509336_9040_GENERIC.zip`

   The directory `ias1022_10gupgrade` is created.
7. Copy the `prepare_exp.sql` file from the `ias1022_10gupgrade/sql/` directory to the `<source_MT_OH>/panama/sql` directory.
8. Drop the spatial indexes in the Oracle9*i*AS Wireless schema in the Oracle9*i*AS Release 1 (1.0.2.2.x) database by issuing the following command:

   `sqlplus <iAS 1022 user>/<password>@<iAS 1022 db SID> @prepare_exp.sql`

   where:

   - *`iAS 1022 user`* is the Oracle9*i*AS Wireless schema user
   - *`password`* is the Oracle9*i*AS Wireless schema user's password
   - *`iAS 1022 db SID`* is the SID

   for example:

   ```
   sqlplus myself/welcome@iasdb @prepare_exp.sql
   ```
9. Create a dump of the Oracle9*i*AS Release 1 (1.0.2.2.x) Wireless schema by issuing the following command:

   `exp system/<password> owner=<iAS 1022 wireless user> consistent=y file=<exported>.dmp log=<exported>.log`

   where:

   - *`iAS 1022 user`* is the iAS database user
   - *`password`* is the iAS database user's password
   - *`exported`* is the descriptive name given to the exported schema and the associated log file

   for example:

   ```
   exp system/welcome owner=myself consistent=y file=my_schema.dmp log=my_schema.log
   ```
10. Install the Oracle Application Server 10*g* (9.0.4) Infrastructure.
11. Install the Oracle Application Server 10*g* (9.0.4) Portal and Wireless middle tier. *Do not configure the Wireless component during installation*.

Note: If you install the Oracle Application Server 10*g* (9.0.4) middle tier on a computer other than that on which the Oracle9*i*AS Release 1 (1.0.2.2.x) middle tier is installed, you need to upload the Oracle9*i*AS Release 1 (1.0.2.2.x) Panama folder to the 10*g* (9.0.4) middle tier computer after installing 10*g* (9.0.4), and unzip the file.

If the Oracle Application Server 10*g* (9.0.4) middle tier is on a different computer from the Oracle9*i*AS Release 1 (1.0.2.2.x) middle tier, set the environment variables as follows in the 10*g* (9.0.4) Oracle home as follows:

- ORACLE_HOME to the Oracle Application Server 10*g* (9.0.4) middle tier Oracle home
- ORACLE_SID to the Wireless schema
- (UNIX) PATH to $ORACLE_HOME/bin:$PATH

  (Windows) PATH to %ORACLE_HOME\bin:%PATH

12. Back up the Oracle Application Server 10*g* (9.0.4) database and middle tier.
13. Create an inventory of all of the alerts and customizations made to Oracle9*i*AS Wireless in Oracle9*i*AS Release 1 (1.0.2.2.x). The alerts from Oracle9*i*AS Release 1 (1.0.2.2.x) are not supported in 10*g* (9.0.4), so they must be re-created there as part of the upgrade process.

5.4 OracleAS Wireless Upgrade Tasks

This section provides instructions for upgrading each item identified in Section 5.1, "OracleAS Wireless Upgrade Items". It is divided into the following subsections:

5.4.1 Obtaining the 10*g* (9.0.4) Instance OracleAS Wireless Schema Password

You need the Wireless database schema password to perform the upgrade. Follow these steps to obtain the password from Oracle Internet Directory:

1. Ensure that Oracle Internet Directory and Oracle Application Server Single Sign-On are running.
2. Execute the `wireless-pwd.sh`. script (available in the `ias1022_10gupgrade/bin` directory) by issuing this command:

   (UNIX) **`./wireless-pwd.sh <OID host> <OID port> <OID password>`**

   (Windows) **`wireless-pwd.bat <OID host> <OID port> <OID password>`**

   where:

   - *`OID host`* is the hostname of the OID server
   - *`OID port`* is the port of the OID server

- *OID password* is the password of the OID administrative user (orcladmin)

for example:

```
./wireless-pwd.sh oidhost.mycompany.com 3060 welcome1
```

The script executes and generates output.

3. Note the value of `orclpasswordattribute`, which is the Wireless schema password you will use when prompted during the upgrade procedure.

5.4.2 Preparing the Middle Tier Oracle Home Environment

To prepare the environment, follow these steps in the 10*g* (9.0.4) middle tier Oracle home:

1. Stop OPMN and all of the components it manages by issuing this command from *<destination_MT_OH>*`/opmn/bin`:

   `opmnctl stopall`

2. Ensure that Oracle Enterprise Manager is running. To start it, issue this command from *<destination_MT_OH>*`/bin`:

   `emctl start em`

5.4.3 Dropping and Creating OracleAS Wireless Schema

On the computer on which the Infrastructure database is installed, follow these steps to drop and create the OracleAS Wireless schema:

1. Issue the commands below to drop the schema:

   `sqlplus <DBA user>/<DBA password>@<service name>`

   `drop user wireless cascade;`

   where:

   - *DBA user* is the DBA user name
   - *DBA password* is the DBA user's password
   - *service name* is the database name

   for example:

   ```
   sqlplus system/welcome1@asdb

   drop user wireless cascade;
   ```

2. Create new wireless schema using the `create_aq_user.sql` script, available in *<destination_MT_OH>*`/wireless/repository/sql/`. Issue this command:

   `sqlplus <DBA user>/<DBA password>@<service name> @create_aq_user.sql WIRELESS <wireless password>;`

   where:

   - *DBA user* is the DBA user name
   - *DBA password* is the DBA user's password
   - *service name* is the database name
   - *wireless password* is the password you obtained in Section 5.4.1, "Obtaining the 10g (9.0.4) Instance OracleAS Wireless Schema Password" on page 5-4

for example:

```
sqlplus system/welcome1@asdb @create_aq_user.sql WIRELESS
UmS7rubt;
```

3. On the computer on which the 10*g* (9.0.4) Infrastructure is installed, set the ORACLE_HOME environment variable to *<destination_Infra_OH>* and the PATH environment variable to $ORACLE_HOME/bin.
4. If the source and destination Oracle homes are on different computers, use FTP to transfer the `.dmp` file generated in Section 5.3, "Preliminary OracleAS Wireless Upgrade Tasks" on page 5-2 to the current computer.
5. Determine whether the Wireless schema's tablespace in Oracle9*i*AS Release 1 (1.0.2.2.x) exists in the Oracle Application Server 10*g* (9.0.4) Infrastructure database by following these steps:
   a. Connect to the Oracle9*i*AS Release 1 (1.0.2.2.x) database as the SYS user.
   b. Execute the query:

      `select username, default_tablespace from dba_users where username 'WIRELESS';`

      The query result resembles the following:

      ```
      USERNAME                       DEFAULT_TABLESPACE
      ------------------------------ ------------------------
      WIRELESS                       USERS
      ```

   c. Connect to the Oracle Application Server 10*g* (9.0.4) database as the SYS user.
   d. Execute the query:

      `select tablespace_name from dba_tablespaces where tablespace_name '<tablespace name>';`

      If the tablespace exists, the query returns the tablespace name (in the examples, the name is `USERS`). If it does not, then create the tablespace by following Step e. If it exists, continue with Step f.
   e. Issue this command:

      `create tablespace <tablespace name> datafile '<datafile name>' SIZE <free space>M;`

      where:

      `tablespace name` is the name of the tablespace (the name returned by the query on the Oracle9*i*AS Release 1 (1.0.2.2.x) database)

      `datafile name` is the name of the datafile (the same name as in the Oracle9*i*AS Release 1 (1.0.2.2.x) database)

      `free space` is the amount of memory in megabytes to allocate to the tablespace

      for example:

      ```
      create tablespace USERS datafile
      '/private1/mig/as10ginfra/oradata/asdb/user.dbf' SIZE
      200M;
      ```

   f. Ensure that the memory of the existing tablespace is at least 200 MB by executing this query:

```
select (SUM (BYTES)/(1024 * 1024)) from dba_free_space
where TABLESPACE_NAME='<tablespace name>';
```

The query returns a value for MEMORY. If it is less than 200, continue with Step g to increase it. If it is 200 or more, continue with Step 6.

g. Obtain the file name of the tablespace with this query:

```
select TSNAME, FNAME from sysfiles where
TSNAME='<tablespace name>'
```

h. Allocate additional memory to the tablespace by issuing this command:

```
alter tablespace <tablespace name> datafile '<datafile
name>' RESIZE <free space>M;
```

where:

tablespace name is the name of the tablespace (the same name as in Oracle9*i*AS Release 1 (1.0.2.2.x))

datafile name is is the name of the datafile specified by the FNAME value in the query result produced in step g.

free space is the amount of memory in megabytes to allocate to the tablespace; in this case it needs to be 200 or greater

for example:

```
alter tablespace USERS datafile
'/private1/mig/as10ginfra/oradata/asdb/users01.dbf' RESIZE
200M;
```

6. Import the Wireless data from the .dmp file into the Metadata Repository, using the import tool imp. To do this, issue the following command:

```
imp system/<password> fromuser=<1022 wireless schema user>
touser=<904 wireless schema user>  file=exported.dmp commit=y
ignore=n log=imported.log
```

where:

- *password* is the SYS user password
- *1022 wireless schema user* is the owner of the Wireless schema in Oracle9*i*AS Release 1 (1.0.2.2.x)
- *904 wireless schema user* is the owner of the Wireless schema in Oracle Application Server 10*g* (9.0.4)

for example:

```
imp system/welcome1 fromuser=scott touser=fred
file=exported.dmp commit=y ignore=n log=imported.log
```

You may safely ignore the warning message "Import terminated successfully with warnings."

7. Ensure that the ORACLE_HOME and PATH environment variables and the ORACLE_SID are set to point the 10*g* (9.0.4) middle tier.

8. Configure the database connection in the <*destination_MT_OH*>/wireless/server/classes/oracle/panama/core/admin/system.properties file by following these steps:

a. Set standalone=true.

b. Modify `db.connect.string` to use the THIN driver and comment out all remaining `db.connect.string` settings.

The `db.connect.string` has the following format:

`wireless/<wireless password>@hostname:port:sid`

for example:

`wireless/UmS7rubt@usunnad27.us.oracle.com:1521:asdb`

c. Set `db.driver=THIN`.

9. If the Oracle9*i*AS Release 1 (1.0.2.2.x) and the Oracle Application Server 10*g* (9.0.4) middle tiers are on different computers:

In the `<source_MT_OH>/panama/server/classes/oracle/panama/spatial/spatial.properties` file, change all absolute paths to the path of the current Oracle9*i*AS Release 1 (1.0.2.2.x) Oracle home.

For example:

`file.providers.config.xml.geocoding = /private1/home/iaswbm/ias1022/panama/server/classes/oracle/panama/spatial/geocoder/Geocoders.xml` (The path resembles this prior to uploading the panama folder to the 10*g* (9.0.4) instance's computer.)

Assume that the new Oracle9*i*AS Release 1 (1.0.2.2.x) instance (the panama folder of Oracle9*i*AS Release 1 (1.0.2.2.x)) in the 10*g* (9.0.4) computer is `/private1/ias1022`. Then the path above would be as follows:

`file.providers.config.xml.geocoding = /private1/ias1022/panama/server/classes/oracle/panama/spatial/geocoder/Geocoders.xml`

5.4.4 Executing the OracleAS Wireless Upgrade Script

A script, `ias1022-902.sh`, is provided in the `ias1022_10gupgrade/bin` directory to perform the upgrade. Follow these steps in the 10*g* (9.0.4) Oracle home to execute the script:

1. Issue the following command:

(UNIX) **`./ias1022-902.sh <source_MT_OH> <destination_MT_OH> <script path> <connect str> <admin password>`**

(Windows) **`ias1022-902.bat <source_MT_OH> <destination_MT_OH> <script path> <connect str> <admin password>`**

where:

- *`source_MT_OH`* is the path to the Oracle home of the Oracle9*i*AS Release 1 (1.0.2.2.x) installation
- *`destination_MT_OH`* is the path to the Oracle home of the 10g installation
- *`script path`* is the path to the `ias1022_10gupgrade` directory
- *`connect str`* is the connect string for the wireless schema (to be specified as `user/password@sid`)
- *`admin passwd`* is the password of the admin user specified during the Oracle9*i*AS Release 1 (1.0.2.2.x) installation

for example:

```
./ias1022-902.sh /private1/mig/ias1022 /private1/mig/as10gmid
/private1/mig/ias1022_10gupgrade wireless/UmS7rubt@asdb
manager
```

Note: *source_MT_OH* can have two different values, depending on the configuration upgraded:

- If the source and destination middle tiers are on different computers, and the `panama` directory of the Oracle9*i*AS Release 1 (1.0.2.2.x) was uploaded to the 10*g* (9.0.4) computer under `/private1`, then *source_MT_OH* is private1.
- If the source and destination middle tiers are on the same computer, then *source_MT_OH* is the path to the Oracle9*i*AS Release 1 (1.0.2.2.x).

