

Oracle® Application Server 10g

Concepts

10g (9.0.4)

Part No. B10375-01

September 2003

Oracle Application Server 10g Concepts, 10g (9.0.4)

Part No. B10375-01

Copyright © 2002, 2003 Oracle Corporation. All rights reserved.

Primary Author: Theresa Robertson

Contributor: Thomas Kurian, Pavana Jain, Juliana Button, Steve Button, Mukul Paithane, Ashesh Parekh, Rakesh Dhoopar, Blaise Ribet, Christina Kolotouros, Regis Louis, John Lang, Milton Wan, Bart van der Hoeven, Chris Bussler, Ken Morse, David Saslav, Charlie Berger, Michael Mesaros, Richard Soule, Paul Narth, Phillip Weckerle, John Heimann, Doug Clarke, Jacques-Anton Dube, Christine Chan, Kalle Radage, Debbie Steiner, Uppili Srinivasan, Todd Vender, Robert Giljum, Naresh Kumar, Ric Goell, Valerie Kane, Preeti Somal, Deborah Steiner, Mason Ng, Wei Hu, Kai Li, Lee Cooper, Jesse Anton, Valerie Moore

The Programs (which include both the software and documentation) contain proprietary information of Oracle Corporation; they are provided under a license agreement containing restrictions on use and disclosure and are also protected by copyright, patent and other intellectual and industrial property laws. Reverse engineering, disassembly or decompilation of the Programs, except to the extent required to obtain interoperability with other independently created software or as specified by law, is prohibited.

The information contained in this document is subject to change without notice. If you find any problems in the documentation, please report them to us in writing. Oracle Corporation does not warrant that this document is error-free. Except as may be expressly permitted in your license agreement for these Programs, no part of these Programs may be reproduced or transmitted in any form or by any means, electronic or mechanical, for any purpose, without the express written permission of Oracle Corporation.

If the Programs are delivered to the U.S. Government or anyone licensing or using the programs on behalf of the U.S. Government, the following notice is applicable:

Restricted Rights Notice Programs delivered subject to the DOD FAR Supplement are "commercial computer software" and use, duplication, and disclosure of the Programs, including documentation, shall be subject to the licensing restrictions set forth in the applicable Oracle license agreement. Otherwise, Programs delivered subject to the Federal Acquisition Regulations are "restricted computer software" and use, duplication, and disclosure of the Programs shall be subject to the restrictions in FAR 52.227-19, Commercial Computer Software - Restricted Rights (June, 1987). Oracle Corporation, 500 Oracle Parkway, Redwood City, CA 94065.

The Programs are not intended for use in any nuclear, aviation, mass transit, medical, or other inherently dangerous applications. It shall be the licensee's responsibility to take all appropriate fail-safe, backup, redundancy, and other measures to ensure the safe use of such applications if the Programs are used for such purposes, and Oracle Corporation disclaims liability for any damages caused by such use of the Programs.

Oracle is a registered trademark, and Oracle*MetaLink*, Oracle Store, Oracle*9i*, Oracle Discoverer, SQL*Plus, SQL*Net, and PL/SQL are trademarks or registered trademarks of Oracle Corporation. Other names may be trademarks of their respective owners.

Contents

Send Us Your Comments	xvii
Preface.....	xix
Intended Audience	xix
Documentation Accessibility	xix
Organization.....	xx
Related Documentation	xxi
Conventions.....	xxii
1 Introduction to Oracle Application Server	
The Challenges of Creating and Maintaining an E-Business	1-1
Development Challenges	1-1
J2EE Applications.....	1-2
Web Services	1-3
Portals.....	1-3
Wireless.....	1-3
Personalization	1-3
Enterprise Integration.....	1-3

Deployment Challenges.....	1-4
Availability	1-4
Scalability.....	1-4
Performance.....	1-5
Caching	1-5
Systems Management	1-5
Security and Identity Management	1-5
Building an E-Business Solution with the Oracle Platform.....	1-5
Overview of Oracle Application Server.....	1-6
Oracle Application Server Solutions.....	1-6
J2EE and Internet Applications	1-7
Portals.....	1-7
Wireless.....	1-8
Business Intelligence	1-8
E-Businesses Integration.....	1-8
Availability and Scalability	1-8
Caching	1-8
Systems Management	1-9
Identity Management.....	1-9
Oracle Application Server Components.....	1-9

Part I Development

2 J2EE, Web Services, and Internet Applications

Application Development and Deployment in Oracle Application Server.....	2-1
What Types of Applications Can Be Developed for Deployment in Oracle Application Server?	2-2
Oracle HTTP Server.....	2-3
Oracle HTTP Server Components.....	2-4
Oracle HTTP Server Features.....	2-5
Oracle HTTP Server Architecture.....	2-6
Modular Architecture	2-7
Common Gateway Interface (CGI) Support.....	2-8

Oracle Application Server Containers for J2EE	2-9
Introduction to J2EE Application Development.....	2-9
What Is a J2EE Application?.....	2-10
J2EE Distributed Multi-tiered Application Model.....	2-10
Types of J2EE Clients.....	2-12
Types of J2EE Application Components.....	2-12
Types of J2EE Containers.....	2-12
J2EE Application Packaging Concepts.....	2-13
Oracle Application Server Containers for J2EE Features.....	2-13
Oracle Application Server Containers for J2EE Containers.....	2-13
J2EE Services.....	2-16
Oracle J2EE Services.....	2-18
Oracle Application Server Containers for J2EE Architecture.....	2-19
Oracle Application Server TopLink	2-21
Advantages of OracleAS TopLink.....	2-21
The OracleAS TopLink Problem Space.....	2-21
The OracleAS TopLink Solution.....	2-22
OracleAS TopLink Components.....	2-23
OracleAS TopLink Development Components.....	2-24
Oracle Application Server TopLink Mapping Workbench.....	2-24
Oracle Application Server TopLink Foundation Library.....	2-26
OracleAS TopLink Metadata.....	2-28
Application Development with OracleAS TopLink.....	2-31
Mapping.....	2-31
Session Management.....	2-32
Querying.....	2-32
Transactions.....	2-33
Packaging and Deployment.....	2-33
Monitoring and Performance Tuning.....	2-33
OracleAS TopLink Architectures Overview.....	2-34
Three-Tier.....	2-34
EJB Session Bean Facade.....	2-34
EJB Entity Beans with CMP.....	2-35
EJB Entity Beans with BMP.....	2-35
Two-Tier.....	2-35

Oracle JDeveloper	2-35
Oracle JDeveloper Features.....	2-36
Oracle Business Components for Java	2-37
Oracle Business Components for Java Features.....	2-38
Object-Relational Mapping	2-38
Validation Model.....	2-38
Scalability and Performance	2-38
Flexible Deployment	2-38
Rich User Interface Support.....	2-39
Layered Customization Support.....	2-39
Tight Integration with JDeveloper	2-39
Integration with Oracle <i>interMedia</i>	2-40
Oracle Application Server Web Services	2-40
Oracle Application Server Web Services Features.....	2-41
Oracle Application Server Web Services Development Features.....	2-41
Oracle Application Server Web Services Deployment and Management Features ..	2-42
Oracle Application Server Web Services Architecture.....	2-42
Web Services Framework.....	2-44
Oracle Application Server Forms Services	2-45
Oracle Application Server Forms Services Features.....	2-46
Automatic Internet Scalability and High Performance.....	2-46
Open Access to Internet Standards.....	2-48
Oracle Application Server Forms Services Architecture.....	2-48
Oracle XML Developer Kit	2-49
Oracle XML Developer Kit Tools	2-50
XML and XSLT Parsers.....	2-50
XML Schema Processors.....	2-51
XML Class Generators	2-51
XSQL Servlet	2-51
XML Transviewer Beans.....	2-52

Oracle Application Server PL/SQL Platform	2-52
Oracle Application Server PL/SQL Tools	2-52
mod_plsql	2-52
Oracle PL/SQL Server Pages.....	2-52
Oracle PL/SQL Web Toolkit.....	2-53
Oracle Application Server PL/SQL Architecture	2-53
Oracle Content Management Software Developer Kit	2-54
Oracle Content Management SDK Features.....	2-54
Oracle Content Management SDK Architecture.....	2-54
Oracle Application Server MapViewer	2-55
Oracle Application Server MapViewer Features	2-56
Oracle Application Server MapViewer Architecture	2-56

3 Portal Applications

Introduction to Oracle Application Server Portal	3-1
What is Oracle Application Server Portal?	3-1
E-Business Support with Oracle Application Server Portal.....	3-2
Oracle Application Server Portal Features.....	3-3
Portal Page Creation, Management, and Customization	3-4
Portal Content Publishing and Management	3-4
Content Searching	3-5
Content Syndication	3-6
Portals and Wireless Devices.....	3-7
Portal Integration with Oracle Application Server Single Sign-On	3-7
Application Access and Integration.....	3-7
Integrating with Portlet Providers	3-8
Oracle Application Server Portlets	3-8
Partner Portlets	3-9
Custom Portlets	3-9
Oracle Application Server Portal Integration with Oracle Application Server Web Cache	3-10
Oracle Application Server Web Cache Deployment with Oracle Application Server Portal	3-10
Oracle Application Server Portal Architecture	3-11

4 Wireless Applications

Introduction to Oracle Application Server Wireless	4-1
Oracle Application Server Wireless Overview.....	4-1
Oracle Application Server Wireless Features	4-2
Multi-Channel Server	4-3
J2ME Support	4-4
Notifications and Multimedia Messaging.....	4-4
Wireless Development Kit.....	4-5
Web Clipping	4-6
Location Services	4-6
Mobile Office Applications	4-7
Oracle Application Server Wireless Architecture	4-8
Oracle Application Server Wireless Adapters.....	4-9
Oracle Application Server Wireless Transformers	4-9

5 Business Intelligence

Introduction to Oracle Application Server Business Intelligence	5-1
Oracle Application Server Reports Services	5-4
Introduction to Oracle Application Server Reports Services	5-4
Oracle Application Server Reports Services Features	5-6
Event-Based Reporting	5-6
Extensible Architecture.....	5-6
Enhanced Report Bursting and Distribution	5-7
Oracle Application Server Reports Services Architecture	5-8
Oracle Application Server Discoverer	5-9
Introduction to Oracle Application Server Discoverer	5-9
Oracle Discoverer Tools.....	5-11
Oracle Application Server Discoverer Viewer	5-11
Oracle Application Server Discoverer Plus	5-11
Oracle Application Server Discoverer Portlet Provider.....	5-12
Oracle Discoverer Desktop.....	5-12
Oracle Discoverer Administrator	5-12
Oracle Application Server Discoverer Architecture	5-13

Oracle Application Server Personalization	5-15
Introduction to Oracle Application Server Personalization.....	5-15
Oracle Application Server Personalization Features.....	5-16
Real-Time Recommendation Engine Architecture.....	5-16
Data Mining Technology.....	5-16
Single Administrative Interface.....	5-16
Oracle Application Server Personalization Architecture.....	5-17
Integrating Oracle Application Server Business Intelligence	5-18
Leveraging Single Sign-on Functionality.....	5-19
Leveraging Oracle Application Server Portal Functionality.....	5-20
Creating an E-Business Dashboard Using Oracle Application Server Portal.....	5-20
Using Oracle Enterprise Manager Application Server Control for Management	5-21
Implementing Enterprise Reporting with Oracle Reports Builder	5-21
Improving Performance with Oracle Application Server Web Cache	5-22