---

2. In the `<`*destination_MT_OH*`>/wireless/server/classes/oracle/panama/core/admin/system.properties` file, set `standalone=false`. This causes the upgrade process to obtain database connections from Oracle Internet Directory.
3. Configure the OracleAS Wireless 10*g* (9.0.4) component by following these steps:
   a. Access Oracle Enterprise Manager with the following URL:

      `http://<`*hostname*`>:<`*port*`>`

      A login dialog box appears for the Oracle Enterprise Manager administrative user. The default name of this user is `ias_admin`, and the default port is 1810.

   b. Enter the administrative user password.

      The **Application Servers** page appears.

   c. In the **Standalone Instances** section, click the middle tier instance.

   d. In the **System Components** section, click **Configure Component**.

      The **Select Component** page appears.

   e. Select **Wireless** from the drop-down list.

      A password dialog appears.

   f. Enter the orcladmin password and then click **Finish**.

      The configuration process begins, and continues for approximately one hour.
4. Register OracleAS Wireless with Oracle Application Server Single Sign-On using the `ssoreg.sh` script on UNIX or the ssoreg.bat script on Windows. The script is available in the `ias1022_10gupgrade/bin` directory. Issue this command:

   (UNIX) **`./ssoreg.sh`**

   (Windows) **`ssoreg.bat`**
5. Migrate the users using the `ptgUsrMigrate.sh` script on UNIX or the `ptgUsrMigrate.bat` script on Windows. The script is available in the `ias1022_10gupgrade/bin` directory. Issue this command:

   (UNIX) **`./ ptgUsrMigrate.sh <ldap host> <ldap port> <ldap dn> <ldap password> <connect str> <script path>`**

   (Windows)

ptgUsrMigrate.bat <*ldap host*> <*ldap port*> <*ldap dn*> <*ldap password*> <*connect str*> <*script path*>

where:

- *ldap host* is the location of the OID server
- *ldap port* is the port of the OID server
- *ldap dn* is the DN of the OID admin user like cn=<admin_user>
- *ldap password* is the password of the OID admin user
- *connect str* is the connect string to the wireless schema, in the format user/password@hostname:port:sid
- *script path* is the path to the ias1022_10gupgrade directory

for example:

```
./ptgUsrMigrate.sh usunnad27.us.oracle.com 3060 cn=orcladmin
welcome1 wireless/UmS7rubt@usunnad27.us.oracle.com:1521:asdb
/private1/mig/ias1022_10gupgrade
```

6. Execute the populateGuid.sh script on UNIX or the populateGuid.sh script on Windows. The script is available in the ias1022_10gupgrade/bin directory. Issue this command:

   (UNIX) **./populateGuid.sh <*connect str*>**

   (Windows) **populateGuid.bat <*connect str*>**

   where *connect str* is the connect string for the Wireless schema

   for example:

   ```
   ./populateGuid.sh wireless/UmS7rubt@asdb
   ```

7. Start the opmn process. Issue this command:

   <*destination_MT_OH*>/opmn/bin/opmnctl startall

5.5 Validating the OracleAS Wireless Upgrade

After the upgrade is complete, follow these steps to validate its success:

1. Log in to OracleAS Wireless as the user that managed Oracle9*i*AS Wireless, and verify that you can access all of the services you could access in Oracle9*i*AS Release 1 (1.0.2.2.x).
2. Check the groups and services to verify that they are upgraded. Use the web-based tool to see if the groups and users in Oracle9*i*AS Release 1 (1.0.2.2.x) are upgraded properly.
3. Test the new features of OracleAS Wireless in 10*g* (9.0.4). Refer to the *Oracle Application Server Wireless Administrator's Guide* in the 10*g* (9.0.4) documentation library.

6

Upgrading mod_plsql

This chapter contains step-by-step instructions for upgrading the mod_plsql configuration: Database Access Descriptors (DADs) and cache settings. It contains the following sections:

6.1 mod_plsql Upgrade Items

An upgrade item is an aspect of configuration that is acted upon during upgrade: a file, executable, or a setting that you must add, change, or delete/replace in the 10g (9.0.4) installation. `mod_plsql` upgrade items are described in Table 6–1.

Table 6–1 mod_plsql Upgrade Items

Upgrade Item	Location in Release 1 (1.0.2.2.x) Oracle home	Location in 10*g* (9.0.4) Oracle home	Notes
`wdbsvr.app`	`Apache/modplsql/cfg/`		
`cache.cfg`	`Apache/modplsql/cfg/`		
`dads.conf`		`Apache/modplsql/conf/`	

6.2 mod_plsql Upgrade Tasks

A command line script, (`modPlsqlMigration.csh` on UNIX and `modPlsqlMigration.cmd` on Windows) moves the definitions to the new format. The `modPlqlMigration` script replaces the `dadMigration` script and allows the migration of cache settings as well as the DADs. The scripts are located in the OracleAS middle-tier 10g (9.0.4) Oracle home, in the `bin` directory.

The scripts operate on the <*source_MT_OH*>`/Apache/modplsql/cfg/wdbsvr.app` and `cache.cfg` source files. The scripts update the <*destination_MT_OH*>`/Apache/modplsql/conf/dads.conf` and `cache.conf` files.

The command syntax for executing the script is:

```
<migration script> -em <em home> [-s <source_MT_OH>] [-t
<destination_MT_OH>] [[-c [-dir <cache directory>]] | -d]
```

Table 6–2 describes the variables in the command.

Table 6–2 mod_plsql Upgrade Script Variables

Variable	Definition
<*migration script*>	The migration script being run (`modPlsqlMigration.csh` on UNIX and `modPlsqlMigration.cmd` on Windows).
`-em` <*em home*>	When migrating from a Oracle9*i*AS Release 1 (1.0.2.2.x) middle tier to an Oracle Application Server 10*g* (9.0.4) middle tier, the Oracle Enterprise Manager home should always be specified as theOracle Application Server 10*g* (9.0.4) middle tier Oracle home. This option is necessary, because the scripts support upgrades to other versions as well. There is no default value for this option.
`-s` <*source_MT_OH*>	The Oracle9*i*AS Release 1 (1.0.2.2.x) middle tier Oracle home that is the source of the migration. If not specified, then the current directory is used.
`-t` <*destination_MT_OH*>	The Oracle Application Server 10*g* (9.0.4) middle tier Oracle home that is the target of the migration. If not specified, the value of the ORACLE_HOME environment variable is used.
`-c`	Perform only a cache upgrade. Do not use this option at the same time as the `-d` option. By default, both upgrades (cache and DAD) are performed.
`-dir` <*cache directory*>	The value for the cache directory property. Do not use this option at the same time as the `-d` option. If no value is specified, the `Apache/modplsql/cache` directory in the destination OracleAS middle-tier 10g (9.0.4) Oracle home will be used.
`-d`	Perform only a DAD migration. Do not use this option at the same time as the `-c` option. By default, both upgrades (cache and DAD) are performed.

Examples:

On UNIX, the command below migrates DADs and cache settings using the current directory as the source Oracle9*i*AS Release 1 (1.0.2.2.x) middle tier Oracle home, the value of the ORACLE_HOME environment variable as the destination Oracle Application Server 10*g* (9.0.4) middle tier Oracle home, and the default cache directory:

```
$ORACLE_HOME/bin/modPlsqlMigration.csh -em $ORACLE_HOME
```

On Windows, the command below migrates only the DADs and specifies the source and destination Oracle homes. No defaults are used in this example:

```
%ORACLE_HOME%\bin\modPlsqlMigration.cmd -em c:\oracle\ora904mid
-s c:\oracle\ora1022mid -t c:\oracle\ora904mid -d
```

On UNIX, the command below migrates only the cache settings. It specifies the current directory as the source Oracle9*i*AS Release 1 (1.0.2.2.x) middle tier Oracle home, the value of the ORACLE_HOME environment variable as the destination Oracle Application Server 10*g* (9.0.4) middle tier Oracle home, and the default cache directory:

```
$ORACLE_HOME/bin/modPlsqlMigration.csh -em /oracle/ora904mid -c
```

6.2.1 Upgrading from One Computer to Another

If necessary, you can upgrade the DAD and cache information from one computer to another, by following the steps below:

1. Create the following directory structure on the destination computer (for example, by copying files from your Oracle9iAS Release 1 install). The script looks for these files in the Oracle9iAS Release 1 Oracle home:

   `<ORACLE_HOME>/Apache/modplsql/cfg/wdbsvr.app`

   `<ORACLE_HOME>/Apache/modplsql/cfg/cache.cfg`

2. Execute the upgrade script, referencing this pseudo Oracle home using the `-s` option to specify the directory, or by making the pseudo Oracle home the current directory.

7

Upgrading to OracleAS Discoverer

This chapter contains step-by-step instructions for upgrading Oracle9*i*AS Discoverer to OracleAS Discoverer. The instructions were excerpted from the *Oracle Application Server Discoverer Configuration Guide* and the *Oracle Discoverer Administrator Administration Guide*. This chapter contains the following sections:

7.1 OracleAS Discoverer Upgrade Items

An upgrade item is an aspect of configuration that is acted upon during upgrade: a file, executable, or a setting that you must add, change, or delete/replace in the 10g (9.0.4) installation. OracleAS Discoverer upgrade items are described in Table 7–1.

Table 7–1 OracleAS Discoverer Upgrade Items

Upgrade Item	Location in Release 1 (1.0.2.2.x) Oracle home	Location in 10*g* (9.0.4) Oracle home	Notes
`pref.txt` Default user preferences	`discwb4/util/pref.txt`	`discoverer/util/pref.txt`	In 10*g* (9.0.4), there are additions to the `pref.txt` file
`.reg_key.dc` User-level preferences	`discwb4` directory (UNIX) Registry (Windows)	`.reg_key.dc` file (the location is specified in the `DC9_REG` environment variable) (UNIX) `discoverer/.reg_key.dc` (Windows)	In 10*g* (9.0.4), there are additions to the `.reg_key.dc` file.
End User Layer (EUL)	Not applicable; database schema	Not applicable; database schema	To upgrade the Oracle Application Server Discoverer End User Layer schema, you use Oracle Discoverer Administrator, shipped with the Oracle Developer Suite 10*g* (9.0.4). For more information, see: *Oracle Discoverer Administrator Administration Guide* in the Oracle Developer Suite documentation library.

Table 7–1 (Cont.) OracleAS Discoverer Upgrade Items

Upgrade Item	Location in Release 1 (1.0.2.2.x) Oracle home	Location in 10*g* (9.0.4) Oracle home	Notes

URL references	`http://hostname/Discwb4/html/ english/ms_ie/start_ie.htm`
	OR
	`http://hostname/Discwb4/html/english/netscape/start_nn.htm`
	AND
	`http://hostname/Discoverer4i/Viewer`
		`http://hostname/discoverer/plus`
		`http://hostname/discoverer/viewer`
			There are multiple changes in URL names.

Session timeout configuration	`Apache/JServ/servlets/disco4iviewer.properties`	`j2ee/OC4J_BI_Forms/applications/discoverer/web/WEB-INF/web.xml`	The file in the Release 1 (1.0.2.2.x) installation is a text file, plugged into Oracle HTTP Server. The file in 10*g* (9.0.4) is an xml file that contains many more configuration entities, and is plugged into the OC4J stack.

Discoverer Viewer customizations	`Apache\Apache\htdocs\disco4iv\html\disco4iv.xml`
	`Apache\Apache\htdocs\disco4iv\html\ viewer_config.xml`
	`Apache\Apache\htdocs\disco4iv\html\errors.xsl`
	`Apache\Apache\htdocs\disco4iv\html\functions.xsl`
	`Apache\Apache\htdocs\disco4iv\html\gui_components.xsl`
	`Apache\Apache\htdocs\disco4iv\html\render_table.xsl`
	`Apache\Apache\htdocs\disco4iv\html\scripts.xsl`
	`Apache\Apache\htdocs\disco4iv\html\style.xsl`
		`j2ee/OC4J_BI_Forms/applications/discoverer/web/WEB-INF/configuration.xml`
		`j2ee/OC4J_BI_Forms/applications/discoverer/web/common/xsl/ui/ui_config.xml`
		`j2ee/OC4J_BI_Forms/applications/discoverer/web/common/xsl/*`
		`j2ee/OC4J_BI_Forms/applications/discoverer/web/viewer_files/xsl/*`
			The file in the Release 1 (1.0.2.2.x) installation is a text file, plugged into Oracle HTTP Server. The file in 10*g* (9.0.4) is an xml file that contains many more configuration entities, and is plugged into the OC4J stack.