6 Oracle Application Server Integration

Introduction to Oracle Application Server Integration.....	6-1
What is Oracle Application Server ProcessConnect?	6-3
Oracle Application Server ProcessConnect Features.....	6-5
Visual Design Toolset	6-7
Design-Time and Runtime Repositories	6-8
Runtime Integration Manager	6-9
Adapters and Adapter Framework.....	6-10
Adapters	6-10
Adapter Framework.....	6-11
Data Management	6-12
Business Process and Business Activity Monitoring.....	6-13
Systems Monitoring and Management	6-14
Business to Business (B2B) Connectivity and Prepackaged Integrations	6-15
Oracle Application Server ProcessConnect Architecture	6-16
Oracle Application Server ProcessConnect Summary	6-17

7 Oracle Application Server Infrastructure

Introduction to Oracle Application Server Infrastructure	11-1
What is Oracle Application Server Infrastructure?.....	11-2
Oracle Application Server Infrastructure Components	11-2
Oracle Application Server Metadata Repository	11-3
Using Oracle Application Server Infrastructure with Middle Tier Installations	11-4
Oracle Application Server Metadata Repository Contents	11-5
Oracle Identity Management	11-5
Oracle Application Server Single Sign-On.....	11-6
Oracle Internet Directory.....	11-6
Oracle Application Server Certificate Authority	11-7
Oracle Application Server Infrastructure Architecture	11-7

Part II Deployment

8 Scalability, High Availability, and Recovery

Scalability	7-1
Web Cache Clusters.....	7-1
Oracle Application Server Containers for J2EE (OC4J) Routing	7-2
Choosing an Intelligent Routing Algorithm.....	7-3
Enterprise JavaBean (EJB) Client Routing	7-4
High Availability	7-4
High Availability for Planned Maintenance	7-5
Rolling Upgrade.....	7-5
Cloning	7-5
Application High Availability	7-5
Session Replication	7-5
Session Persistence	7-6
Distributed Cache	7-6

Oracle Application Server Middle Tier High Availability	7-7
Oracle Application Server Single Instance Availability Features.....	7-7
Oracle Application Server Clusters.....	7-8
Types of Oracle Application Server Clusters.....	7-8
Benefits of Oracle Application Server Clustering.....	7-9
Improving Availability with an External Load Balancer.....	7-10
Types of External Load Balancers.....	7-10
Benefits of External Load Balancing.....	7-11
Improving Availability with Operating System Clusters.....	7-11
Oracle Application Server Infrastructure High Availability	7-11
High Availability Requirements for Oracle Application Server Infrastructure.....	7-11
Oracle Application Server Cold Failover Clusters	7-12
Oracle Application Server Middle Tiers in a Cold Failover Cluster Environment ...	7-13
Oracle Application Server Active Failover Clusters.....	7-13
Installing the Metadata Repository into an Existing RAC Database.....	7-14
Recovery Solutions	7-14
Restarting Processes.....	7-14
Cold Backup and Restore	7-15
Online Backup and Restore.....	7-15
Disaster Recovery	7-16
Distributed Configuration Management Archiving Feature.....	7-17

9 Performance and Caching

Introduction to Performance	8-1
Performance Methodology.....	8-2
Performance Targets.....	8-2
User Expectations.....	8-3
Performance Evaluation	8-3
Improving Performance	8-3
Factors in Improving Performance.....	8-3
Countering the Effects of Excessive Demand	8-4
Making Adjustments to Relieve Performance Problems.....	8-4
Overview of Caching Solutions	8-4
Introduction to Server Accelerators.....	8-5
Introduction to Oracle Application Server Web Cache	8-5

Oracle Application Server Web Cache Features	8-6
Compression and Caching	8-6
Automatic Content Compression.....	8-6
Full Page Static and Dynamic Content Caching	8-7
Cache Invalidation and Expiration	8-7
Partial-Page Caching and Personalized Page Assembly	8-7
Workload Management	8-8
Web Server Load Balancing and Failover	8-8
Performance Assurance	8-8
Cache Clustering.....	8-8
End-User Experience Management.....	8-8
End-User Performance Monitoring	8-9
New for Oracle Application Server Web Cache 10g (9.0.4)	8-9
Support for Client-side SSL Certificates.....	8-9
Flexible Multi-version Caching Rules	8-9
Integration with Oracle Process Manager and Notification Server (OPMN).....	8-9
Inline Invalidation and Search Key Invalidation	8-10
Oracle Application Server Web Cache Deployment Architecture	8-10
Additional Caching Components	8-12
Java Object Cache.....	8-12
Web Object Cache.....	8-12

10 System Management

Introduction to System Management	9-1
Introduction to Oracle Enterprise Manager.....	9-1
Oracle Enterprise Manager Application Server Control Features	9-2
Complete Oracle Application Server Administration.....	9-2
Monitoring Oracle Application Server	9-3
Oracle Enterprise Manager Application Server Control Architecture	9-3
Oracle Enterprise Manager Application Server Control Underlying Technologies.....	9-4

11 Security and Identity Management

Introduction to Security.....	10-1
Introduction to Identity Management.....	10-1

Security Components and Features	10-2
Oracle Identity Management	10-3
Oracle Application Server Single Sign-On	10-4
Oracle Internet Directory	10-6
Oracle Application Server Certificate Authority	10-8
Java Authentication and Authorization Service (JAAS)	10-8
OracleAS Web Cache Security	10-8
Restricted Administration	10-9
Secure Sockets Layer (SSL) Support	10-9
Oracle HTTP Server Security	10-10
Session Renegotiation Support	10-10
SSL Hardware Acceleration Support	10-10
Port Tunnelling	10-11
OHS to OC4J SSL Support	10-11
Portal Security	10-11
User Authentication in OracleAS Portal	10-11
Access Control in OracleAS Portal	10-11
Security Architecture	10-12

12 Recommended Topologies

Introduction to Oracle Application Server Recommended Topologies	11-1
General Development Topologies.....	11-2
General Deployment Topologies	11-2
Special Topologies.....	11-2
Key Considerations	11-2
Installation.....	11-2
Application Deployment and Performance	11-3
Security	11-3
Management.....	11-3
High Availability Deployment.....	11-3
Third Party Products.....	11-3

Recommended General Development Topologies	11-3
Java Developer Topology: Low-end Machines with Java IDE Tools.....	11-4
Portal and Wireless Developer Topology: Medium-sized Machines	11-5
Forms, Reports, and Discoverer Developer Topology: Medium-sized Machines	11-7
Integration Architect and Process Modeler Topology: Medium-sized Machines.....	11-9
Recommended General Deployment Topologies	11-11
Enterprise Data Center Topology: Multiple Departments Sharing the Same Data Center.....	1-12
Enterprise Data Center Topology for Java Applications	11-14
Enterprise Data Center Topology for Portal, Wireless, Business Intelligence, and Forms Applications	11-16
Departmental Topology: Departments Hosting Their Applications	11-18
Development Life Cycle Support Topology: Test to Stage to Production Support.....	11-21
Special Topologies	11-23
High Availability Deployments.....	11-23
Cold Failover Cluster	11-23
Active Failover Clusters	11-25

Glossary

Index

List of Figures

1-1	Application Development Challenges: Application Requirements.....	1-2
1-2	Application Deployment Challenges: System Requirements.....	1-4
1-3	Oracle Application Server Solutions.....	1-7
2-1	Oracle HTTP Server Request Flow	2-5
2-2	Oracle HTTP Server Process Architecture	2-7
2-3	Oracle HTTP Server HTTP Request-Response Cycle.....	2-8
2-4	J2EE Distributed Multi-tiered Application Architecture.....	2-11
2-5	Oracle Application Server Containers for J2EE Architecture	2-20
2-6	TopLink Components in the Development Cycle.....	2-23
2-7	The OracleAS TopLink Mapping Workbench in a TopLink Environment	2-25
2-8	OracleAS TopLink Application Components	2-26
2-9	OracleAS TopLink Metadata	2-29
2-10	Oracle Application Server Web Services Architecture	2-44
2-11	Oracle Application Server Forms Services Request Flow	2-49
2-12	Oracle Content Management SDK Architecture.....	2-55
2-13	MapViewer Architecture.....	2-57
3-1	Sample Portal Page.....	3-2
3-2	OracleAS Portal Request Flow.....	3-12
4-1	Oracle Application Server Wireless.....	4-2
4-2	Oracle Application Server Wireless Overview	4-3
4-3	Oracle Application Server Wireless Architecture.....	4-9
4-4	Oracle Application Server Wireless - Adapters and Transformers.....	4-10
5-1	Oracle Application Server Business Intelligence Components.....	5-3
5-2	Oracle Reports Example	5-5
5-3	Oracle Application Server Reports Services Architecture.....	5-8
5-4	Oracle Application Server Discoverer Worksheet Example	5-10
5-5	Oracle Application Server Discoverer Architecture.....	5-13
5-6	Oracle Application Server Personalization Architecture.....	5-17
6-1	Oracle Application Server ProcessConnect Components	6-6
6-2	Adapter Framework, Adapters, and Integration Manager (Runtime).....	6-11
7-1	Oracle Application Server Infrastructure Components.....	11-8
9-1	Caching Architecture	8-11
11-1	Oracle Application Server Security Architecture	10-13
12-1	Java Developer Topology	11-5
12-2	Portal and Wireless Developer Topology	11-7
12-3	Forms, Reports, and Discoverer Topology	11-9
12-4	Integration Architect and Process Modeler Topology	11-11
12-5	Enterprise Data Center Topology for Java Applications.....	11-15
12-6	Enterprise Data Center Topology for Portal, Wireless, Business Intelligence, and Forms Applications	11-17

12-7	Departmental Topology	11-21
12-8	Cold Failover Cluster	11-24
12-9	Active Failover Clusters	11-26

List of Tables

1-1	Oracle Application Server Components	1-10
2-1	Supported Technologies and Programming Languages	2-3
2-2	Prominent Oracle HTTP Server Modules	2-4
2-3	Oracle Application Server Containers for J2EE Supported APIs	2-9
6-1	Integration Manager Coordination.....	6-10
7-1	Metadata and Infrastructure Components	11-4
8-1	Intelligent Routing Algorithms Matrix	7-2
12-1	Key Considerations for Java Developers	11-4
12-2	Key Considerations for Portal and Wireless Developers	11-6
12-3	Key Considerations for Forms and Reports Developers	11-8
12-4	Key Considerations for Integration Architects and Process Modelers.....	11-10
12-5	Key Considerations for the Enterprise Data Center Topology	11-12
12-6	Key Considerations for the Departmental Topology	11-18
12-7	Key Considerations for the Development Life Cycle Support Topology	11-22

Send Us Your Comments

Oracle Application Server 10g Concepts, 10g (9.0.4)

Part No. B10375-01

Oracle Corporation 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: 650-506-7375 Attn: Oracle Application Server Documentation Manager
- Postal service:
Oracle Corporation
Oracle Application Server Documentation
500 Oracle Parkway, M/S 10p6
Redwood Shores, CA 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 contains these topics:

- Intended Audience
- Documentation Accessibility
- Organization
- Related Documentation
- Conventions

Intended Audience

Oracle Application Server 10g Concepts is intended for anyone with an interest in Oracle Application Server or the latest Internet technologies.