7.2 Components Related to OracleAS Discoverer

In 10*g* (9.0.4), OracleAS Discoverer is integrated with OPMN, Oracle Enterprise Manager, and the OC4J Business Intelligence and Forms stack.

7.3 OracleAS Discoverer Upgrade Tasks

This section explains how to upgrade Discoverer from version 4.1 (the version shipped in Oracle9*i*AS Release 1 (1.0.2.2.x)) to 10*g* (9.0.4).

Before you begin the upgrade process, ensure that Oracle Application Server 10*g* (9.0.4) was installed successfully. All of the instructions in this section assume that Oracle9*i*AS Oracle9*i*AS Release 1 (1.0.2.2.x) is installed in `<source_MT_OH>` and Oracle Application Server 10*g* (9.0.4) is installed in `<destination_MT_OH>`.

Upgrade tasks are described in the following subsections:

7.3.1 Upgrading Preferences

This section explains how to upgrade Discoverer preferences.

Hint: If Discoverer scripts do not execute on UNIX, run the script file `discwb.sh` to set the Discoverer environment.

7.3.1.1 Upgrading Default User Preferences

The `pref.txt` file specifies the default user preferences. To upgrade default user preferences, follow the steps below:

1. Compare the Oracle9*i*AS Release 1 (1.0.2.2.x) `pref.txt` file with the 10*g* (9.0.4) `pref.txt` file, as follows:

   Compare `<source_MT_OH>/discwb4/util/pref.txt` with `<destination_MT_OH>/discoverer/util/pref.txt`.

2. If changes have been made to the `pref.txt` file in the `<source_MT_OH>` directory, edit the `pref.txt` file in the `<destination_MT_OH>` directory and make the same changes.

3. Apply the default preferences.

7.3.1.2 Upgrading User-Level Preferences on UNIX

If you are upgrading from one Oracle home to another on the same computer using UNIX, follow the instructions below to upgrade user level preferences:

1. Copy the `.reg_key.dc` file from the location specified by the `DC_REG` environment variable to the location specified in the `discwb.sh` file in `<destination_MT_OH>/<installation>/discwb4`.

2. Issue this command:

   `<destination_MT_OH>/discoverer/util/migrateprefs.sh -from 102`

If you are upgrading from one UNIX computer to another UNIX computer, follow the instructions below to upgrade user level preferences:

1. Copy the `.reg_key.dc` file from the location specified by the `DC_REG` environment variable on the computer on which Oracle9*i*AS Release 1 (1.0.2.2.x) is installed to the location specified by the `DC9_REG` environment variable on the

computer on which Oracle Application Server 10*g* (9.0.4) is installed.

2. On the computer on which Oracle Application Server 10*g* (9.0.4) is installed, issue this command:

   <*destination_MT_OH*>/discoverer/util/migrateprefs.sh -from 102

7.3.1.3 Upgrading User-Level Preferences on Windows

If you are upgrading from one Oracle home to another on the same Windows computer, follow the instructions below to upgrade user level preferences:

1. Issue this command:

   <*destination_MT_OH*>/discoverer/util/migrateprefs.bat -from 102

   User-level preferences are copied from the Windows registry to the `reg_key.dc` file.

If you are upgrading from one Windows computer to another Windows computer, follow the instructions below to upgrade user-level preferences from Discoverer Release 4.1 to Discoverer Version 9.0.4:

1. On the machine on which Oracle9*i*AS Release 1 (1.0.2.2.x) is installed, from the **Start** menu, select **Run**.

   The **Run** dialog box appears.

2. Type **regedit** and press **Enter**.

   The **Registry Editor** window opens.

3. Open the `HKEY_LOCAL_MACHINE\Software\Oracle\WebDisco4` registry key.

4. Choose **Registry ->Export Registry File** to export the registry key to a file.

5. Specify a name for the registry export file (e.g. `disco41prefs.reg`).

6. Copy the registry export file from the computer on which Oracle9*i*AS Release 1 (1.0.2.2.x) is installed to the computer on which Oracle Application Server 10*g* (9.0.4) is installed.

7. On the machine on which Oracle Application Server 10*g* (9.0.4) is installed, from the Windows **Start** menu, select **Run**.

   The **Run** dialog box appears.

8. Type **regedit** and press **Enter**.

   The **Registry Editor** window opens.

9. Choose **Registry-> Import Registry File**.

10. Specify the name of the registry export file to import (e.g. `disco41prefs.reg`).

11. Upgrade the preferences by issuing the following command:

    <*destination_MT_OH*>\discoverer\util\migrateprefs.bat -from 102

7.3.2 Upgrading the End User Layer (EUL)

The version of Discoverer that ships with Oracle Application Server 10*g* (9.0.4) requires an EUL created or upgraded by Discoverer Administrator Version 9.0.4 (which is shipped with Oracle Developer Suite 10g (9.0.4)). If you have an existing Discoverer Release 4.1 EUL, you must upgrade the EUL from Release 4.1 to Version 5.1 using Discoverer Administrator Version 9.0.4.

For more information about upgrading the EUL, see the *Oracle Discoverer Administrator Administration Guide* in the Oracle Developer Suite Documentation Library.

Before you do anything else, you will create a back up of the Release 4.1 EUL as described in Section 7.3.2.1, "Backing Up the Release 4.1 End User Layer" on page 7-6. To use Discoverer Version 9.0.4, Discoverer users must have access to a Discoverer Version 5 End User Layer (EUL). If users were using a Discoverer Release 4.1 EUL, that EUL must be upgraded before it can be accessed with Discoverer Version 9.0.4.

The EUL upgrade process creates new Version 5 EUL tables, and then copies metadata from the Release 4.1 EUL tables into the new tables. Existing users can continue working with Discoverer Release 4.1 during the upgrade process because the upgrade process is non-destructive (i.e. the Release 4.1 EUL tables are not deleted).

The prerequisites for upgrading to 10*g* (9.0.4) are as follows:

- EUL owners must have the following privileges:
  - CREATE PROCEDURE
  - CREATE SESSION
  - CREATE TABLE
  - CREATE VIEW
  - CREATE SEQUENCE
- When upgrading to an Oracle 8.1.7 (or later) database, summaries are implemented as materialized views, so the EUL owners must have the following privileges:
  - CREATE ANY MATERIALIZED VIEW
  - DROP ANY MATERIALIZED VIEW
  - ALTER ANY MATERIALIZED VIEW
  - GLOBAL QUERY REWRITE

> **Note:** To maintain the consistency of data transferred to the upgraded URLs, it is very important that no Discoverer Administrator sessions are opened on the earlier version of the EUL while the upgrade is in progress.

7.3.2.1 Backing Up the Release 4.1 End User Layer

Before you can upgrade a Discoverer Release 4.1 EUL to a Discoverer Version 5 EUL, you must first back up the EUL tables by exporting the EUL owner using the standard database export utility.

How to export the EUL owner will depend on:

- the version of the Oracle client software installed on the computer
- the version of the Oracle database on which the Release 4.1 EUL resides

The EUL upgrade process creates new Version 5 EUL tables, and then copies metadata from the Release 4.1 EUL tables into the new tables. During the upgrade process, existing users can continue working with Discoverer Release 4.1, because the Release 4.1 tables are not deleted during the upgrade process.

Oracle Corporation recommends that the version of the Oracle database and the version of the Oracle database client software installed on the computer are the same. If the versions are not the same (e.g. if the EUL is on an Oracle8i database and Oracle9i client software is installed on your machine), then you might not be able to follow the instructions below. If you are unable to export the EUL, ask your database administrator to export the EUL for you.

To back up a Release 4.1 EUL that resides on an Oracle9*i* database, using a computer on which Oracle Developer Suite is installed, follow the steps below.

1. Open a command shell.
2. Navigate to the directory location to which the End User Layer will be exported.
3. Issue this command:

   exp <*eulowner*>/<*password*>@<*dbname*> file=<*filename*>.dmp owner=<*eulowner*>

   where:

   - *eulowner* is the database user who owns the End User Layer
   - *password* is the password of the user who owns the End User Layer
   - *filename* is the name of the `.dmp` file that the command creates

   For example, to export an End User Layer owned by the database user `hrmgr` to a file named `hreul.dmp`, the command would be:

   ```
   exp hrmgr/hrpswrd@HRDB file=hreul.dmp owner=hrmgr
   ```

   The End User Layer tables, associated synonyms, views, and other definitions are exported to the specified file in the current directory. When the export is complete, the following message appears:

   ```
   Export terminated successfully without warnings.
   ```

4. Close the command shell window.

7.3.2.2 Upgrading the Release 4.1 EUL

After the Release 4.1 EUL is backed up, you can upgrade to Version 5. To upgrade, you open the EUL in the latest version of Discoverer, as described below. The same database user that owned the Release 4.1 EUL will own the Version 5 EUL.

> **Note:** Release 4.1 end users are not affected by upgrade, and can continue using the Release 4.1 EUL. However, any modifications to workbooks that are made in the Release 4.1 EUL after the upgrade process has begun will not be present in the upgraded EUL.
>
> To maintain the consistency of the data transferred to the Version 5 EUL, ensure that no Discoverer Administrator sessions are opened on the Release 4.1 EUl while the upgrade is in progress.

Follow these steps to upgrade a Release 4.1 EUL to Version 5:

1. In Oracle Developer Suite, start the Discoverer Administrator and display the Connect to Oracle Discoverer Administrator dialog. In Windows, the navigation path is:

   Start->Programs->Oracle Developer Suite - <*Oracle home name*>-> Discoverer Administrator

   The **Connect to Oracle Discoverer Administrator** dialog opens.

2. Provide the database user name and password of the owner of the Discoverer Release 4.1 EUL that you want to upgrade and click **OK**.

   The EUL Upgrade dialog appears.

3. Select the **Yes, I have completed a database export** check box to confirm that you have exported the Discoverer Release 4.1 End User Layer.

4. Click **Upgrade** to create a new Version 5 End User Layer using the data from the Release 4.1 End User Layer.

   The EUL Upgrade progress bar displays the status of the upgrade. During the upgrade process, messages are displayed in the **End User Layer Upgrade Log** dialog. For more information about messages that refer to invalid characters in identifiers, see Section 7.4.7, "Identifiers Containing Invalid Characters" on page 7-16.

   If there are Summary folders in the upgraded End User Layer, a message dialog appears, stating that the summary folders are invalid and must be refreshed.

5. If the above message dialog appears, click **OK**.

   The **Load Wizard** appears.

6. Click **Cancel** to close the **Load Wizard**.

   The End User Layer Upgrade is complete. You can now use this End User Layer to connect to the database using Discoverer Administrator 9.0.4.

7.3.2.3 Refreshing the Summary Folders

If there are summary folders in the EUL you have upgraded, you must refresh them.

The definitions of any Release 4.1 summary folders are copied into the new Version 5 EUL and the status of the upgraded summary folders is changed to **Refresh Required**. The database tables or materialized views on which the upgraded summary folders are based are not actually created until the summary folders are refreshed.

When you refresh upgraded summary folders, Discoverer changes the database as follows, depending on the version:

- On Oracle Standard Edition databases, Discoverer creates summary tables
- On Oracle 8.1.7 (or later) Enterprise Edition databases, Discoverer creates materialized views

When the summary tables or materialized views have been created, Discoverer can then use the summary folders to optimize queries.

Follow these steps to refresh the upgraded summary folders:

1. Connect to Discoverer Administrator as the owner of the upgraded EUL.
2. Open the business area containing the summary folders that you want to refresh.
3. Go to the **Summaries** tab of the work area.
4. Select the summary folders to refresh.

   Note: You will have to refresh all of the summary folders before Discoverer can use them. However, depending on the amount of data, you might not want to refresh all the summary folders at the same time.

5. Select **Tools->Refresh Summaries**.

   For any summary folders not based on external summary tables, the **Refresh Summaries** dialog is displayed.

6. Specify refresh options as required.

   Note: You can refresh summary folders immediately or specify a time for the refresh. Depending on the amount of data, you might want to schedule the refresh for an off-peak time.

7. Click **OK** to refresh the summary folders.

   Note: If an upgraded EUL contains summary folders based on external summary tables, you must also refresh those summary folders. During the refresh, Discoverer updates EUL metadata for summary folders based on external summary tables. If you are using an Oracle 8.1.7 database (or later), Discoverer creates a materialized view that references the external summary table.

7.3.2.4 Migrating Users to Discoverer Version 9.0.4

After the EULs are upgraded successfully, you can migrate Discoverer Plus and Discoverer Viewer users to Discoverer Version 9.0.4.