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

Organization

This document contains:

Chapter 1, "Introduction to Oracle Application Server"

This chapter provides an overview of application servers and Oracle Application Server.

Part I, "Development"

Chapters in this part discuss application development topics.

Chapter 2, "J2EE, Web Services, and Internet Applications"

This chapter describes the Java 2 Platform, Enterprise Edition (J2EE) and Internet application development and deployment environment in Oracle Application Server.

Chapter 3, "Portal Applications"

This chapter provides an overview of Oracle Application Server Portal features and benefits.

Chapter 4, "Wireless Applications"

This chapter provides an overview of Oracle Application Server Wireless features and benefits.

Chapter 5, "Business Intelligence"

This chapter provides an overview of the Oracle Application Server Business Intelligence features and benefits.

Chapter 6, "Oracle Application Server Integration"

This chapter describes Oracle Application Server integration features and benefits.

Chapter 7, "Oracle Application Server Infrastructure"

This chapter provides an overview of Oracle Application Server Infrastructure features and benefits.

Part II, "Deployment"

Chapters in this part discuss deployment topics.

Chapter 8, "Scalability, High Availability, and Recovery"

This chapter provides an overview of Oracle Application Server high availability and scalability features and benefits.

Chapter 9, "Performance and Caching"

This chapter provides an overview of Oracle Application Server caching features and benefits.

Chapter 10, "System Management"

This chapter provides an overview of Oracle Application Server system management features and benefits.

Chapter 11, "Security and Identity Management"

This chapter provides an overview of Oracle Application Server security features and benefits.

Chapter 12, "Recommended Topologies"

This chapter provides an overview of Oracle Application Server recommended development and deployment configurations.

Glossary

The glossary defines terminology used throughout this guide and the Oracle Application Server documentation set.

Related Documentation

For more information, see these Oracle resources:

- Oracle Application Server Documentation Library
- Oracle Application Server Platform-Specific Documentation on Oracle Application Server Disk 1

Printed documentation is available for sale in the Oracle Store at

<http://oraclestore.oracle.com/>

To download free release notes, installation documentation, white papers, or other collateral, please visit the Oracle Technology Network (OTN). You must register online before using OTN; registration is free and can be done at

<http://otn.oracle.com/membership/>

If you already have a username and password for OTN, then you can go directly to the documentation section of the OTN Web site at

<http://otn.oracle.com/documentation/>

Conventions

This section describes the conventions used in the text and code examples of this documentation set. It describes:

- Conventions in Text
- Conventions in Code Examples
- Conventions for Microsoft Windows Operating Systems

Conventions in Text

We use various conventions in text to help you more quickly identify special terms. The following table describes those conventions and provides examples of their use.

Convention	Meaning	Example
Bold	Bold typeface indicates terms that are defined in the text or terms that appear in a glossary, or both.	When you specify this clause, you create an index-organized table .
<i>Italics</i>	Italic typeface indicates book titles or emphasis.	<i>Oracle Application Server 10g Concepts</i> Ensure that the recovery catalog and target database do <i>not</i> reside on the same disk.

Convention	Meaning	Example
UPPERCASE monospace (fixed-width) font	Uppercase monospace typeface indicates elements supplied by the system. Such elements include parameters, privileges, datatypes, RMAN keywords, SQL keywords, SQL*Plus or utility commands, packages and methods, as well as system-supplied column names, database objects and structures, usernames, and roles.	You can specify this clause only for a NUMBER column. You can back up the database by using the BACKUP command. Query the TABLE_NAME column in the USER_TABLES data dictionary view. Use the DBMS_STATS.GENERATE_STATS procedure.
lowercase monospace (fixed-width) font	Lowercase monospace typeface indicates executables, filenames, directory names, and sample user-supplied elements. Such elements include computer and database names, net service names, and connect identifiers, as well as user-supplied database objects and structures, column names, packages and classes, usernames and roles, program units, and parameter values. Note: Some programmatic elements use a mixture of UPPERCASE and lowercase. Enter these elements as shown.	Enter sqlplus to open SQL*Plus. The password is specified in the orapwd file. Back up the datafiles and control files in the /disk1/oracle/dbs directory. The department_id, department_name, and location_id columns are in the hr.departments table. Set the QUERY_REWRITE_ENABLED initialization parameter to true. Connect as oe user. The JRepUtil class implements these methods.
lowercase italic monospace (fixed-width) font	Lowercase italic monospace font represents placeholders or variables.	You can specify the <i>parallel_clause</i> . Run <i>Uold_release</i> .SQL where <i>old_release</i> refers to the release you installed prior to upgrading.

Conventions in Code Examples

Code examples illustrate SQL, PL/SQL, SQL*Plus, or other command-line statements. They are displayed in a monospace (fixed-width) font and separated from normal text as shown in this example:

```
SELECT username FROM dba_users WHERE username = 'MIGRATE';
```

The following table describes typographic conventions used in code examples and provides examples of their use.

Convention	Meaning	Example
[]	Brackets enclose one or more optional items. Do not enter the brackets.	DECIMAL (<i>digits</i> [, <i>precision</i>])
{ }	Braces enclose two or more items, one of which is required. Do not enter the braces.	{ENABLE DISABLE}
	A vertical bar represents a choice of two or more options within brackets or braces. Enter one of the options. Do not enter the vertical bar.	{ENABLE DISABLE} [COMPRESS NOCOMPRESS]
...	Horizontal ellipsis points indicate either: <ul style="list-style-type: none"> ▪ That we have omitted parts of the code that are not directly related to the example ▪ That you can repeat a portion of the code 	CREATE TABLE ... AS <i>subquery</i> ; SELECT <i>col1</i> , <i>col2</i> , ... , <i>coln</i> FROM employees;
.	Vertical ellipsis points indicate that we have omitted several lines of code not directly related to the example.	
Other notation	You must enter symbols other than brackets, braces, vertical bars, and ellipsis points as shown.	acctbal NUMBER(11,2); acct CONSTANT NUMBER(4) := 3;
<i>Italics</i>	Italicized text indicates placeholders or variables for which you must supply particular values.	CONNECT SYSTEM/ <i>system_password</i> DB_NAME = <i>database_name</i>
UPPERCASE	Uppercase typeface indicates elements supplied by the system. We show these terms in uppercase in order to distinguish them from terms you define. Unless terms appear in brackets, enter them in the order and with the spelling shown. However, because these terms are not case sensitive, you can enter them in lowercase.	SELECT last_name, employee_id FROM employees; SELECT * FROM USER_TABLES; DROP TABLE hr.employees;

Convention	Meaning	Example
lowercase	<p>Lowercase typeface indicates programmatic elements that you supply. For example, lowercase indicates names of tables, columns, or files.</p> <p>Note: Some programmatic elements use a mixture of UPPERCASE and lowercase. Enter these elements as shown.</p>	<pre>SELECT last_name, employee_id FROM employees; sqlplus hr/hr CREATE USER mjones IDENTIFIED BY ty3MU9;</pre>

Conventions for Microsoft Windows Operating Systems

The following table describes conventions for Microsoft Windows operating systems and provides examples of their use.

Convention	Meaning	Example
Choose Start >	How to start a program.	To start the Oracle Database Configuration Assistant, choose Start > Programs > Oracle - <i>HOME_NAME</i> > Configuration and Migration Tools > Database Configuration Assistant.
File and directory names	File and directory names are not case sensitive. The following special characters are not allowed: left angle bracket (<), right angle bracket (>), colon (:), double quotation marks ("), slash (/), pipe (), and dash (-). The special character backslash (\) is treated as an element separator, even when it appears in quotes. If the file name begins with \\, then Windows assumes it uses the Universal Naming Convention.	<pre>c:\winnt\"\"system32 is the same as C:\WINNT\SYSTEM32</pre>
C:\>	Represents the Windows command prompt of the current hard disk drive. The escape character in a command prompt is the caret (^). Your prompt reflects the subdirectory in which you are working. Referred to as the <i>command prompt</i> in this manual.	C:\oracle\oradata>

Convention	Meaning	Example
	<p>The backslash (\) special character is sometimes required as an escape character for the double quotation mark (") special character at the Windows command prompt. Parentheses and the single quotation mark (') do not require an escape character. Refer to your Windows operating system documentation for more information on escape and special characters.</p>	<pre>C:\>exp scott/tiger TABLES=emp QUERY=\ "WHERE job='SALESMAN' and sal<1600\" C:\>imp SYSTEM/password FROMUSER=scott TABLES=(emp, dept)</pre>
<i>HOME_NAME</i>	<p>Represents the Oracle home name. The home name can be up to 16 alphanumeric characters. The only special character allowed in the home name is the underscore.</p>	<pre>C:\> net start OracleHOME_ NAMEtnsListener</pre>

Convention	Meaning	Example
<i>ORACLE_HOME</i> and <i>ORACLE_BASE</i>	<p>In releases prior to Oracle8i release 8.1.3, when you installed Oracle components, all subdirectories were located under a top level <i>ORACLE_HOME</i> directory that by default used one of the following names:</p> <ul style="list-style-type: none"> ■ C:\orant for Windows NT ■ C:\orawin95 for Windows 95 ■ C:\orawin98 for Windows 98 <p>This release complies with Optimal Flexible Architecture (OFA) guidelines. All subdirectories are not under a top level <i>ORACLE_HOME</i> directory. There is a top level directory called <i>ORACLE_BASE</i> that by default is C:\oracle. If you install Oracle9i release 1 (9.0.1) on a computer with no other Oracle software installed, then the default setting for the first Oracle home directory is C:\oracle\ora90. The Oracle home directory is located directly under <i>ORACLE_BASE</i>.</p> <p>All directory path examples in this guide follow OFA conventions.</p> <p>Refer to <i>Oracle9i Database Getting Starting for Windows</i> for additional information about OFA compliances and for information about installing Oracle products in non-OFA compliant directories.</p>	Go to the <i>ORACLE_BASE\ORACLE_HOME\rdms\admin</i> directory.

Introduction to Oracle Application Server

This chapter provides an overview of Oracle Application Server. The topics include:

- The Challenges of Creating and Maintaining an E-Business
- Building an E-Business Solution with the Oracle Platform
- Overview of Oracle Application Server

The Challenges of Creating and Maintaining an E-Business

The Internet introduces significant opportunities for companies to reach new markets and to streamline their internal business processes. At the same time, these opportunities generate new challenges because applications must be delivered quickly and must serve vast numbers of users.

Quickly delivering highly scalable applications often requires integrating diverse products that were not designed to work together. When this is the only way to get a consolidated set of business applications, companies must spend large amounts of capital, time, and effort, both initially and continuously, to support the complicated technologies or systems that they have built.

Companies that create **e-business** Web sites experience common problems and challenges associated with developing and deploying applications.

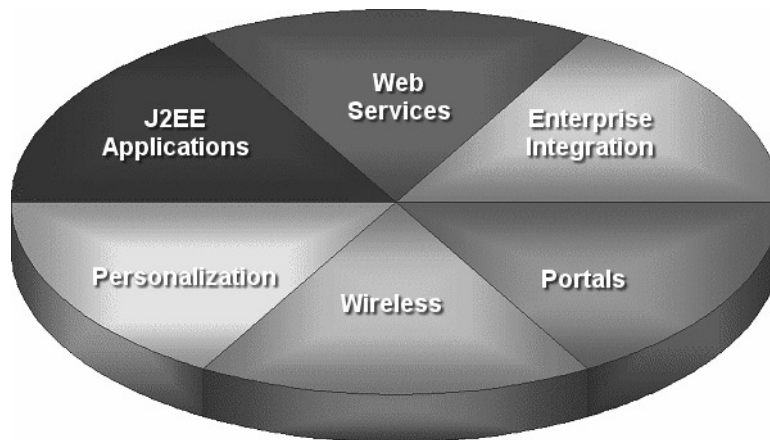
Development Challenges

There are numerous challenges associated with application development. Developing applications in a complex Internet environment involves dozens of interfaces, programming languages, and platforms. Assembling the infrastructure to support such applications requires comprehensive knowledge on the part of

Internet application developers. In addition, developing, testing, and deploying applications is very time consuming.

There is a common set of requirements that today's Internet applications need to meet. The challenge for e-businesses is to create applications that integrate all of these requirements. Figure 1-1 shows the six main requirements that contribute to the challenge of developing integrated Internet applications: Java2 Platform, Enterprise Edition (J2EE) requirements, Web services, enterprise integration, portals, wireless, and personalization.

Figure 1-1 Application Development Challenges: Application Requirements



J2EE Applications

Most e-business application developers face the fact that development cycles have become shorter, while demand for an increasing number of programming languages and multiple platforms keeps growing. In recent years, developers began using **Java**, a platform-independent programming language, in application development. However, their applications are still vendor-dependent because of proprietary Application Programming Interfaces (APIs). Complying with industry standards to create a level of cross-vendor standardization has become an increasing demand and challenge. The J2EE platform and component specifications define a standard platform for developing multi-tier, Web-based enterprise applications. Using J2EE as a development requirement addresses the issue of conflicting industry standards.

Web Services

As e-business evolves, companies rely increasingly on their Web sites to conduct business. Web services provide a standards-based infrastructure through which any business can both offer internal business process services and dynamically link its business processes with those of its partners. Web services support direct interactions with other software applications using XML-based messages and Internet-based products.

Portals

Over the past several years, the portal has emerged as the new Internet desktop, enabling users to access information through Web browsers and to combine information from different sources into a single entry point. Portals also support personalized views, so that users or user groups can customize the content and the appearance of the portal to suit individual preferences and requirements. Secure, easy-to-use, and dynamic portals that provide access to all applications and Web services are essential infrastructure elements for e-business.

Wireless

An increasing number of people conduct business out of the office. Wireless applications allow users rapid access through mobile devices, such as Web-enabled phones, personal digital assistants, or pagers. However, wireless applications are limited by small screen size, limited data entry capacity, and heterogeneity in standards supporting wireless devices.

Personalization

Customers, through Web site navigation, purchases, demographics, and ratings data, provide valuable information about themselves and their interests. When this information is extracted effectively, you can deliver tailored customer experiences and drive business decisions.

Enterprise Integration

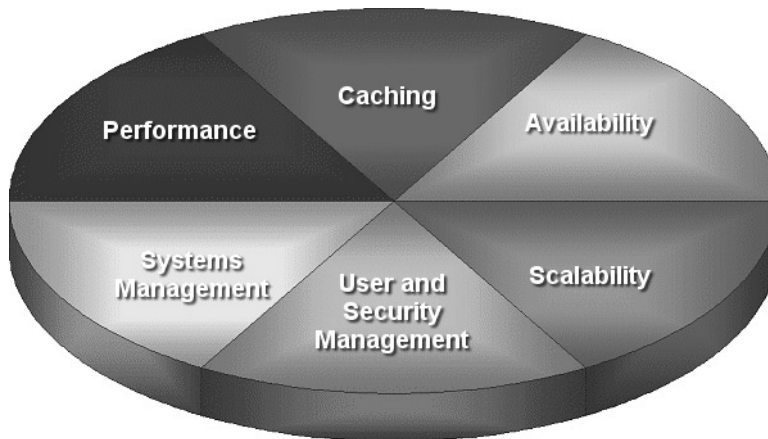
Most companies need to integrate their existing applications and data sources with new business processes. As companies grow, it is essential to integrate existing applications and data sources with those of their partners, customers, and suppliers. These companies must perform a complete enterprise **application integration** without rewriting the applications or building extensive customized program logic.

Deployment Challenges

The key to the success of any Web site is how quickly and reliably the **server** can deliver the appropriate content to users. If a server takes too long to respond or if it fails, then users will take their business elsewhere. If content and applications are not secure, then sensitive information or information technology assets are vulnerable. If deployed applications cannot use hardware efficiently, then information technology budgets are depleted quickly on equipment purchases.

Figure 1–2 shows the concerns associated with application deployment, which include performance, **caching**, **availability**, **scalability**, user and security management, and systems management.

Figure 1–2 Application Deployment Challenges: System Requirements



Availability

The **availability** of an overall system is measured by the percentage of time that it works normally. A successful e-business requires a 100% available operation without downtime. To improve availability, companies may use **redundant** components, but only when the back-up component can take over for a failed primary component immediately.

Scalability

The **scalability** of a system refers to how well it responds as user demands increase. The number of Web users has increased tremendously in the past few years. It is

crucial to maintain high quality Web sites that scale when you increase your hardware resources.

Performance

Increasing the performance of your Web site and increasing the speed of your applications without redesigning or rebuilding the Web site are common goals. Typical performance factors include the ability to handle multiple requests simultaneously, competition for resources, latency, response time, service time, throughput, wait time, and scalability.

Caching

Many high-volume Web sites serve thousands of users concurrently and need to provide them with accurate data in a dynamic environment where content changes frequently. Caching is one of the key technologies that promises to alleviate the computational and economic burdens resulting from complicated Web site infrastructure and technologies. However, it is often difficult to cache the dynamic, personalized content that the e-business model requires.

Systems Management

Administrators of application servers require sophisticated tools to monitor and manage the systems infrastructure, and the **privileges** and access rights of individual users. Maintaining a simplified process in managing your Web site allows you to manage the site infrastructure and monitor the performance from one integrated console.

Security and Identity Management

With the Internet continually growing, the threat to information traveling over the network increases exponentially. Maintaining a secure Web site is crucial. Once your Web site is running, you must protect the databases and the servers on which they reside. You must administer and protect the rights of internal database users, and you must guarantee the confidentiality of e-business customers as they access your database.

Building an E-Business Solution with the Oracle Platform

Oracle Application Server is a part of the Oracle platform, a complete and integrated e-business platform. The Oracle platform consists of:

- **Oracle Developer Suite** for developing applications

- **Oracle Application Server** for deploying Internet applications
- **Oracle Database Server** for storing content

To deliver scalable and high performance e-business solutions successfully, you must be able to leverage an integrated, comprehensive, flexible, and open platform. Oracle provides integrated development, deployment, and management tools, as well as a runtime platform. The Oracle platform simplifies the process of creating and deploying the applications that run your business on the Internet.

See Also: The product pages on the Oracle Technology Network at <http://otn.oracle.com> for more information about Oracle Database Server and Oracle Developer Suite

Overview of Oracle Application Server

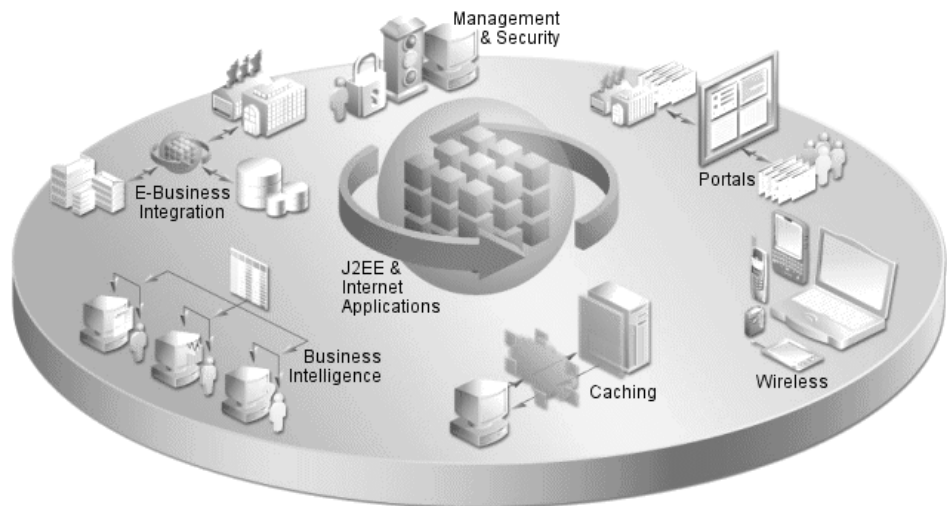
Oracle Application Server is a completely standards-based application server that provides a comprehensive and fully integrated platform for running Web sites, **J2EE** applications, and Web services. It addresses all the challenges that you face as you refine your business processes to become an e-business.

Oracle Application Server provides full support for the **J2EE** platform, **XML**, and emerging Web services, and grid standards. With Oracle Application Server you can simplify information access for your customers and trading partners by delivering enterprise **portals**, which can be customized and accessed from a network browser or from wireless devices. It allows you to redefine your business processes, and integrate your applications and data sources with those from your customers or partners. You can deliver tailored customer experiences via real-time personalization, and assess and correlate customer navigation, purchasing, ratings, and demographic data.

Oracle Application Server allows you to save on Web site infrastructure by deploying your fast, scalable Internet applications through built-in **Web caching**, load balancing and clustering capabilities.

Oracle Application Server Solutions

Figure 1-3 illustrates solutions that Oracle Application Server provides to address the development and deployment challenges common to e-business Web sites. These solutions include **J2EE** and Internet applications, portals, wireless, business intelligence, e-business integration, caching, management, and security, built on a scalable and highly available infrastructure.

Figure 1–3 Oracle Application Server Solutions

J2EE and Internet Applications

Oracle Application Server is built entirely on a J2EE framework that supports the latest industry standard technologies and programming languages, including J2EE API specifications, XML, and Web services. This comprehensive and flexible framework allows you to design, develop, and deploy dynamic Web sites, portals, and transactional applications using familiar programming languages and technologies such as servlets, JavaServer pages, XML, PL/SQL Server Pages, and SOAP. Oracle Application Server also provides comprehensive Web services to expose business functions to authorized parties over the Internet from any Web device.

Portals

Oracle Application Server provides an out-of-the-box portal that does not require endless programming and maintenance. You can use Oracle Application Server Portal to build, deploy, and maintain self-service and integrated enterprise portals. Oracle Application Server Portal allows for self-service content management and publishing, wizard-based development, and deploying, publishing, and consuming Web services on an extensible framework.

Wireless

Oracle Application Server Wireless simplifies wireless development and deployment by providing the ability to deliver content to any device, to use any protocol, and to work across any wireless network. In addition, Oracle Application Server Wireless includes wireless services, such as e-mail and location-based services, that simplify wireless-enabling applications and portals. Oracle Application Server provides application developers independence from the underlying wireless infrastructure. Oracle Application Server Wireless is built on Oracle Application Server Containers for J2EE (OC4J), leveraging open standards support in XML and J2EE to deliver a high-performance and scalable wireless infrastructure.

Business Intelligence

Oracle Application Server provides comprehensive personalization and **business intelligence** services. Using Oracle Application Server Business Intelligence features, you can dynamically serve personalized content recommendations to both registered and anonymous visitors as they browse your site; perform dynamic, ad-hoc query reporting and analysis using a standard Web browser; and publish high quality, dynamically generated reports on a scalable, secure platform.