While you are performing the migration across the organization, users can continue to work with Discoverer Release 4.1 (accessing the original EUL), or with Discoverer 10*g* (9.0.4) (accessing the upgraded EUL). However, any changes made using one version of Discoverer Administrator will not be available to users of the other version of Discoverer. The steps for migrating users depends on the Discoverer tools being used.

7.3.2.4.1 Migrating Discoverer Plus and Discoverer Viewer Users To migrate Discoverer Plus and Discoverer Viewer users to Discoverer Version 9.0.4, follow the instructions in:

- Section 7.3.3, "Updating URL References" on page 7-13
- Section 7.3.4, "Configuring Session Time-out" on page 7-13
- Section 7.3.5, "Upgrading Discoverer Viewer Customizations" on page 7-14

7.3.2.4.2 Migrating Discoverer Desktop Users To migrate Discoverer Desktop users:

Install Discoverer Desktop from the Discoverer Desktop CD (shipped with Oracle Developer Suite) to the client computers.

The Oracle Installer automatically installs Discoverer Desktop Version 9.0.4 in a separate directory from Discoverer Release 4.1.

Note: When a user logs in to Discoverer for the first time, using Discoverer Plus, Discoverer Desktop, or Discoverer Viewer, Discoverer first searches for any Version 5 EULs to which the user has access.

- If a valid EUL is found, then the user can connect and start work in Discoverer 10*g* (9.0.4).
- If no valid EUL is found, Discoverer displays a message that the user cannot connect to the database. The user should contact the Discoverer administrator, because the likely cause is that an EUL has not been upgraded. The user can still use Discoverer Release 4.1 against an existing Release 4.1 EUL.

7.3.2.5 Removing the Release 4.1 EULs

When you have upgraded the Release 4.1 EULs and migrated all users to Discoverer 9.0.4, you can remove the Release 4.1 EULs.

Initially, you will probably want to prevent access to the original EULs and only allow users to access the upgraded EULs. When you are confident that all users are successfully accessing the upgraded EULs, you can remove the old EULs.

A SQL script, `eul4del.sql`, is shipped with Discoverer to enable you to remove Release 4.1 EULs and associated tables, including summary tables and materialized views.

Note: Consider the following before you use the eul4del.sql script:

- When you run the `eul4del.sql` script, you are prompted for the password of the SYSTEM user. If you do not know the password of the SYSTEM user, contact your database administrator. Rather than give you the SYSTEM user's password, your database administrator might prefer to run the script for you.
- The `eul4del.sql` script will not complete successfully if there are any summary folders owned by users other than the EUL owner. The script will list any such summary folders and their owners. The summary folder owners must delete their summary folders before you can remove the Release 4.1 EULs.
- If you have access to Discoverer Administration Edition Release 4.1, you can delete End User Layers using the EUL Manager instead of using the `eul4del.sql` script.
- If users have modified the Release 4.1 EUL since you upgraded it, their changes will not be in the Version 9.0.4 EUL. If you think significant changes might have been made, consider taking another backup of the Release 4.1 EUL before removing it (for more information, see Section 7.3.2.1, "Backing Up the Release 4.1 End User Layer" on page 7-6.

Follow these steps to remove a Release 4.1 EUL with the `eul4del.sql` script:

1. Start SQL*Plus. In Windows, the navigation path is:

   Start->ORACLE->Application Development ->SQL*Plus

   The SQL*Plus Log On dialog box appears.

2. Enter the database user name, password and database for the owner of the Discoverer Release 4.1 EUL that you want to remove.

   The command prompt appears.

3. Type the following:

   `@<source_MT_OH>\discoverer\sql\eul4del.sql`

   The deinstallation script removes a specified Release 4.1 EUL and any associated database objects, including summary tables and materialized views. A summary of the deinstallation script appears:

   ```
   Removing summary refresh jobs ...

   Discoverer End User Layer Database Tables (4.x Production)
   ```

```
deinstallation

This script will remove a version 4.x EUL and any associated
database objects.

It will:

1. Ask you to enter the ORACLE SYSTEM password and connect
string.

2. Ask you to enter the name and password of the 4.x EUL
owner.

3. Confirm that you wish to drop the 4.x EUL.

4. Check for database jobs for users other than the 4.x EUL
owner.

5. Confirm whether to drop 4.x tutorial tables (if any).

6. Log in as the 4.x EUL owner and remove any database jobs
for it.

7. Remove all summary database objects for the 4.x EUL.

8. Remove all scheduled workbook database objects for the 4.x
EUL.

9. Remove the 4.x EUL tables.

10. Remove user and public synonyms (if any) for the 4.x EUL
tables.
```

4. Enter the password and database connect string for the SYSTEM user. If you do not know the password of the SYSTEM user, contact your database administrator.
5. Enter the name and password of the owner of the Release 4.1 EUL.

   The following text appears:

   ```
   Preparing to remove EUL 4.x owned by <username> at <today's
   date>

   If you continue, the specified 4.x End User Layer will be
   PERMANENTLY dropped. All End User Layer information and
   workbooks stored in the database will be deleted.

   Any 5.x End User Layer tables will NOT be affected by this
   process.

   THIS PROCESS IS NON-REVERSIBLE.

   Do you wish to continue [N]:
   ```

6. Type **Y** to confirm that you want to drop the Release 4.1 End User Layer.

   If the script detects the Release 4.1 tutorial tables, you are prompted to confirm whether to delete the tables. During the process of dropping the Release 4.1 EUL, the script will check whether there are any summary folders owned by users other than the EUL owner. If there are no summary folders owned by other users, then the Release 4.1 EUL is dropped. Messages resembling the following appear:

   ```
   Removing summary refresh jobs ...

   Dropping internally managed summary data ...

   Removing scheduled workbook jobs ...
   ```

```
Dropping scheduled workbook data ...

Dropping 4.x End User Layer Tables ...

Removing public synonyms (if any) ...

Connected.

Finished removing 4.x End User Layer.
```

If there are summary folders owned by users other than the EUL owner, the script will halt and display messages similar to the following:

```
The following users have managed summaries which must be
dropped before this EUL can be deinstalled:

User: SCOTT, Summary: <Scott's summary>

User: FRED, Summary: <Fred's summary>

Quitting - no changes made.
```

If this message appears, the specified users must log into Discoverer Administration Edition Release 4.1 and remove the specified summary folders (for more information, see Chapter 14 of the *Oracle Discoverer Administrator Administration Guide* in the Oracle Developer Suite documentation library). When these summary folders have been dropped, run the `eul4del.sql` script again to drop the Release 4.1 EUL.

7.3.2.6 Removing the Release 4.1 Products from Client Computers

When users have connected successfully to Discoverer Version 9.0.4 and you are confident that summary folders are working correctly, you can remove Discoverer Release 4.1 products (i.e. Discoverer Administration Edition, Discoverer Desktop Edition) from client computers.

Follow these steps to remove Discoverer Release 4.1 products from client computers:

1. Insert the CD-ROM from which you installed Discoverer Release 4.1 products into your computer's CD-ROM drive.

   The Oracle Universal Installer starts automatically. If it does not start automatically, access the CD-ROM with Windows Explorer and run `setup.exe` from the CD root directory.

2. Use the Oracle Universal Installer deinstall option to remove Discoverer Release 4.1 products.

   Note: Workbooks created in Discoverer Release 4.1 and saved in the <*ORACLE_HOME*>/discvr4 directory will not be deleted.

3. (Optional) To retain workbooks from the <*ORACLE_HOME*>/discvr4 directory for future use:

   a. Move Discoverer Release 4.1 workbooks from the <*ORACLE_HOME*>/discvr4 directory to a new directory for use with Discoverer Version 9.0.4.

   b. Delete the <*ORACLE_HOME*>/discvr4 directory.

7.3.3 Updating URL References

All Discoverer Viewer and Discoverer Plus URL references have changed between Discoverer Release 4.1 and Discoverer Version 9.0.4. These changes include (but are not limited to) links within the Web site and client bookmarks. You must manually replace all occurrences of old URLs with the new URLs. Table 7–2 provides the syntax of the old and new URLs and an example of the required change.

Table 7–2 OracleAS Discoverer URL References

Oracle9*i*AS Release 1 (1.0.2.2.x) URL	10*g* (9.0.4) URL	Example

`http://<hostname>/Discwb4/html/english/ms_ie/start_ie.htm`
or
`http://<hostname>/Discwb4/html/english/netscape/start_nn.htm`
	`http://<hostname>/discoverer/plus`	Change
		`http://<host>:<port>/Discwb4/english/...`
		to
		`http://<host>:<port>/discoverer/plus/...`

`http://hostname/Discoverer4i/Viewer`	`http://hostname/discoverer/viewer`	Change
		`http://<host>:<port>/Discoverer4i/Viewer/...`
		to
		`http://<host>:<port>/discoverer/viewer/...`

7.3.4 Configuring Session Time-out

The session timeout parameter controls the HTTP session timeout (i.e. the number of minutes the Discoverer servlet waits for a browser to make another HTTP or HTTPS request before terminating the user's HTTP session).

In Discoverer Release 4.1, the session timeout parameter is in the `disco4iviewer.properties` file in the following directory:

`<source_MT_OH>/Apache/JServ/servlets`

In Discoverer Version 9.0.4, the session timeout parameter is in the `web.xml` file in the following directory:

`<destination_MT_OH>/j2ee/OC4J_BI_Forms/applications/discoverer/web/WEB-INF`

To set the session timeout parameter, add it to the `web.xml` file immediately following the line that reads `</servlet>` and before the bottom of the `<web-app>` section of the file, as shown below:

```
<web-app>
    <servlet>
    ...
    </servlet>
    <session-config>
       <session-timeout>15</session-timeout>
    </session-config>
</web-app>
```

7.3.5 Upgrading Discoverer Viewer Customizations

A number of files control the appearance and behavior of OracleAS Discoverer Viewer. Those files might have been modified to customize Discoverer Viewer to meet certain customer-specific requirements. The installation of Oracle Application Server 10*g* (9.0.4) installs new files with different names. Specifically:

- `disco4iv.xml` is renamed `configuration.xml` and `viewer_config.xml` is renamed `ui_config.xml`
- The content of various `*.xsl` files is moved into new `*.xsl` files.

If the original configuration files and `*.xsl` files were modified, then you must edit the new 10*g* (9.0.4) files and reapply the changes from the Release 4.1 files. The configuration files do not correspond one-to-one by name in the old and new releases, so you will need to determine the file in which a modification should be placed. See Table 7–3 for file names.

> **Note:** Do not simply replace the new files with the Release 4.1 files (i.e. do not simply rename the new files). Follow the steps in this section to upgrade only the customizations, not all of the differences, from the original Release 4.1 files to the new files.

To upgrade Discoverer Viewer customizations:

1. For each original Release 4.1 file in *<source_MT_OH>*`/Apache/Apache/htdocs/disco4iv/html`:
   a. Compare the original Release 4.1 file with the equivalent Version 9.0.4 file in the *<destination_MT_OH>*`/J2EE/OC4J_BI_Forms/applications/discoverer/web/` subdirectories. Table 7–3 lists the Release 4.1 files and the equivalent 10*g* (9.0.4) files.
   b. Edit the equivalent 10*g* (9.0.4) file to incorporate any customizations found in the original Release 4.1 file.

> **Note:** If you modify the `configuration.xml` file, consider using Oracle Enterprise Manager Application Server Control to make the changes. The Oracle Enterprise Manager Application Server Control provides a user interface for editing settings in the `configuration.xml` file, and provides useful information about each setting.

Table 7–3 Equivalent Versions of Discoverer Files

Oracle9*i*AS Release 1 (1.0.2.2.x) Discoverer File	Oracle Application Server 10*g* (9.0.4) Discoverer File
`disco4iv.xml`	*<destination_MT_OH>*`/j2ee/OC4J_BI_Forms/applications/discoverer/web/WEB-INF/configuration.xml`
`viewer_config.xml`	(UNIX) *<destination_MT_OH>*`/j2ee/OC4J_BI_Forms/applications/discoverer/web/common/xsl/ui/ui_config.xml` (Windows) *<destination_MT_OH>*`/j2ee/OC4J_BI_Forms/applications/discoverer/web/common/xsl/ui_config.xml`

Table 7–3 (Cont.) Equivalent Versions of Discoverer Files

Oracle9*i*AS Release 1 (1.0.2.2.x) Discoverer File	Oracle Application Server 10*g* (9.0.4) Discoverer File
errors.xsl functions.xsl gui_components.xsl page_layouts.xsl render_table.xsl scripts.xsl style.xsl	Various new XSL files in the following directories: ■ <*destination_MT_OH*>/j2ee/OC4J_BI_Forms/applications/discoverer/web/common/xsl ■ <*destination_MT_OH*>/j2ee/OC4J_BI_Forms/applications/discoverer/web/viewer_files/xsl

7.4 Notes About Upgrading to OracleAS Discoverer

This section identifies characteristics of the upgraded OracleAS Discoverer component in Oracle Application Server 10*g* (9.0.4) of which you should be aware.