E-Businesses Integration

Oracle Application Server has a powerful set of features that provide communications and integration capabilities for e-business applications. Using Oracle Application Server, you can integrate enterprise applications, trading partners, and Web services, emphasizing **scalability** and manageability, and provide seamless query and **transaction** access to many non-Oracle data sources.

Availability and Scalability

Oracle Application Server provides a flexible deployment model that allows you to architect your system for high availability and scalability. Oracle Application Server provides a variety of options for improving availability and scalability, and provides features for implementing fault tolerance, death detection, and failover. Additionally, Oracle Application Server supports such high availability options as cold failover clusters and active failover clusters.

Caching

Oracle Application Server provides a Web caching solution with the unique capability of caching both static and dynamically generated Web content. The **Oracle Application Server Web Cache** significantly improves the performance and

scalability of heavily loaded Web sites. In addition, the Web cache provides a number of features to ensure consistent and predictable responses. These features include page fragment caching, Edge Side Includes (ESI) and Edge Side Includes for Java (JESI) support, compression, dynamic content assembly, Web server load balancing, Web cache clustering, and **failover**.

Systems Management

Oracle Application Server provides a set of management facilities that are based on industry standards to simplify all aspects of Web site administration. Using Oracle Application Server, you can:

- Configure and monitor Oracle Application Server **instances** to optimize them for performance and scalability from a centralized console
- Respond to problem conditions from a centralized console
- Use encrypted secure sockets layer (SSL) connections, user and client certificate-based **authentication**, and **single sign-on** across all applications
- Implement Oracle Internet Directory, an **LDAP**-compliant directory that provides a single repository and administration environment for user accounts
- Perform transaction tracing and performance monitoring

Identity Management

The Oracle Application Server identity management infrastructure allows you to manage user identity throughout the application security life cycle. Oracle Application Server provides components for handling authentication, security services, authorization, and user provisioning to ensure the security of your Internet applications.

Oracle Application Server Components

Table 1–1 presents the Oracle Application Server components that are associated with these solutions.

Table 1–1 Oracle Application Server Components

Solution	Oracle Application Server Components
J2EE and Internet Applications	Oracle HTTP Server Oracle Application Server Containers for J2EE Oracle Application Server TopLink Oracle Business Components for Java Oracle Application Server Web Services Oracle JDeveloper Oracle Application Server Forms Services Oracle XML Developer Kit Oracle PL/SQL Oracle Content Management SDK Oracle Application Server MapViewer
Portals	Oracle Application Server Portal Oracle Application Server Portal Developer Kit Oracle Ultra Search Oracle Application Server Syndication Services
Wireless	Oracle Application Server Wireless Oracle Application Server Wireless Developer Kit
Business Intelligence	Oracle Application Server Reports Services Oracle Application Server Discoverer Oracle Application Server Personalization
E-Business Integration	Oracle Application Server InterConnect Oracle Application Server ProcessConnect
Caching	Oracle Application Server Web Cache
System Management	Oracle Enterprise Manager
Identity Management and Security	Oracle Application Server Single Sign-On Oracle Application Server Certificate Authority Java Authentication and Authorization Service Oracle Internet Directory

Part I

Development

Part I discusses topics related to application development, and contains the following chapters:

- Chapter 2, "J2EE, Web Services, and Internet Applications"
- Chapter 3, "Portal Applications"
- Chapter 4, "Wireless Applications"
- Chapter 5, "Business Intelligence"
- Chapter 6, "Oracle Application Server Integration"
- Chapter 7, "Oracle Application Server Infrastructure"

J2EE, Web Services, and Internet Applications

This chapter describes the Java 2 Platform, Enterprise Edition (J2EE), Web Services, and Internet application development and deployment environment in Oracle Application Server, explaining its features and concepts. The topics include:

- Application Development and Deployment in Oracle Application Server
- Oracle HTTP Server
- Oracle Application Server Containers for J2EE
- Oracle Application Server TopLink
- Oracle Business Components for Java
- Oracle Application Server Web Services
- Oracle Application Server Forms Services
- Oracle XML Developer Kit
- Oracle Application Server PL/SQL Platform
- Oracle Content Management Software Developer Kit
- Oracle Application Server MapViewer

Application Development and Deployment in Oracle Application Server

Oracle Application Server provides an integrated, standards-based environment that provides for both developing applications and then deploying them, reliably delivering the applications to users across an enterprise. The following sections describe how this environment aids application development and deployment.

What Types of Applications Can Be Developed for Deployment in Oracle Application Server?

Oracle Application Server allows Web application developers to develop their sites in a variety of languages and technologies:

- **Java and J2EE**
- Web services
- **XML**
- **PL/SQL**

Table 2-1 lists the different technologies and programming languages that you can use to build applications for deployment with Oracle Application Server.

Table 2–1 Supported Technologies and Programming Languages

Java and J2EE	XML	PL/SQL	Web Services
<ul style="list-style-type: none"> ■ JavaServer Pages (JSP) v. 1.2 ■ Java Servlets v. 2.3 ■ Enterprise JavaBeans (EJB) v. 2.0 ■ Java Database Connectivity (JDBC) v. 2.0 Extensions ■ Java Transaction API (JTA) v. 1.0 ■ Java Naming and Directory Interface (JNDI) v. 1.2 ■ Java Message Service (JMS) v. 1.0.2b ■ Java Authentication and Authorization Service (JAAS) v. 1.0 ■ J2EE Connector Architecture v. 1.0 ■ Java API for XML Parsing (JAXP) v. 1.1 ■ Java Mail v. 1.0 	<ul style="list-style-type: none"> ■ XML v. 1.0 ■ XML Namespaces v. 1.0 ■ Document Object Model (DOM) v. 1.0/2.0 ■ Extensible Stylesheet Language Transformations (XSLT) v. 1.0 ■ XML Schemas v. 1.0 ■ Simple API for XML (SAX) v. 1.0/2.0 + Extensions ■ XML Path Language (XPath) v. 1.0 ■ XSQL ■ Internet Data Access Presentation (IDAP) 	<ul style="list-style-type: none"> ■ PL/SQL Server Pages v. 9.0.4 ■ PL/SQL Web Toolkit v. 9.0.4 ■ Oracle Application Server Forms Services v. 9.0.4 	<ul style="list-style-type: none"> ■ Web Services Description Language (WSDL) v. 1.1 ■ Universal Description, Discovery, and Integration (UDDI) v. 2.0 ■ Simple Object Access Protocol (SOAP) v. 1.1

The following sections explain how Oracle Application Server supports these technologies and programming languages.

Oracle HTTP Server

Oracle HTTP Server is the underlying deployment platform for all programming languages and technologies that Oracle Application Server supports. It provides a **Web listener** for Oracle Application Server Containers for J2EE (**OC4J**) and the

framework for hosting static and dynamic pages and applications over the Web. Based on the proven technology of the **Apache** HTTP Server, Oracle HTTP Server includes significant enhancements that facilitate load balancing, administration, and configuration. It also includes a number of enhanced **modules**, or **mods**, which are extensions to the **HTTP server** that extend its functionality for other enterprise applications and services.

Oracle HTTP Server allows developers to program their sites in a variety of languages and technologies, such as **Java**, Perl, C, C++, PHP, and PL/SQL. Additionally, it can serve as either a forward or reverse **proxy server**. The following sections describe how Oracle HTTP Server provides a robust deployment platform for dynamic Web sites and applications.

Oracle HTTP Server Components

Oracle HTTP Server consists of several components that run within the same process. These components provide the extensive list of features that Oracle HTTP Server offers when handling **client** requests. Major components include the following:

- **HTTP listener:** Oracle HTTP Server is based on an Apache **HTTP listener** to serve client requests.
- **Modules (mods):** Many of the standard Apache mods are included with Oracle HTTP Server. Oracle also includes several internal modules that are specific to Oracle HTTP Server components. Table 2–2 lists some of the Oracle HTTP Server modules.

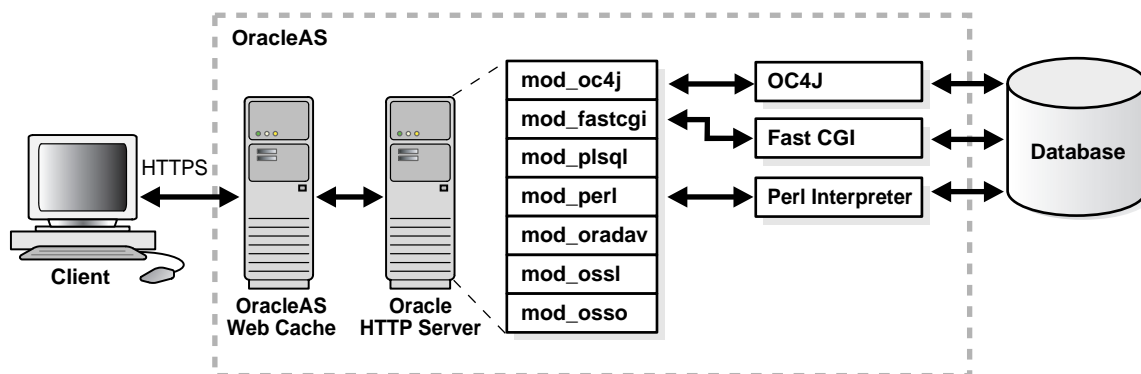
Table 2–2 *Prominent Oracle HTTP Server Modules*

Module	Description
mod_fastcgi	Supports FastCGI, which allows C, C++, and Java CGI programs to run in a performant environment
mod_perl	Routes requests to the Perl Interpreter
mod_plsql	Routes requests for stored procedures to the database server
mod_oc4j	Supports communication with Oracle Application Server Containers for J2EE and also performs some load balancing tasks
mod_oradav	Supports file as well as database distributed authoring and versioning
mod_ossll	Supports Secure Sockets Layer (SSL) and certificate sharing
mod_osso	Routes requests to Oracle Application Server Single Sign-On server

See Also: *Oracle HTTP Server Administrator's Guide* for a complete list of modules

Figure 2-1 shows the path of various requests through Oracle HTTP Server components.

Figure 2-1 Oracle HTTP Server Request Flow



Oracle HTTP Server Features

In addition to the standard Web server functionality of serving client requests to other Oracle Application Server components, Oracle HTTP Server provides the necessary features for both creating dynamic applications and implementing enterprise support. It also contains security enhancements that help protect important business resources. Key features of Oracle HTTP Server include the following:

- Dynamic monitoring services (DMS):** These services automatically measure runtime performance statistics for both Oracle HTTP Server and **Oracle Application Server Containers for J2EE (OC4J)** processes. As applications run, DMS collects detailed performance statistics. This data allows you to monitor the duration of important request processing phases and status information. With this information, you can locate performance bottlenecks and tune the application server to maximize throughput and minimize response time.
- Request ID:** To enhance request tracking through various components, a request ID is now attached to each request. This provides more detailed

tracking information, allowing you to see how much time a particular request spends in any component or layer.

- **External API for performance monitoring:** This API allows you to use external, third-party performance monitoring tools to monitor Oracle Application Server-based J2EE components, such as servlets and JSPs, as well as J2EE containers.
- **Proxy Plug-In for Using Non-Oracle HTTP Listeners:** To accommodate requirements for non-Oracle HTTP listeners, Oracle HTTP Server provides a proxy plug-in that can be plugged directly into Sun ONE or Microsoft **Internet Information Server (IIS)**. This proxy plug-in is used to forward requests for Oracle Application Server component services to Oracle HTTP Server, which is placed behind the non-Oracle listener.
- **OC4J Plug-In for IIS and Sun ONE:** The OC4J Plug-In provides a way for you to use the IIS and Sun ONE third-party listeners to access servlets running in OC4J without having to use OHS as a proxy. The OC4J Plug-In routes requests directly from the third-party HTTP listener to OC4J.
- **Oracle Application Server Single Sign-On Plug-in:** This plug-in is the Oracle single sign-on solution for third-party listeners such as Sun ONE and IIS. The plug-in is designed to protect native third-party listener applications using the single sign-on infrastructure. Using this plug-in, you can be authenticated to different third-party listener applications using only one password. You can integrate these protected third-party listener applications with other single sign-on enabled applications as long as they are all protected by the same single sign-on server.

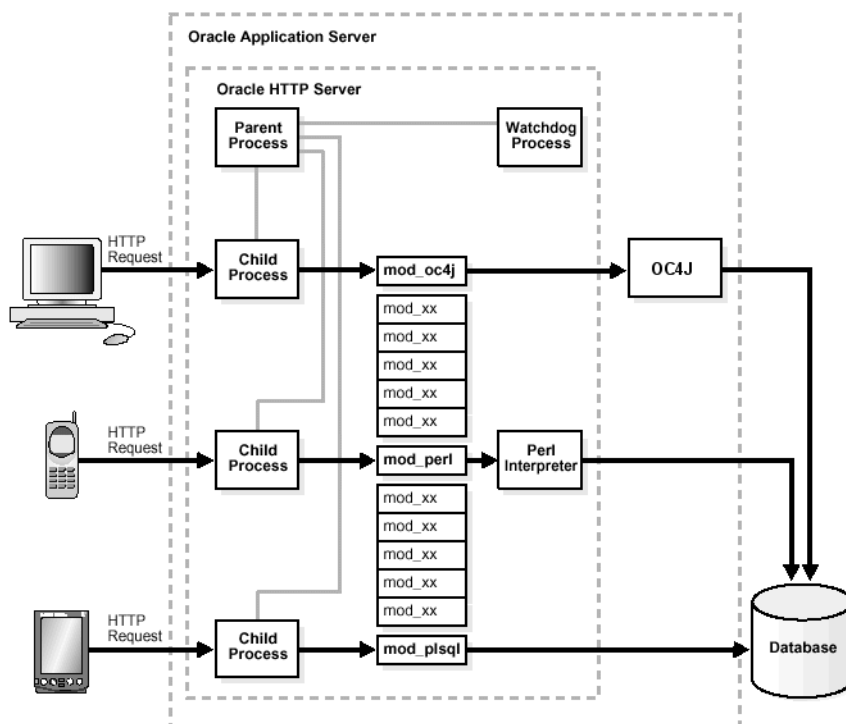
Oracle HTTP Server Architecture

At startup, the Web server parent process loads the entire configuration and the associated mods, and spawns a preconfigured number of child processes.

Note: On Windows systems, the Web server main process is a child process that spawns multiple threads.

Figure 2-2 shows the process architecture of Oracle HTTP Server in a UNIX environment.

Figure 2–2 Oracle HTTP Server Process Architecture



The parent process does not listen to **HTTP** requests. Its sole job is to ensure that the child processes are running or that new ones are started when the load requires it.

On UNIX platforms, each child process handles a single HTTP request. The child processes determine who should take the next request based on a mutex mechanism that you can configure.

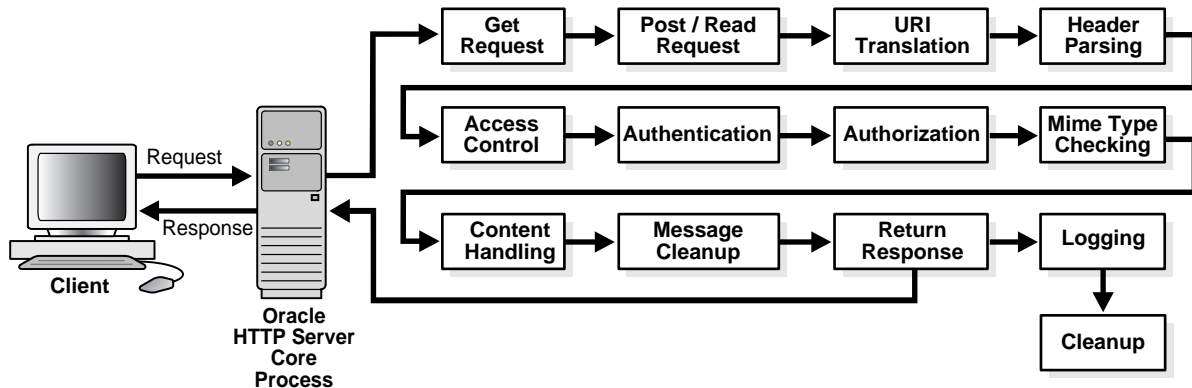
Note: On Windows platforms, threads handle HTTP requests instead of child processes.

Modular Architecture

The architecture of Oracle HTTP Server is modular. The core HTTP listener is very small, and all capabilities are implemented as modules that plug in and are invoked

at the appropriate place during the HTTP request lifecycle. Figure 2-3 shows the lifecycle of an HTTP request in Oracle HTTP Server.

Figure 2-3 Oracle HTTP Server HTTP Request-Response Cycle



A child process guides the request through this entire lifecycle. The modules register their application programming interfaces (APIs), which are then either invoked automatically when the request reaches a certain stage in its lifecycle, or can be configured to be invoked only in certain situations.

Common Gateway Interface (CGI) Support

Requests that are sent to a common gateway interface (CGI) program may invoke two new processes—the child process that handles the HTTP request and the CGI program itself. It is possible to avoid this overhead by configuring Oracle HTTP Server to pre-start child processes and keep them running, leveraging the Perl module to run the CGI programs in memory, or using the FastCGI mechanism.

The following Oracle HTTP Server features support CGI:

- FastCGI:** FastCGI supports applications written with Perl, C, C++, and Java. Each CGI application runs in a single child process of the Web server. This improves server performance because it eliminates the need to start a new process for every application request.
- Perl Interpreter:** The Perl Interpreter supports Perl applications and runs them inside of the Web server process. This allows for greater performance because running applications does not start new processes. Since the interpreter runs

inside of the Web server process, it has access to Web server services such as log files.

Oracle Application Server Containers for J2EE

Oracle Application Server Containers for J2EE (OC4J) is a fast, lightweight, and scalable **J2EE 1.3** certified server implementation that is written in **Java** and runs on a standard Java Virtual Machine (**JVM**). It has been designed to support the standard **APIs** in Table 2-3.

Table 2-3 Oracle Application Server Containers for J2EE Supported APIs

API	Version
JavaServer Pages (JSP)	1.2
Java Servlet	2.3
Enterprise JavaBeans (EJB)	2.0
Java Database Connectivity (JDBC)	2.0 Extensions
Java Transaction API (JTA)	1.0
Java Message Service (JMS)	1.0.2b
JavaMail	1.2
JavaBeans Activation Framework	1.0
Java API for XML (JAXP)	1.1
J2EE Connector Architecture	1.0
Java Authentication and Authorization Service (JAAS)	1.0

Introduction to J2EE Application Development

The following sections provide introductory definitions and summaries of application programming technologies that Oracle Application Server supports. For detailed information on the technologies introduced in this section, refer to the following information sources.

See Also:

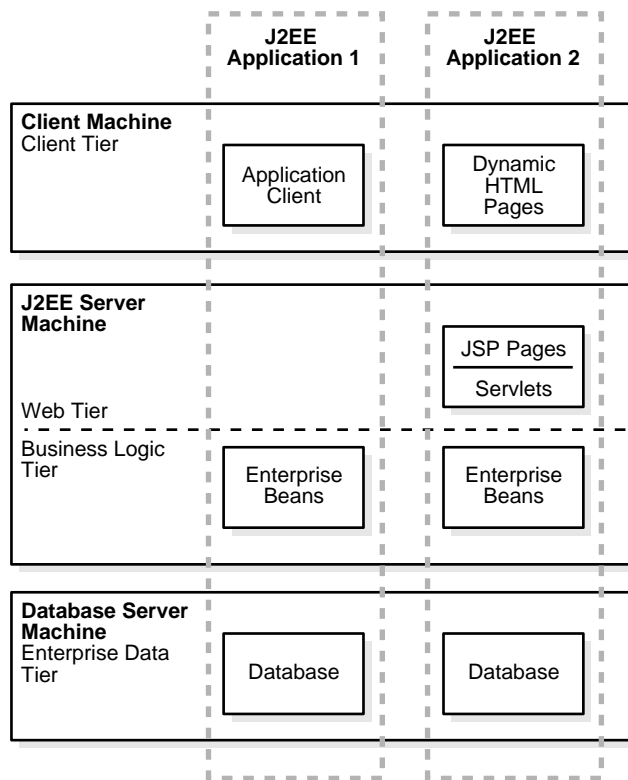
- The official J2EE Web site at <http://java.sun.com/j2ee>
- *The J2EE Tutorial* at <http://java.sun.com/j2ee/tutorial>
- The Oracle Application Server Documentation Library

What Is a J2EE Application?

A J2EE application is an application that is written in Java using the J2EE APIs. It can be deployed, managed, and executed on a J2EE-compatible server. The J2EE application itself is composed of a set of components, such as Web presentation modules, business logic modules, and data access modules. Each component is assembled into the overall application with all of its related classes and XML deployment descriptors.

J2EE Distributed Multi-tiered Application Model

The J2EE platform provided in Oracle Application Server uses a multi-tiered distributed application model. A multi-tiered distributed application model divides application logic into components according to function, and the various application components that make up J2EE applications can be installed on different machines, depending on which tier in the multi-tiered J2EE environment the application component belongs. Figure 2-4 shows two multi-tiered J2EE applications divided into the **client**, Web, business logic, and enterprise data tiers.

Figure 2–4 J2EE Distributed Multi-tiered Application Architecture

You can distribute J2EE applications across the four tiers shown in this figure, but generally they are considered to be three-tier applications because they are usually distributed over the following machine locations:

- client machines
- J2EE Server machines hosting presentation services, like JSPs and **servlets**, and business logic components, like EJBs
- database **servers** or legacy machines at the back end

Three-tiered applications that run in this way extend the standard two-tiered client and server model by placing an application server between the client and the back-end storage.

Types of J2EE Clients

J2EE applications support the following clients:

- **Application Clients:** Applications running on a client machine that directly access **enterprise beans** that are running in the business logic tier. Application clients can also open an **HTTP** connection to establish communication with a servlet running on the Web tier if a J2EE application requires it.
- **Dynamic HTML and XML Pages:** In the context of J2EE applications, dynamic HTML and XML pages are either generated by servlets or created with JavaServer Pages technology running in the Web tier. These pages can be extensions to traditional static HTML pages, allowing application developers to offer customized and personalized pages to the client.