7.4.1 Oracle Applications EULs

When upgrading an Oracle Applications EUL, be aware that the MAXEXTENTS storage property of the EUL tables might have been increased to a value greater than the MAXEXTENTS storage property of the tablespace in which the EUL was created. If this situation exists, any attempt to upgrade the EUL will fail with an ORA01631 error, because:

- The upgrade process uses the original EUL tablespace's MAXEXTENTS value when creating the new EUL tables
- The new EUL tables are therefore not large enough to contain the data from the original EUL tables

If the MAXEXTENTS value of the EUL tables is greater than the MAXEXTENTS value of the EUL's tablespace, ask your database administrator to increase the MAXEXTENTS value of the original EUL's tablespace before attempting to upgrade the EUL. The new EUL tablespace and the new EUL tables will be created with the larger MAXEXTENTS value.

7.4.2 EULs Containing Analytic Functions

When you upgrade a Discoverer Release 4.1 EUL to a Discoverer Version 5 EUL, new Oracle9i analytic functions are added to EUL tables.

Where existing user-defined functions have the same name (or the same unique identifier) as the new functions, Discoverer does the following:

- Renames the existing user-defined function
- Displays a log message at the end of the upgrade process
- Provides the option to save the messages to a log file

Note: Discoverer's internal reference system ensures that end users can still open workbooks that contain renamed user-defined functions.

7.4.3 The Video Stores Tutorial

The Video Stores tutorials are specific to particular Discoverer releases.

When you upgrade to a new version of Discoverer, Oracle Corporation recommends that you re-install the Video Stores tutorial using the appropriate version of Discoverer Administrator instead of upgrading the tutorial.

7.4.4 Scheduled Workbooks

When you upgrade a Release 4.1 EUL to a Version 5 EUL, scheduled workbooks are automatically upgraded. However, scheduled workbook results are not copied to the Version 5 EUL. The results for an upgraded scheduled workbook will only be available after the scheduled workbook has next run.

7.4.5 Workbooks Saved in .dis Files

If Discoverer Desktop users save workbooks to the filesystem in `.dis` files, the `.dis` files must be upgraded before they can be used with the latest version of Discoverer.

To upgrade the `.dis` files, open the `.dis` files in the latest version of Discoverer and save the files back to the filesystem, or to the database (to open them in Discoverer Plus and Discoverer Viewer). Note that after a `.dis` file has been upgraded, it is not backward-compatible with earlier versions of Discoverer Desktop.

7.4.6 Discoverer systems that use the EUL Gateway

A Discoverer system might use the EUL Gateway to populate EULs with metadata from a source other than the database's on-line dictionary (e.g. from Oracle Designer).

If you upgrade such a system, you will have to re-install and re-configure the EUL Gateway after the upgrade process is complete. For more information about installing and configuring the EUL Gateway, see the `eulgatew.doc` document located in the `<destination_MT_OH>\discoverer\kits` directory.

7.4.7 Identifiers Containing Invalid Characters

In a future release of Discoverer, there will be a change to the valid characters that can be used in identifiers and some characters will be de-supported.

If you upgrade an EUL that has identifiers containing invalid characters, messages will be displayed indicating the affected identifiers. Modify any identifiers that use the invalid characters so that the identifiers can be used in future releases of Discoverer.

7.5 Validating the OracleAS Discoverer Upgrade

Follow the steps below to verify that OracleAS Discoverer is functioning correctly after the upgrade:

1. Connect to the URLs in OracleAS Discoverer 10*g* (9.0.4) and enter TNS information. Verify that this works in Discoverer Plus and Discoverer Viewer.
2. Ensure that any configuration changes or customizations that were made to Discoverer Plus and Discoverer Viewer in Release 1 (1.0.2.2.x) are present in the new URL connections.

8

Upgrading to OracleAS Forms Services

This chapter contains step-by-step instructions for upgrading Oracle9*i*AS Forms. It contains the following sections:

8.1 OracleAS Forms Services Upgrade Items

An upgrade item is an aspect of configuration that is acted upon during upgrade: a file, executable, or a setting that you must add, change, or delete/replace in the 10*g* (9.0.4) installation. Oracle9*i*AS Forms Services upgrade items are described in Table 8–1.

Table 8–1 OracleAS Forms Services Upgrade Items

Upgrade Item	Location in Release 1 (1.0.2.2.x) Oracle home	Location in 10*g* (9.0.4) Oracle home	Notes
Oracle HTTP Server configuration file: `6iserver.conf` (upgrades to `forms90.conf`)	`6iserver/conf/`	`/forms90/server`	Contains virtual path mappings.
Servlet environment file: `default.env`	`6iserver/forms60/server`	`/forms90/server`	Contains environment variables settings for the Forms Servlets runtime environment.
Configuration files with Forms servlet alias: `jserv.properties` (upgrades to `web.xml`)	`/Apache/jserv/conf`	`/j2ee/OC4J_BI_Forms/ /applications/ forms90app/forms90web/ WEB-INF/web.xml`	Contains Forms servlet aliases.
Application configuration file: `formsweb.cfg`	`6iserver/forms60/server`	`/forms90/server`	Contains Forms Services application configuration information.
Forms servlet template html files: (`*.htm`, `*.html`)	`6iserver/forms60/server`	`/forms60/server`	Default and user defined Forms servlet template HTML files.
Forms application modules (`fmb`/`fmx` files)			Forms modules (`fmb` and `fmx` files) deployed to Oracle9*i*AS Release 1 (1.0.2.2.x) Forms services must be upgraded to be deployed to OracleAS Forms Services in Oracle Application Server 10*g* (9.0.4).

8.2 Components Related to OracleAS Forms Services

This section discusses OracleAS Forms Services' relationship to other components. OracleAS Forms Services integration with Oracle Application Server is dependent on a set of OracleAS Forms Services configuration files. The dependencies are listed below in Table 8–2.

Table 8–2 Oracle Application Server Component and OracleAS Forms Services Configuration File Dependencies

Component	Configuration File
Oracle HTTP Server	`forms90.conf`
Oracle Application Server Containers for J2EE	`web.xml`, `forms90app.ear`
Oracle Application Server Single Sign-On/Oracle Internet Directory	`formsweb.cfg`
Oracle Enterprise Manager	`formsweb.cfg`
Oracle Application Server Reports Services	Forms internal PL/SQL built-in

8.3 OracleAS Forms Services Upgrade Tasks

This section explains how to perform the Oracle9*i*AS Forms Services upgrade. It is divided into the following sub-sections:

You will perform only the tasks in that apply to the method of deployment used, since the deployment options differ between the old and new releases, as follows:

Oracle9*i*AS Forms Services Release 6*i* in Oracle9*i*AS Release 1 (1.0.2.2.x) supports the following deployment options:

Web interfaces

- Static HTML files
- Common Gateway Interface
- Forms Servlet

Forms Servlet Request dispatcher types (Listeners)

- Forms Listener

- Forms Listener Servlet

In Oracle Application Server 10*g* (9.0.4), not all of these are supported. The OracleAS Forms Services architecture only supports:

Web interface:

- Forms Servlet

Forms Servlet Request dispatcher types (Listener)

- Forms Listener Servlet

8.3.1 Upgrade Recommendations and Troubleshooting Tips

Consider the following recommendations and considerations while upgrading Forms applications to Oracle Application Server 10*g* (9.0.4):

- Keep the Oracle9*i*AS Release 1 (1.0.2.2.x) installation until successful deployment and testing of applications to Oracle Application Server 10*g* (9.0.4).
- Deploy Oracle9*i*AS Forms Services 6*i* using the new Listener Servlet architecture.
- Use the Forms servlet rather than the Forms CGI.
- Upgrade source files first, and back up and secure all application files.
- Replace `Run_Product` calls to integrated Reports with `Run_Report_Object` calls to Oracle Application Server Reports Services (or use the PL/SQL conversion utility, FMA, in Forms).
- Install Oracle Application Server 10*g* (9.0.4) and configure the `forms90/server/formsweb.cfg` file with the information used by your applications.
- Copy the environment files used by the applications to the same directory.
- Copy the upgraded Forms application module files to the computer on which Oracle Application Server 10*g* (9.0.4) is installed, if it is not the same computer.
- After starting Oracle Application Server 10*g* (9.0.4), access the Forms Services Listener Servlet test page with this URL:

  `http://<`*`hostname`*`>:<`*`port`*`>/forms90/f90servlet/admin`

- Verify that the application setting is added to the `formsweb.cfg` file and that the environment variable `Forms90_Path` contains the directory of the application modules.
- Verify that you can connect to the database using SQL*Plus.
- Use the following URL to invoke upgraded applications:

  `http://<`*`hostname`*`>:<`*`port`*`>/forms90/f90servlet?config=<`*`your application name`*`>`

8.3.2 Upgrading the OracleAS Forms Services Application Modules

This section provides instructions for upgrading from Forms Application Modules (`fmb` files) that were deployed in Oracle9iAS Forms Services Release 6i. Follow these steps to upgrade Forms Application Modules (`fmb` files) deployed in Oracle9*i*AS Forms Services Release 6*i* to an OracleAS Forms Services.

1. Copy the Forms application files to a new directory.
2. Use the Forms Migration Assistant to upgrade the Forms Application Modules (`fmb` files).
3. Use the Forms Compiler (`f90genm.sh` or `ifcmp90.exe`) to regenerate the Forms Application executable files (`fmx` files).

For more information, see the document *Migrating Forms Applications from Forms 6i* at:

`http://otn.oracle.com/documentation/9i_forms.html`

8.3.3 Upgrading the Common Gateway Interface (CGI) to the Oracle Forms Servlet

This section provides instructions to upgrade Forms CGI to the Forms Servlet deployment. Follow these steps if you are using the Oracle9*i*AS Forms Services Common Gateway Interface to dynamically render the Forms Applet start HTML file for applications.

The CGI deployment for Forms applications was introduced in Oracle9*i*AS Forms Services Release 6*i* to enable the Forms Applet start HTML file to be rendered dynamically. The Forms CGI uses the `formsweb.cfg` configuration file and an HTML template to create the start HTML file for an application. The CGI interface is configured by an entry in the Forms HTTP configuration file `6iserver.conf` (it is referenced by an `Include` directive in the Oracle HTTP Server `oracle_apache.conf` file), which contains a `ScriptAlias` directive identifying `dev60cgi` for the directory structure containing the `ifcgi60.exe` file.

The Forms servlet renders the HTML in the same manner as the CGI, but also provides an automatic browser type detection, supporting the Internet Explorer native VM. The Forms servlet is configured when you install OracleAS Forms Services, and is named `f90servlet`.

To access the Forms Servlet, request the URL:

`http://<`*`hostname`*`>:<`*`port`*`>/forms90/f90servlet`

This URL is similar to the URL used with the CGI Interface in Oracle9*i*AS Forms Services Release 6*i*. To call an application configured as `myapp` in the custom configuration section of the `forms90/server/formsweb.cfg` file, request the URL:

`http://<`*`hostname`*`>:<`*`port`*`>/forms90/f90servlet?config=myapp`

The Forms Servlet is automatically configured during installation. The installer creates a virtual path `/forms90/` pointing to the OracleAS Forms Services configuration, `forms90app` and `forms90web`, in the Oracle Application Server Containers for J2EE Business Intelligence and Forms instance (`<`*`destination_MT_OH`*`>/j2ee/oc4j/OC4J_BI_Forms`).

Follow these steps to upgrade an Oracle9*i*AS Forms Services Release 6*i* CGI environment to an OracleAS Forms Services servlet environment:

1. Copy all of the application-specific configurations from `<`*`source_MT_OH`*`>/Forms60/Server/formsweb.cfg` and append them to `<`*`destination_MT_OH`*`>/forms90/server/formsweb.cfg`.

> **Note:** Do not copy and replace the entire `formsweb.cfg` file in <*source_MT_OH*> to <*destination_MT_OH*>. The file in Release 6*i* is different from the OracleAS Forms Services file in 10*g* (9.0.4). Copy only the application configuration to <*destination_MT_OH*>`/forms90/server/formsweb.cfg`.

2. Configure `Forms90_Path` in the `forms90/server/default.env` file to point to the upgraded OracleAS Forms Services application modules.

> **Note:** You can create a new environment file by copying `default.env`, modifying it for use with a particular application, and adding `envFile=`<*created environment file*> to the custom application section in the `formsweb.cfg` file.

3. If you changed the Oracle9*i*AS Forms HTML template files, then make the same changes to the OracleAS Forms Services HTML template files.