Types of J2EE Application Components

You can use the following components in J2EE applications:

- **Servlets:** A servlet is a Java class that executes behind a Web server and can extend the capability of the Web server to provide services for dynamic page creation or application logic. The servlet works in the standard HTTP request-response model.
- **JavaServer Pages:** JavaServer Pages (JSPs) are text files that contain two types of information: static template data, which can be expressed in any text-based format, such as HTML, XML, or WML (Wireless Markup Language); and JSP elements, which construct dynamic content.
- **Enterprise Beans:** Enterprise beans are server-side components that encapsulate the business logic of an application.

Types of J2EE Containers

A **container** provides the runtime support for J2EE application components. Containers provide a federated view of the underlying J2EE APIs to the application components. J2EE application components never interact directly with other J2EE application components. They use the protocols and methods of the container for interacting with each other and with platform services. Interposing a container between the application components and the J2EE services allows the container to transparently inject the services defined by the components' deployment descriptors, such as declarative transaction management, security checks, resource pooling, and state management.

Before a Web or enterprise bean component can run, it must be assembled into a J2EE application and deployed into a J2EE container. The assembly process involves

specifying container settings for each component, which customize the underlying support provided by the J2EE server. These settings can be standard J2EE settings or container-specific settings, depending on your application requirements. Some of the container settings that you can specify include security services, **transaction** model, naming and directory lookup, and remote connectivity model.

There are two primary types of J2EE container:

- **Enterprise JavaBeans Container:** The Enterprise JavaBeans container (**EJB container**) manages the execution of all enterprise beans for J2EE applications. Enterprise beans and their container run on the J2EE server.
- **Web Container:** Web components such as JSP pages and servlets are managed and executed in the **servlet container**. The container provides services such as request dispatching, security, concurrency, and life cycle management. The Web container also gives **Web components**, typically JSPs and servlets, access to the J2EE APIs such as naming, transactions, and JDBC.

J2EE Application Packaging Concepts

J2EE components are packaged separately and bundled into a J2EE application. Each component, with its related files such as GIF and HTML files or server-side utility classes, are packaged together with a deployment descriptor (DD), and are assembled into a **module** that is added to the J2EE application. Typically, a J2EE application is composed of one or more enterprise beans and Web or application client component modules.

Oracle Application Server Containers for J2EE Features

Oracle Application Server Containers for J2EE (OC4J) has the following features.

Oracle Application Server Containers for J2EE Containers

OC4J supplies the following J2EE containers:

- A servlet container that complies with the servlet 2.3 specification
- A JSP container that complies with the Sun JSP 1.2 specification
- An EJB container that complies with the EJB 2.0 specification

Oracle Application Server Containers for J2EE Servlet Container A servlet is a Java program that runs on a J2EE server, such as OC4J. A servlet is one of the application component types of a J2EE application. It must execute under the control of a servlet container, which is part of the OC4J Web container. The servlet container

calls the servlet's methods and provides services that the servlet needs when running.

See Also: "Types of J2EE Application Components" on page 2-12

About the Servlet Container

The servlet container executes and manages servlets. It provides the servlet with access to properties of the HTTP request, such as headers and parameters. Also, the container provides the servlet with access to other Java APIs, such as **JDBC** to access a database, remote method invocation (**RMI**) to call remote **objects**, or **JMS** to perform asynchronous messaging.

How the Servlet Container Works

When a request is mapped to a servlet, the servlet container performs the following steps:

1. If an **instance** of the servlet does not exist, the container does the following:
 - a. Loads the servlet class
 - b. Instantiates an instance of the servlet class
 - c. Initializes the servlet instance
2. The container then invokes the servlet, passing request and response objects. The request object contains information about the client, request parameters, and **HTTP headers**. The response object returns the servlet's output to the client.

The servlet extracts information from the client request, accesses external resources, and then populates the response based on that information.

Oracle Application Server Containers for J2EE JavaServer Pages Container JavaServer Pages provide a convenient way to generate dynamic content in Web pages. JSP technology, which is closely coupled with servlet technology, allows you to include Java code fragments and make calls to external Java components (in the form of tags and directives) from within your Web pages. Typically, the markup code used to compose your Web pages is HTML or XML. JSPs work well as a front-end for business logic and dynamic functionality in **JavaBeans** or Enterprise JavaBeans (EJBs).

A JSP is translated into a Java servlet before being run, and it processes HTTP requests and generates responses like any servlet. However, JSP technology provides a more convenient way to code a servlet. Translation occurs the first time

the application is run. A **JSP translator** is triggered by the `.jsp` file name extension in a **URL**.

JSPs are fully interoperable with servlets. You can include output from a servlet or forward the output to a servlet, and a servlet can include output from a JSP or forward output to a JSP.

About the JSP Translator

The JSP translator has a translator and a compiler. The JSP translator translates a JSP into a Java source file. The container compiles the source file into a Java bytecode (`.class`) file, which executes as a servlet in the servlet container using the JSP runtime library. The servlet container provides access to Java APIs and other services.

How the JSP Translator Works

When a user requests a URL that maps to a JSP file, such as `http://host/Hello.jsp`, the following steps occur:

1. The Web server invokes the JSP translator, which translates `Hello.jsp` and produces the file `Hello.java`.
2. The Java compiler is invoked, creating a `Hello.class` servlet.
3. `Hello.class` runs, using the JSP runtime library, which contains the supporting files to interpret the tags and directives from the JSP.
4. If the `Hello` class requires information from a database, then the servlet container provides JDBC access to the class so it can retrieve the information and return its output to the client browser.

Oracle Application Server Containers for J2EE Enterprise JavaBeans Container The OC4J **EJB container** manages the execution of enterprise beans for J2EE applications. Like the OC4J Web container, the EJB container uses the J2SE **virtual machine**. The following sections describe what services the EJB container provides to J2EE applications and how it works.

Enterprise beans are the J2EE components that implement Enterprise JavaBeans technology. Enterprise beans run in the EJB container. An enterprise bean is a portable server-side component that encapsulates the business logic of an application. There are three types of EJBs: **session beans**, **entity beans**, and **message-driven beans**.

About the Oracle Application Server Containers for J2EE EJB Container

The OC4J EJB container provides system-level services to EJBs similar to the services that the Web container provides to servlets and JSPs. The container has configurable settings that customize the underlying support provided by OC4J, the J2EE server. The configurable settings include security, transaction management, Java Naming and Directory Interface (JNDI) lookups, and remote connectivity. In addition to the configurable settings, the container also manages EJB life cycles, database connection resource pooling, data persistence, and access to the J2EE APIs.

How the EJB Container Works

How the EJB container works depends on what type of enterprise bean you are using. The container manages the execution of the enterprise bean for one J2EE application.

For **session beans**, the EJB container provides all of the services that the Web container provides to Web components, such as access to APIs and the **virtual machine**, **transaction services** like Container Managed Transactions (CMTs), and secure and authorized EJB method invocation.

For entity beans, which represent business objects in a persistent storage mechanism, there are models for how the persistence is performed. You can have either **bean-managed persistence (BMP) beans** or **container-managed persistence (CMP) beans**. With bean-managed persistence, the entity bean code contains the calls that access the database and the EJB container triggers callback methods on your code. Entity beans with bean-managed persistence execute in the EJB container with the typical container support and services. However, if you are using container-managed persistence, then the EJB container automatically generates the necessary database access calls. The EJB methods do not require any JDBC code to manage EJB data persistence.

J2EE Services

Java 2 Platform Enterprise Edition (J2EE) provides core services for writing J2EE components. The J2EE containers manage access to these services for the application components. The services are as follows:

- **Java Database Connectivity (JDBC):** This service lets you invoke **SQL** commands from Java programming methods. You use the JDBC API in an enterprise bean when you override the default container-managed persistence or have a session bean access the database. You can also use the JDBC API from a servlet or a JSP to access a database directly without going through an enterprise bean.

Oracle Application Server includes the following drivers to provide highly scalable and reliable connectivity to both Oracle and non-Oracle data sources:

- **Oracle JDBC drivers:** The Oracle JDBC drivers, in addition to providing standard JDBC API support, have extensions to support Oracle-specific datatypes and to enhance their performance. They are meant to be used with the Oracle database.
- **J2EE Connectors:** The J2EE Connector architecture, part of the J2EE platform, provides a Java-based solution for connecting various application servers and enterprise information systems that are already in place.
- **DataDirect Connect Type 4 JDBC drivers:** DataDirect JDBC drivers are meant specifically for connecting to non-Oracle databases, such as Microsoft SQL Server and Sybase.
- **Java Message Service (JMS):** This service is a messaging standard that allows J2EE application components to create, send, receive, and read **messages**. It enables distributed communication that is loosely coupled, reliable, and asynchronous.

For this release, JMS support has been enhanced by the addition of a lightweight JMS provider in addition to the Oracle JMS offered in previous versions. The new lightweight JMS is fully JMS 1.0 compatible and can support durable messaging through a file-based persistence mechanism, which provides improved stability and performance. Support for Message Driven Beans is also now available for both Oracle JMS (AQ) and the lightweight OC4J JMS.

- **Java Transaction API (JTA):** This service provides a standard demarcation interface for demarcating transactions. Typically, it is used in J2EE applications that use two or more separate database access operations that depend on each other to demarcate where the entire transaction begins, rolls back, and commits.

OC4J provides additional support for two-phase commits for applications that require commit coordination across machines and containers. The two-phase commit engine is responsible for ensuring that when a distributed transaction ends, changes to all participating databases are all either committed or rolled back.

- **Java Naming and Directory Interface (JNDI):** This service provides a standard interface to naming and directory services. J2EE applications use JNDI to find other distributed objects. The JNDI Interface has two parts: an application-level interface used by application programs to access naming and directory services, and a service provider interface to attach a provider of naming and directory services.

- **JavaMail Technology:** This service provides J2EE applications with a JavaMail API and a JavaMail service provider with which to send e-mail notifications.
- **Java API for XML (JAXP):** This service provides support for the industry standard SAX and DOM APIs for parsing XML documents, as well as support for XSLT transform engines. It enables applications running in a J2EE container to make use of XML.
- **J2EE Connector Architecture:** This service provides an interface for resource adapters that allow J2EE applications to access and interact with databases and other enterprise information systems (EIS). Oracle Application Server provides several out-of-the box adapters, and also allows you to build your own.
- **Java Authentication and Authorization Service (JAAS):** This service provides a way for a J2EE application to authenticate users against different security provider systems and authorize a specific user or group of users to run J2EE applications under **role** permission and control enforcement.

Oracle Application Server provides an implementation of Java Authentication and Authorization Service (JAAS) that integrates with the Oracle Application Server J2EE security infrastructure to enforce security constraints for Web components (servlets and JSPs) and EJB components. The Oracle Application Server Java Authentication and Authorization Service implementation does the following:

- Integrates Java-based applications with Oracle Application Server Single Sign-On, including **authentication**, thereby giving you extensible security for Java-based applications
- Manages access control policies centrally in Oracle Internet Directory, controls access by role, and partitions security **policy** by subscriber
- Supports impersonation of a specific user, allowing an enterprise bean, servlet, or JSP to run with the permissions associated with the current client or a specified user

See Also: The Sun Microsystems, Inc. published specifications for these services, and *The J2EE Tutorial* for instructions on how to use them in J2EE applications, at:

<http://java.sun.com>

Oracle J2EE Services

In addition to the standard J2EE Services, Oracle also provides the following services to Java developers:

- **Java Object Cache** stores frequently accessed or resource-intensive objects in memory or on disk. It is a low-level object caching API, supporting generic object types such as memory objects, disk objects, pooled objects, and StreamAccess objects. Java Object Cache uses distributed object management to coordinate updates and invalidations of Java objects.