> **Note:** You must make the changes in all three files: `basejini.htm`, `basejpi.htm`, and `baseie.htm`, because the servlet supports JInitiator, Java plug-ins, and the Internet Explorer native VM.

8.3.4 Upgrading Static HTML Start Files to Generic Application HTML Start Files

Each application deployed to OracleAS Forms Services has a custom application definition, configured in the `formsweb.cfg` configuration file. It automatically inherits the general system settings, such as the JInitiator version used or the names and locations of the base HTML template files.

The name of the custom application definition becomes part of the Forms application URL. The following custom settings define two different applications:

```
[MyHR_app]
IE=Jinit
serverURL=/forms90/l90servlet
Form = hr_main.fmx
lookAndFeel=oracle
Otherparams=myParam1=12
Userid=scott/tiger@orcl
```

The following URL invokes this application:

`http://`<*hostname*>`:`<*port*>`/forms90/f90servlet?config=MyHR_app`

Another custom application definition might look like this:

```
[booking_app]
IE=native
ServerURL=/forms90/l90servlet
Form = book.fmx
lookAndFeel=oracle
Otherparams=
Userid=
```

The following URL invokes this application:

`http://`<*hostname*>`:`<*port*>`/forms90/f90servlet?config=booking_app`

For each static HTML file, you must create a custom application definition. Part of the static HTML file is the `archive` parameter directive, specifying at least the `f90all.jar` file in OracleAS Forms Services. If you added a custom archive file, then the `archive` parameter directive would resemble the following: `Archive=f90all.jar,custom.jar`. Using the Forms servlet and the `formsweb.cfg` file, the archive settings are defined under the User Parameter section. All custom application settings inherit these values, so you don't have to explicitly set this parameter, unless you add a `custom.jar` file as required by an application.

If `custom.jar` was added, then you can add the following lines to the custom application definition. The example below assumes that you are using JInitiator or another VM, but not Internet Explorer native.

```
[booking_app]
archive_jini=f90all_jinit.jar, custom.jar
archive=f90all.jar, custom.jar
ServerURL=/forms90/l90servlet
Form = book.fmx
lookAndFeel=oracle
Otherparams=
Userid=
```

Follow these steps to upgrade applications:

1. Edit the `forms90/server/default.env` file, adding the location of the OracleAS Forms Services application modules to the `Forms90_Path`.
2. Edit the `forms90/server/formsweb.cfg` file, appending a custom application section for each static HTML application that you want to replace.
3. Name each custom application section, using a name that contains no spaces and is enclosed in square brackets, for example: `[booking_app], [MyHR_app]`.
4. Start the application using this URL:

   `http://<`*`hostname`*`>:<`*`port`*`>/forms90/f90servlet?config=<`*`name`*`>`

8.3.4.1 Using Static HTML Files with OracleAS Forms Services

If you need to, you can continue to use static HTML files in OracleAS Forms Services. However, with static HTML files, some features (such as Oracle Application Server Single Sign-On) are not available for use by Forms applications.

The Forms Listener servlet by default points to `/forms90/l90servlet` after installation. To use static HTML files in OracleAS Forms Services, you must modify each static start HTML file to include a value for the `serverURL` parameter. The `serverPort` and `serverHost` parameters are no longer used, and can be left undefined. OracleAS Forms Services uses JInitiator version 1.3.x, so you must also change those settings. The required values are found in the `forms90/server/formsweb.cfg` file.

Follow these steps to use static HTML files with OracleAS Forms Services:

1. Configure `Forms90_Path` in the `forms90/server/default.env` file to point to the upgraded OracleAS Forms Services application modules.
2. Create virtual directories in the `<`*`destination_MT_OH`*`>/forms90/server/forms90.conf` file to point to the location of the static HTML start files.
3. Modify the application start HTML files as follows:
   a. Add the `serverURL` value `/forms90/l90servlet`.

b. Change the JInitiator version number.

4. Change the `codebase` parameter to `forms90/java`.
5. Navigate to `<destination_MT_OH>/j2ee/OC4J_BI_Forms/applications/forms90app/forms90web/WEB-INF` and edit the `web.xml` file.
6. Set the envFile initialization parameter for the Listener Servlet to point to the environment file (usually `<destination_MT_OH>/forms90/server/default.env`).

After editing, the entry in the `web.xml` file for the Forms listener servlet should resemble the following:

```
<!--Forms 9i listener servlet-->
<servlet>
  <servlet-name>l90servlet</servlet-name>
     <servlet-class>oracle.forms.servlet.ListenerServlet</servlet-class>
  <init-param>
     <param-name>envFile</param-name>
     <param-value>destination_MT_OH/forms90/server/default.env</param-value>
  </init-param>
</servlet>
```

8.3.5 Upgrading the Forms 6*i* Listener to the Forms Listener Servlet

The Forms 6*i* Listener is a C program that starts a Forms runtime process on behalf of an incoming Forms Web request. The Forms Web runtime process is then directly accessed by the Forms client applet, using a direct socket or an HTTP socket connection. The Forms Listener is then no longer involved in the application Web client-server communication process, and is free to handle other incoming Web requests.

The Forms Listener Servlet, a Java program, also takes incoming Web requests for a Forms application and starts the Forms server-side Web runtime process. Unlike the Forms 6*i* Listener, the Forms Listener Servlet remains between the Forms application applet-server communication.

While the Forms 6*i* Listener listens on a specific port (by default, 9000), the Forms Servlet doesn't need an extra port, and is accessed by the HTTP listener port. The Forms Listener Servlet was introduced in the Forms 6*i* patch 4, and is the only listener supported in Forms Services.

The Forms Listener Servlet is automatically configured during the Oracle Application Server 10*g* (9.0.4) installation. The installer creates a virtual path `/forms90/` pointing to the OracleAS Forms Services configuration, `forms90app` and `forms90web`, in the Oracle Application Server Containers for J2EE Business Intelligence and Forms instance (`<destination_MT_OH>/j2ee/oc4j/OC4J_BI_Forms`).

To access the Forms Listener Servlet servlet, request the following URL:

`http://<hostname>:<port>/forms90/f90servlet/admin`

Ability to access this page means that the Forms Listener Servlet is configured and ready to use. `f90servlet` is the access name configured for the Forms Servlet during installation. The name of the Listener Servlet is `l90servlet`.

If the Forms Listener Servlet is accessed with the Forms servlet, then only the custom application settings from the `Forms60/server/formsweb.cfg` file need to be appended to the `forms90/server/formsweb.cfg` file. All application

configurations automatically inherit the `serverURL` parameter value `/forms90/l90servlet` from the global system parameter settings.

Note: Do not copy and replace the entire `formsweb.cfg` file in *<source_MT_OH>* to *<destination_MT_OH>*. The file in Release 6*i* is different from the OracleAS Forms Services file in 10*g* (9.0.4). Copy only the application configuration to *<destination_MT_OH>*`/forms90/server/formsweb.cfg`.

To change a Forms application deployment from the Forms Listener architecture to the Listener Servlet architecture, you need only supply a value for the `serverURL` parameter in the `formsweb.cfg` file. During installation, this parameter is set to `/forms90/l90servlet`.

Follow these steps to upgrade to the Forms Listener Servlet:

1. Copy the Forms application files to a new directory and upgrade them to OracleAS Forms Services modules as described in Section 8.3.2, "Upgrading the OracleAS Forms Services Application Modules". on page 8-4.
2. Edit the `forms90/server/default.env` file to add the location of the upgraded Forms application modules to the `Forms90_Path` variable.
3. Copy all of the custom application settings from *<source_MT_OH>*`/Forms60/Server/formsweb.cfg` and append them to *<destination_MT_OH>*`/forms90/server/formsweb.cfg`.

   Note: Do not copy and replace the entire `formsweb.cfg` file in *<source_MT_OH>* to *<destination_MT_OH>*. The file in Release 6*i* is different from the OracleAS Forms Services file in 10*g* (9.0.4). Copy only the application configuration to *<destination_MT_OH>*`/forms90/server/formsweb.cfg`.

   ---

4. If an application requires its own environment file, then instead of defining a separate servlet alias for the Listener Servlet, set the `envFile` parameter in the custom application definition section in *<destination_MT_OH>*`/forms90/server/formsweb.cfg` to point to the new environment file. For example:

   ```
   envFile=myEnvFile.env
   ```

   where `myEnvFile.env` is located in the `forms90/server` directory.
5. If you changed the Oracle9*i*AS Forms Services Release 6*i* HTML template files, then make the same changes to the OracleAS Forms Services HTML template files.

Note: If you need to change the underlying HTML files, you should make a copy of the provided template files before editing them. Save the edited HTML files under a different name, and leave the default templates provided with the installation unchanged. This prevents overwriting of your customized HTML template files when patch sets are applied to the application.

To use your own template files with applications, use these parameters in the system section, or one of your custom application definitions:

```
baseHTML=<your base template>.htm
baseHTMLJinitiator=<your base jinit>.htm
baseHTMLie=<your base ie>.htm
```

6. Start the application with this URL:

   ```
   http://<hostname>:<port>/forms90/f90servlet?
   config=<application>
   ```

8.3.6 Upgrading the Forms Listener Servlet Architecture to OracleAS Forms Services

In Oracle9*i*AS Forms Services Release 6*i*, the Listener Servlet, if not aliased, is accessed by the `oracle.forms.servlet.ListenerServlet`. The Listener Servlet configuration exists in the `jserv.properties` file and the `zone.properties` file.

In OracleAS Forms Services, the Forms Listener servlet is the same except for the servlet names, which are `f90servlet` and `l90servlet`, and the servlet container, which is now Oracle Application Server Containers for J2EE (OC4J). As in Oracle9*i*AS Release 1 (1.0.2.2.x), the configuration is performed during installation. The Listener Servlet configuration in OC4J is stored in `<ORACLE_HOME>/j2ee/OC4J_BI_Forms/applications/forms90app/forms90web/WEB-INF/web.xml`. Some initialization parameters, like the `envFile` parameter, need no longer be configured with the servlet engine, because they are moved to the `formsweb.cfg` file.

The Forms Listener Servlet is automatically configured during the Oracle Application Server 10*g* (9.0.4) installation. The installer creates a virtual path `/forms90/` pointing to the OracleAS Forms Services configuration, `forms90app` and `forms90web`, in the Oracle Application Server Containers for J2EE Business Intelligence and Forms instance (`<destination_MT_OH>/j2ee/oc4j/OC4J_BI_Forms`).

To access the Forms Listener Servlet servlet, request the following URL:

```
http://<hostname>:<port>/forms90/f90servlet/admin
```

Ability to access this page means that the Forms Listener Servlet is configured and ready to use. `f90servlet` is the access name configured for the Forms Servlet during installation. The name of the Listener Servlet is `l90servlet`.

Follow these steps to upgrade the Listener Servlet architecture to OracleAS Forms Services:

1. Copy the Forms application files to a new directory and upgrade them to OracleAS Forms Services modules.
2. Edit the `forms90/server/default.env` file, adding the location of the upgraded Forms application modules to the `Forms90_Path` variable.

3. Copy all of the custom application settings from <*source_MT_OH*>/Forms60/Server/formsweb.cfg and append them to <*destination_MT_OH*>/forms90/server/formsweb.cfg.

   Note: Do not copy and replace the entire formsweb.cfg file in <*source_MT_OH*> to <*destination_MT_OH*>. The file in Release 6*i* is different from the OracleAS Forms Services file in 10*g* (9.0.4). Copy only the application configuration to <*destination_MT_OH*>/forms90/server/formsweb.cfg.

4. If an application requires its own environment file, then instead of defining a servlet alias for the Listener Servlet, set the envFile parameter in the custom application definition section in <*destination_MT_OH*>/forms90/server/formsweb.cfg to point to the new environment file. For example:

   ```
   envFile=myEnvFile.env
   ```

   where myEnvFile.env is located in the forms90/server directory.

5. If you changed the Oracle9*i*AS Forms Services Release 6*i* HTML template files, then make the same changes to the OracleAS Forms Services HTML template files.

   Note: If you need to change the underlying HTML files, you should make a copy of the provided template files before editing them. Save the edited HTML files under a different name, and leave the default templates provided with the installation unchanged. This prevents overwriting of your customized HTML template files when patch sets are applied to the application.

   To use your own template files with applications, use these parameters in the system section, or one of your custom application definitions:

   ```
   baseHTML=<your base template>.htm
   baseHTMLJinitiator=<your base jinit>.htm
   baseHTMLie=<your base ie>.htm
   ```

6. Start the application with this URL:

   http://<*hostname*>:<*port*>/forms90/f90servlet?config=<*application*>

8.3.7 Upgrading Load Balancing

The method of upgrading the load balancing in Oracle9*i*AS Forms Services 6*i* depends on the deployment method used.

- With the Forms 6*i* listener, the Metrics Server (a separate process) performs load balancing.
- With the Forms 6*i* servlet, load balancing is configured with the JServ servlet engine, using round robin load balancing among JServ engines.
- In OracleAS Forms Services, load balancing is managed by `mod_oc4j`, an Oracle HTTP Server module. It binds Web requests to the servlet container processing the Forms Servlet and the Forms Listener servlet.

8.3.8 Usage Notes

This section contains hints and tips that may be useful in the upgrade.

8.3.8.1 Deploying Icon Images with the Forms Servlet

Using static HTML start files in Oracle9*i*AS Forms Services Release 6*i* allowed storage of images in a location relative to the start HTML file. The Forms Servlet in OracleAS Forms Services does not support this.

The alternative is to use the `imagebase` parameter with the value of `codebase` as the location for the icon images used by applications. The `codebase` value refers to the `forms90/java` directory, which contains all of the Forms client Java archive files. For performance reasons, it is not a good idea to store images here.

Instead, you should bundle the icons into a separate archive file, which improves performance because archives are cached permanently on the client. Follow these steps to create this archive file.

1. Verify that the `jar` command succeeds. If it does not, then you need to ensure that there is a JDK installed on your system with a correct PATH environment variable entry (pointing to the <JDK_HOME>`/bin` directory).
2. Navigate to the directory containing the application images and issue the command:

   `jar -cvf <`*`application`*`>_images.jar *.<`*`extension`*`>`

   where:

   - *`application`* is the name of the application
   - *`extension`* is the extension of the image file (e.g., .gif)

   A jar file, <*`application`*>`_images.jar`, is created in the current directory.
3. Copy <*`application`*>`_images.jar` to the `forms90/java` directory.
4. Edit the `formsweb.cfg` file, adding the `imageBase=codebase` parameter to the custom application section for the application.
5. Add the <*`application`*>`_images.jar` file to the archive path used by the application by adding the following lines to the custom application section:

```
archive_jini-f90all_jinit.jar,<application>_images.jar
archive_f90all.jar,<application>_images.jar
archive_ie-f90all_.jar,<application>_images.jar
```

Note: `archive_ie` should contain `f90all.cab`, which is the better archive format when using Internet Explorer 5.0 native VM. Because you can't mix archives and `.cab` files, you must create a `.cab` file for all of the images or use `f90all.jar` for Forms applications. The former is recommended.

See the *Oracle Application Server Forms Services Deployment Guide*, Chapter 3, in the Oracle Application Server documentation library for more information on deploying custom icon files with OracleAS Forms Services.

8.3.8.2 Upgrading Integrated Calls to Oracle9i Reports to use Oracle Application Server Reports Services

In Oracle Application Server 10*g* (9.0.4), integrated calls to Oracle Reports in Forms are no longer handled by a client-side background engine. OracleAS Forms Services requires that applications use the `Run_Report_Object` built-in, calling Oracle Application Server Reports Services to process integrated reports. Oracle Application Server Reports Services is set up as part of the Business Intelligence and Forms installation.

Follow these steps to upgrade the call:

1. Change all occurrences of `Run_Product (Reports,...)` to the equivalent call using `Run_Report_Object()`.
2. Add the location of the application's Reports modules to use the `Reports90_Path` of Oracle Application Server Reports Services.
3. Change `Run_Report_Object` to referenceOracle Application Server Reports Services.

   For more information, see http://otn.oracle.com/products/forms/pdf/10g/frmrepparamform.pdf

8.3.8.3 Creating Forms Listener Servlet Alias Names in OC4J

In Oracle9*i*AS Forms Services Release 6*i*, before patch 8, it was necessary to create alias names for the Forms servlet in the `<`*ORACLE_HOME*`>/Apache/Apache/JServ/conf/zone.properties` file in order to use individual environment files for different applications. The Forms servlet in OracleAS Forms Services does not require this. You can set the environment file name in the `formsweb.cfg` file using the `envFile` parameter, shown below:

```
envFile=myApp.env
```

Alias names for the Forms servlet are no longer created in `<`*ORACLE_HOME*`>/Apache/Apache/JServ/conf/zone.properties`. Instead, they are created in `<`*destination_MT_OH*`>/j2ee/OC4J_BI_Forms/applications/forms90app/forms90web/WEB-INF/web.xml`.

To create the alias names, copy the content between the `<servlet>` and `</servlet>` tags and change the servlet's name. To create a URL mapping for the new servlet alias name, add the following to the file:

```
<servlet-mapping>
<servlet-name>new servlet name</servlet-name>
<url-pattern>/new url name*</url-pattern>
</servlet-mapping>
```

8.3.8.4 Accessing the Listener Servlet Administration Page

You can display a test page for the Listener Servlet in Oracle9*i*AS Forms Services Release 6*i* by accessing the following URL:

`http://<hostname>:<port>/servlet/oracle.forms.servlet.ListenerServlet`

The information displayed depends on the value of the initialization parameter `TestMode`. This parameter is set in the `<source_MT_OH>/Apache/Apache/JServ/conf/zone.properties` file.

You can display the test page for OracleAS Forms Services with the following URL:

`http://<hostname>:<port>/forms90/f90servlet/admin`

The information displayed depends on the value of the initialization parameter `TestMode`. This parameter is set in the `<destination_MT_OH>/j2ee/OC4J_BI_Forms/applications/forms90app/forms90web/WEB-INF/web.xml` file. An example is shown below:

```
<init-param>
<!-- Display sensitive options on the /admin page ? -->
    <param-name>TestMode</param-name>
    <param-value>true</param-value>
</init-param>
```

8.4 Validating the OracleAS Forms Services Upgrade

After you complete the upgrade tasks, ensure that the upgraded version of the OracleAS Forms Services is working as expected. You must devise and perform specific tests for applications and configuration elements that are unique to your site. Compare the performance and characteristics of each application in the source and destination installations.

In Oracle9*i*AS Release 1 (1.0.2.2.x), the application URL is typically:

`http://<hostname>:<port>/servlet/<forms servlet alias>?<forms application name>`

In Oracle Application Server 10*g* (9.0.4), the application URL is typically:

`http://<hostname>:<port>/forms90/<forms servlet alias>?<forms application name>`

9

Upgrading to OracleAS Reports Services

This chapter contains step-by-step instructions for upgrading Oracle9*i*AS Reports. It contains the following sections:

9.1 OracleAS Reports Services Upgrade Items

An upgrade item is an aspect of configuration that is acted upon during upgrade: a file, executable, or a setting that you must add, change, or delete/replace in the 10g (9.0.4) installation. OracleAS Reports Services upgrade items are described in Table 9–1.

Table 9–1 OracleAS Reports Services Upgrade Items

Upgrade Item	Location in Release 1 (1.0.2.2.x) Oracle home	Location in 10*g* (9.0.4) Oracle home	Notes
`cgicmd.dat`	`reports60/admin/server`	`reports/conf`	File contents are the same, only the location has changed.
`uifont.ali`	`6iserver/guicommon6/tk60/admin`	`guicommon9/tk90/admin`	Cannot be directly overwritten. The 10*g* (9.0.4) file contains new entries, such as PDF subsetting and embedding.
`uiprint.txt`	`6iserver/guicommon6/tk60/admin`	`guicommon9/tk90/admin`	File contents are the same, only the location has changed.
`reports60.sh/reports.sh`	`6iserver/reports60.sh`	`bin/reports.sh`	You must set `REPORTS_PATH` to specify the default location of RDF & PLLs.
`proxyinfo.xml`	Not present	`reports/conf/proxyinfo.xml`	New configuration file.
`cache` folder	`6iserver/reports60/server/cache`	`reports/cache`	Folder location change.
`printer` folder	`6iserver/reports60/admin/printer`	`reports/printers`	Folder location change.
`resource` folder	`6iserver/reports60/admin/resource`	`reports/res`	Folder location change.

Table 9–1 (Cont.) OracleAS Reports Services Upgrade Items

Upgrade Item	Location in Release 1 (1.0.2.2.x) Oracle home	Location in 10*g* (9.0.4) Oracle home	Notes
`mesg` folder	`6iserver/reports60/mesg`	`reports/mesg`	Folder location change.
`templates` folder	`6iserver/reports60/admin/templates`	`reports/templates`	Folder location change.
`spoolcmd.sh`	`spoolcmd.sh`	`rwlpr.sh`	Executable name change
`rwgenkey.sh`	Not present	`rwgenkey.sh`	Executable name change
Default `.PPD` and other `.ppd` files	`6iserver/guicommon6/tk60/admin/ppd`	`guicommon9/tk90/admin/ppd`	UNIX platforms only. Printer definition files, contents are the same.

9.2 Components Related to OracleAS Reports Services

The Oracle Application Server components listed below are integral to OracleAS Reports Services functionality:

- Oracle HTTP Server, modules for Reports CGI deployment
- OracleAS Web Cache
- Database
- OC4J BI_FORMS, the J2EE container for Reports Servlet and JSP deployment
- OracleAS Single Sign-On, for reports application security
- Oracle Internet Directory, the directory server
- OracleAS Portal, the Reports security repository
- Oracle Process Manager and Notification Server (OPMN), for reports services process monitoring
- Oracle Enterprise Manager 10*g*, to manage reports services
- OracleAS Forms Services, for application deployment
- Oracle Business Intelligence Beans, for graphing

9.3 OracleAS Reports Services Upgrade Tasks

This section explains how to perform the OracleAS Reports Services upgrade. "Related Documents" on page -xiii provides references to documents that may be helpful as you perform the steps.

1. Copy the `cgicmd.dat` file from `<source_MT_OH>/report60/admin/server` to `<destination_MT_OH>/reports/conf`.
2. Copy the customizations (any additions to, or changes from, the original configuration) from the `<source_MT_OH>/6iserver/guicommon6/tk60/admin/uifont.ali` file to the `<destination_MT_OH>/guicommon9/tk90/admin/uifont.ali` file. Do not copy the entire file; the base configuration differs between releases.
3. (UNIX only) Copy the `uiprint.txt` file from `<source_MT_OH>/6iserver/guicommon6/tk60/admin` to `<destination_MT_OH>guicommon9/tk90/admin`.

> **Note:** The contents of this file are the same between releases. In 10*g* (9.0.4), the variable `REPORTS_DEFAULT_DISPLAY=YES` is set by default in the `reports.sh` file during installation. Internally, the display surface is created using the `screenprinter.ppd` file. This setting makes the DISPLAY environment variable and valid printer unnecessary in UNIX. You need not edit this file unless you want to set `REPORTS_DEFAULT_DISPLAY=NO` in the `reports.sh` file.

4. (UNIX only) Copy the custom environment settings from `<source_MT_OH>/6iserver/reports60.sh` to `<destination_MT_OH>bin/reports.sh`. If the Reports ENV variable is in use, refer to the *Oracle9i Reports Obsolescense Plan: Oracle9i Reports Statement of Direction* at `http://otn.oracle.com/products/reports/pdf/movingto9i.pdf` for obsolescence and 10*g* (9.0.4) naming conventions.
5. (UNIX only) Change all references to `spoolcmd.sh` to `rwlpr.sh`. The executable's name has changed in 10*g* (9.0.4).
6. Register the Reports server in `opmn.xml` and `targets.xml` by executing the script `<destination_MT_OH>/bin/addNewServerTarget.sh` (UNIX) or `addNewServerTarget.bat` (Windows).
7. Set the default printer for the in-process server. See `http://download.oracle.com/docs/html/B10314_01/pbr_conf.htm`, Section 3.8, for instructions.
8. Register the installed in-process server with OracleAS Portal. See `http://download.oracle.com/docs/html/B10314_01/pbr_portal.htm#1007992`.

9.4 Validating the OracleAS Reports Services Upgrade

To ensure that the upgrade was successful, run the test report from the Oracle Application Server Reports Services demonstration CD, or known reports from the previous release with the same reports clients or new reports clients. Follow the steps in Running Reports Requests in `http://download.oracle.com/docs/html/B10314_01/toc.htm`

9.5 Configuring OracleAS Reports Services Security

In Oracle9*i*AS Release 1 (1.0.2.2.x), the security for OracleAS Reports Services was Portal-based, and the Reports server was not secure by default (security had to be enabled). In Oracle Application Server 10*g* (9.0.4):

- OracleAS Reports Services is secure by default
- Application and data security is provided by Oracle Application Server Single Sign-On and Oracle Internet Directory, and resource security is provided by OracleAS Portal.

This section explains how to configure and use the security features of OracleAS Reports Services.

9.5.1 Configuring Oracle Internet Directory

OracleAS Reports Services uses Oracle Internet Directory in the Infrastructure installation for the directory services portion of its security implementation. Follow these steps to configure Oracle Internet Directory for OracleAS Reports Services:

1. Set `SINGLESIGNON=NO` in `<destination_MT_OH>/reports/conf/rwservlet.properties`.
2. Restart the OC4J_BI_FORMS OC4J instance.
3. Start the Reports server in secure mode.

9.5.2 Using Oracle Application Server Single Sign-On

OracleAS Reports Services uses OracleAS Single Sign-On in the Infrastructure installation for the authentication portion of its security implementation. It is configured by default, so no configuration steps are required. However, note the following guidelines for its use:

- The Reports server must be started in secure mode.
- Optionally, you can set `SINGLESIGNON=YES` in `<destination_MT_OH>/reports/conf/rwservlet.properties` and then restart the OC4J_BI_FORMS OC4J instance.

9.6 Validating the Infrastructure Configuration

Validating the Infrastructure configuration involves testing the functioning of the database, OracleAS Single Sign-On, and Oracle Internet Directory. Follow these steps:

1. Test the database by following these steps:
   a. Start the OC4J_BI_FORMS instance in the middle tier.
   b. Configure the TNS entries for the database in the `<destination_MT_OH>/network/admin/tnsnames.ora` file.
   c. Run a sample report using the Reports servlet and in-process server deployed in the OC4J_BI_FORMS instance.

      If the report executes, the database was accessed successfully and the database connection to OracleAS Reports Services is functioning as it should.
2. Test the OracleAS Single Sign-On configuration by following these steps:
   a. Start the Reports server in secure mode.
   b. Submit a report request using the Reports servlet.

      The OracleAS Single Sign-On authentication screen appears.
   c. Enter a valid OracleAS Single Sign-On user ID and password.

      If the report executes, the report request was serviced and the OracleAS Single Sign-On configuration is functioning as it should.
3. Test the Oracle Internet Directory configuration by following these steps:
   a. Create users and groups in the LDAP directory.
   b. Start the Reports server in secure mode.

c. Submit a request using the Reports servlet, specifying the authid=user/password on the command line (where the user and password is one created in Step a.).

If the report executes, the report request was serviced and the Oracle Internet Directory is functioning as it should.

10

Upgrading OracleAS InterConnect

This chapter contains step-by-step instructions for upgrading OracleAS InterConnect. It contains the following sections:

10.1 Installing and Upgrading Hub Components, Adapters, and Development Kit

Follow the installation instructions in Oracle Application Server InterConnect components to install the Hub, Adapters, and Development Kits. The information provided in the installation procedure should reflect the configuration of the Release 1 (4.1) components. Ensure that the architecture described in the 10g (9.0.4) installation is the same as the Release 1 (4.1) architecture unless you intend to change the architecture.

10.2 Upgrading Metadata

Follow these steps to upgrade metadata:

1. Install the most recent Release 1 (4.1) Oracle Application Server InterConnect patch set.
2. Run the `oaiexport` script provided with the Release 1 (4.1) installation by issuing the following command:

   <source_InterConnect_OH>/oai/4.1/repository/<repo name>/oaiexport <file name> system/<system password> <connect string>

   where:

   - *`repo name`* is the name of the InterConnect Repository in the Oracle9*i*AS Release 1 (1.0.2.2.x) installation.
   - *`file name`* is the file name to which you wish to save the database schema export (e.g., `mytestrepo.dmp`).
   - *`system password`* is the password of the SYSTEM user in the hub database.
   - *`connect string`* is the TNS service name of the hub database. The service name of the default database will be used if no value is provided.

The metadata is exported to a file in the current directory.

3. Run the `oaiimport` script provided with the 10g (9.0.4) installation by issuing the following command:

   <destination_InterConnect_OH>/oai/9.0.4/repository/oaiimport <file name> <from user> system/<system password> <oaihub904 schema password> <connect string>

   where:

   - *`file name`* is the file name to which you wish to save the database schema export (e.g., `mytestrepo.dmp`).
   - *`from user`* is the user of the hub schema in the Oracle9*i*AS Release 1 (1.0.2.2.x) installation (`oaihub`, typically).
   - *`system password`* is the password of the SYSTEM user in the hub database.
   - *`oaihub904 schema password`* is the password for the 10*g* (9.0.4) hub database
   - *`connect string`* is the TNS service name of the hub database. The service name of the default database will be used if no value is provided.

   The file is imported into the 10g (9.0.4) hub database.

4. Create the tables in the hub schema database by issuing this command in the hub schema:

```
CREATE TABLE emdcomments (emdid NUMBER(10)
                              CONSTRAINT fk_emdcomments_emdid REFERENCES
                                   emd(id) ON DELETE CASCADE
                              CONSTRAINT nn_emdcomments_emdid NOT NULL,
                         name VARCHAR2(4000)
                              CONSTRAINT nn_emdcomments_name NOT NULL,
                         comments  LONG);
create table lcoextratypeattr(parentid NUMBER(10)
                                      CONSTRAINT fk_lcoextratypeattr_parent
                                      REFERENCES lco(id) ON DELETE CASCADE,
                              name     VARCHAR2(600),
                              types    LONG,
                              CONSTRAINT pk_lcoextratypeattr
                              PRIMARY KEY (parentid,name));
create table dsoextratypeattr(parentid NUMBER(10)
                                      CONSTRAINT fk_dsoextratypeattr_parent
                                      REFERENCES dso(id) ON DELETE CASCADE,
                              name     VARCHAR2(600),
                              types    LONG,
                              CONSTRAINT pk_dsoextratypeattr
                              PRIMARY KEY (parentid,name));
```

5. Using SQL*Plus, connect to the hub schema in the Infrastructure database and issue these commands:

```
update emd set type='AQ' where type='XML'

commit;
```

10.2.1 Upgrading Adapters

Follow these steps to upgrade adapters:

1. Install the OracleAS InterConnect 10g adapter, which corresponds to the existing adapter that you have. The prompts for hub database information refer to the Oracle9iAS Release 1 Hub Database.
2. When prompted for all other information, provide the values from your existing configuration.

10.2.2 Upgrading iStudio and SDKs

Follow these steps to upgrade iStudio and the SDKs:

1. Install the OracleAS InterConnect 10g Developer Kits. This includes the new version of the SDKs.
2. Install the OracleAS InterConnect 10g iStudio. As you create an iStudio project, the prompts for hub database information refer to the OracleAS 10g Hub Database.

10.3 Upgrading Management

Follow these steps to upgrade Management:

1. Install the Oracle Enterprise Manager (OMS and Java Console) that gets shipped with the 9.2 database.
2. Configure Oracle Enterprise Manager as per the install documentation for OracleAS InterConnect 10g.
3. Follow the instructions in the 10g (9.0.4) Oracle Application Server InterConnect installation documentation in the Oracle Application Server InterConnect documentation library (accessible from the Oracle Application Server documentation library).

Note: The upgraded Oracle Enterprise Manager Application Server Control cannot be used to manage Oracle Application Server InterConnect components. See the *Oracle Application Server InterConnect Installation Guide* for more information.

11

Upgrading to Oracle Workflow

This chapter contains step-by-step instructions for upgrading to Oracle Workflow. It contains the following sections:

11.1 Oracle Workflow Upgrade Items

An upgrade item is an aspect of configuration that is acted upon during upgrade: a file, executable, or a setting that you must add, change, or delete/replace in the 10g (9.0.4) installation. Oracle Workflow upgrade items are described in Table 11–1.

Table 11–1 Oracle Workflow Upgrade Items

Upgrade Item	Location in Release 1 (1.0.2.2.x) Oracle home	Location in 10*g* (9.0.4) Oracle home	Notes
Workflow process definition files	`wf/res/<lang>/*.wft`	`wf/res/<lang>/*.wft`	
Business Event system definition files	`wf/res/<lang>/*.xml`	`wf/res/<lang>/*.wfx`	File extension changes in new release.
SQL scripts	`wf/sql`	`wf/sql`	
Custom help files	`wf/doc/<lang>/wfcust/*.htm` or `wf/res/<lang>/wfcust/*.hlp`	`wf/doc/<lang>/wfcust/*.htm` or `wf/res/<lang>/wfcust/*.hlp`	

11.2 Oracle Workflow Upgrade Tasks

Follow these steps to upgrade Oracle Workflow from from Oracle9iAS Release 1 (v1.0.2.2) to Oracle Application Server 10*g* (9.0.4):

1. Install Oracle Workflow with 10g (9.0.4), including all pre- and post- installation steps as described in the Oracle Application Server InterConnect Installation Guide or the Oracle Workflow Installation Notes for Oracle Content Management SDK.

   The installation updates the Oracle Workflow server version in the database. See the Oracle Application Server InterConnect Installation Guide or the Oracle Workflow Installation Notes for Oracle Content Management SDK for detailed information.

2. Perform all Oracle Workflow setup steps for your 10g (9.0.4) installation as described in the "Setting Up Oracle Workflow" chapter of the Oracle Workflow Administrator's Guide. In particular, ensure that you:
   - Set your global Workflow preferences appropriately.
   - Set up a directory service for Oracle Workflow.
   - Configure the Workflow Notification Mailer using the Workflow Manager component of Oracle Enterprise Manager. Note that in 10*g* (9.0.4), the Workflow Notification Mailer is run through Workflow Manager as a Java-based service component, and is no longer configured through a separate text configuration file.
3. Copy all customized files from the source Oracle home to the destination Oracle home. Customized files may include:
   - Workflow process definition files (`.wft` files located in `<destination_MT_OH>/wf/res/<lang>`)
   - Business Event System definition files (`.wfx` files located in `<destination_MT_OH>/wf/res/<lang>`)

     Note: In 10*g* (9.0.4), Business Event System definition files have the extension `.wfx`, rather than `.xml`.

   - SQL scripts (`.sql` files located in `<destination_MT_OH>/wf/sql`)
   - Custom help files (`.htm` files located in `<destination_MT_OH>/wf/doc/<lang>/wfcust` or `.hlp` files located in `<destination_MT_OH>/wf/res/<lang>`)

12

Post-Upgrade Considerations

This chapter discusses the importance of a post-upgrade system test and considerations for the source Oracle home after the upgrade.

12.1 Starting the OracleAS Instance

Follow the steps below to start a middle-tier instance. If the middle-tier instance uses Infrastructure services, such as Identity Management or a Metadata Repository, make sure they are started.

1. Start the components with this command:

   ```
   <destination_MT_OH>/opmn/bin/opmnctl startall
   ```

   OPMN and all OPMN-managed processes such as DCM, Oracle HTTP Server, OC4J instances, Web Cache, Forms, and Reports, are started.

2. Start the Application Server Control Console with this command:

   ```
   <destination_MT_OH>/bin/emctl start iasconsole
   ```

12.2 Verify Operation of Middle Tier Components

Follow these steps to verify that the middle tier components that were upgraded are started:

1. In a browser, access the Oracle Enterprise Manager Application Server Control in the 10*g* (9.0.4) middle tier Oracle home by entering its URL. Ensure that you provide the correct host name and port number. For example:

   ```
   http://midtierhost.mycompany.com:1812
   ```

   The Oracle Enterprise Manager page appears. A link for the middle tier instance appears in the **Standalone Instances** section.

2. Click the link.

   The **System Components** page appears.

3. Verify that the components are running.

4. Verify that the configuration information for the components in use is reflected in the 10*g* (9.0.4) Oracle home.

12.3 Check Significant URLs

Follow these steps to verify that you can access the Oracle HTTP Server and application URLs:

1. Verify that you can access the Oracle HTTP Server on the same host and port that you did in the previous release by entering the URL. Ensure that you provide the correct host name and port number. For example:

   ```
   http://midtierhost.mycompany.com:7777
   ```

2. Verify that you can access the URLs for the applications you operated in the previous release, and ensure that the applications are functioning as they did in the previous release.

12.4 Performing a System Test

After the upgrade, you should perform tests for all applications in the upgraded instance to validate the correct functioning and interdependence of all components in the middle tier. If you use Infrastructure services, perform tests to ensure that they are working as expected.

12.5 Decommissioning the Source Oracle Home

When you are certain that the upgrade was successful, you have all of the necessary backups, and have no plans to revert to the source Oracle home, you may elect to remove the files from the source Oracle home. Use the Oracle Universal Installer to deinstall the instance.

> **See Also:** *Oracle9i Application Server Installation Guide* in the Release 1 (1.0.2.2) documentation library

12.5.1 Preserving Application Files and Log Files

If there are application files or log files in the source Oracle home that are being referenced or used by the destination Oracle home, you should move them to another location before you decommission the source Oracle home, and, in the destination Oracle home, change any references to the files to the new location.

Index

Symbols

Numerics

A

B

C

D

E

F

G

H

I

J

L

M

O

P

R

S

T

U

V

W

X

Z