Objects are loaded using a user-defined CacheLoader object and accessed through the easy-to-use API. This eliminates the need to repeatedly create and load information within a Java application. The Java Object Cache retrieves content quickly and greatly reduces the load on Oracle Application Server.

Java Object Cache provides caching for expensive or frequently used Java objects when the application servers use a Java program to supply their content. Cached Java objects may contain generated pages, or they may provide support objects within the program to assist in creating new content. Java Object Cache automatically loads and updates objects as specified by the Java applications and includes APIs to manage the cached Java objects. The generic nature of Java Object Cache makes it an ideal cache repository for higher level caches such as Web Object Cache, easing the development effort and reducing complexity.

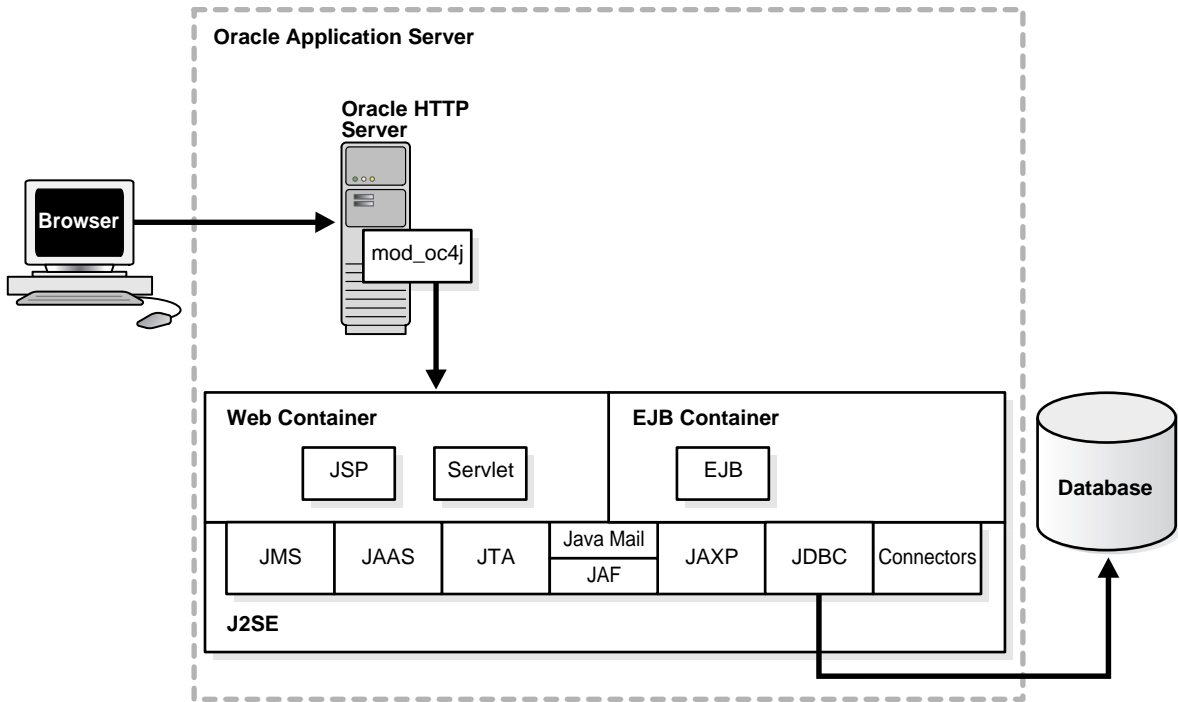
- **Oracle Business Components for Java (BC4J)** is a Java and XML framework that enables productive development, portable deployment, and flexible customization of multi-tier database applications from reusable business components. BC4J is an application component framework that provides developers with a set of intelligent software building-blocks that cooperate to manage all of the common facilities required to do the following:
 - Productively author and test business logic in components that automatically integrate with relational databases
 - Flexibly reuse business logic through multiple SQL-based views of data, supporting different application tasks
 - Efficiently access and update the views from servlets, JavaServer Pages (JSPs), XML clients and thin-Java Swing clients
 - Easily customize application functionality in layers without requiring modification of the delivered application

See Also: "Oracle Business Components for Java" in this chapter

Oracle Application Server Containers for J2EE Architecture

Figure 2–5 shows the architecture of OC4J within Oracle Application Server.

Figure 2-5 Oracle Application Server Containers for J2EE Architecture



OC4J is supported by the Java 2 Platform, Standard Edition (**J2SE**) infrastructure as shown in Figure 2-5. This means that the OC4J Web container and OC4J **EJB container** use the **J2SE virtual machine**. J2EE applications are modularized for reuse of application components, such as:

- User interfaces, which can be composed of JavaServer Pages (**JSPs**), dynamic **HTML**, and so on
- Business logic, which is usually contained in an Enterprise JavaBean (**EJB**) or normal Java classes

The J2EE containers also perform services for applications, such as providing access to the APIs and lifecycle management.

Oracle Application Server TopLink

Oracle Application Server TopLink is an advanced object-to-relational persistence framework, suitable for a wide range of Java 2 Enterprise Edition (J2EE) and Java application architectures. OracleAS TopLink development tools and runtime capabilities reduce development and maintenance efforts, and increase enterprise application functionality. Use OracleAS TopLink to build high-performance applications that store persistent data in a relational database.

The following sections introduce OracleAS TopLink and include discussions on these topics:

- Advantages of OracleAS TopLink
- OracleAS TopLink Components
- Application Development with OracleAS TopLink
- OracleAS TopLink Architectures Overview

Advantages of OracleAS TopLink

Enterprise applications rely on Java-to-database integration to implement objects and logic. OracleAS TopLink enables developers to efficiently develop and refine enterprise applications. To fully understand OracleAS TopLink, you must understand the problems that enterprise application developers face and how OracleAS TopLink resolves them.

The OracleAS TopLink Problem Space

Java-to-database integration is a widely underestimated problem in enterprise Java applications. This complex problem involves more than reading from and writing to a database. The database world includes elements such as tables, rows, columns, and primary and foreign keys; the Java and J2EE world contains entity classes (regular Java classes or Enterprise JavaBeans (EJB) entity beans), business rules, complex relationships, and inheritance. Bridging these two fundamentally different technologies is a challenging and resource-intensive problem.

The process of translating object-oriented data into relational data is referred to as *object-relational (O-R) mapping*. To enable an O-R solution, developers must resolve the following O-R bridging issues:

- Fundamentally different technologies
- Different skill sets

- Different staff and ownership for each of the technologies
- Different modeling and design principles

Application developers need a product that enables them to integrate Java applications and relational databases, without compromising ideal application design or database integrity. In addition, Java developers need the ability to store (or *persist*) and retrieve business domain objects using a relational database as a repository.

The OracleAS TopLink solution is a persistence framework that manages O-R mapping in a seamless manner and enables developers to rapidly build applications that combine the best aspects of object technology and relational databases.

The OracleAS TopLink Solution

OracleAS TopLink provides a mature and powerful solution that addresses the disparity between Java objects and relational databases. OracleAS TopLink enables developers to:

- Persist Java objects in virtually any relational database supported by a JDBC 2.0 compliant driver
- Map any object model to any relational schema, using the Oracle Application Server TopLink Mapping Workbench graphical mapping tool
- Use OracleAS TopLink successfully, even if they are unfamiliar with SQL or JDBC, because OracleAS TopLink provides a clean, object-oriented view of relational databases

Other OracleAS TopLink Advantages In addition to providing industry-leading O-R mapping capabilities, OracleAS TopLink provides flexibility, increases performance and maximizes the productivity of your applications. OracleAS TopLink provides the following features:

- Advanced object caching that improves performance by minimizing database access.
- Rich query support that provides easy access to sophisticated, dynamic query languages and tools such as *query by example*, Java expression-based queries, EJB QL, and SQL.
- A transactional framework that enables developers to easily create and modify mapped objects. This framework integrates the complexities of a shared memory space and caches, and provides scalability that supports multiple server instances (clustering). Although the mechanisms involved are complex,

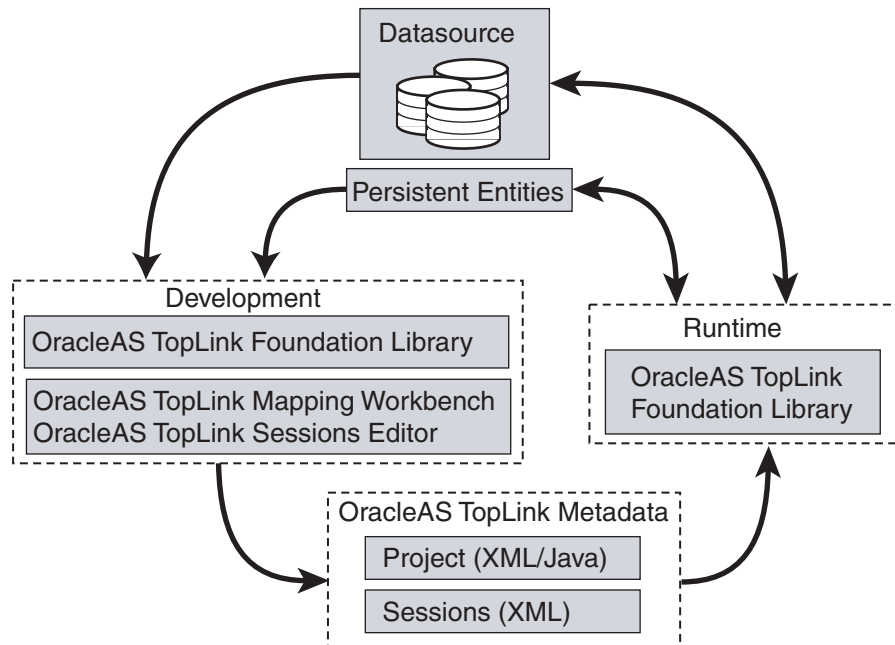
OracleAS TopLink makes it easy to leverage this functionality by simplifying the task of writing transactional code that complies with database referential integrity and optimal access patterns.

OracleAS TopLink Components

At its core, OracleAS TopLink is a runtime engine that provides Java or J2EE applications with access to persistent entities stored in a relational database. In addition to runtime capabilities, the Oracle Application Server TopLink Foundation Library includes the OracleAS TopLink Application Programming Interface (API). This API enables applications to access OracleAS TopLink runtime features, as well as development tools that simplify application development. The tools capture mapping and runtime configuration information in metadata files that TopLink passes to the runtime.

Figure 2-6 shows how the OracleAS TopLink components relate to one another throughout the development cycle.

Figure 2-6 TopLink Components in the Development Cycle



OracleAS TopLink Development Components

OracleAS TopLink application development comprises three elements: the development environment, the OracleAS TopLink runtime, and the metadata that ties them together.

Development To create an OracleAS TopLink application, map the object and relational models using the OracleAS TopLink Mapping Workbench, and capture the resulting mappings and additional runtime configurations in the OracleAS TopLink project file (the `project.xml` file). Then build a session configuration file (the `sessions.xml` file) in the OracleAS TopLink Sessions Editor. These files together represent your entire OracleAS TopLink project.

During development, developers leverage the OracleAS TopLink API to define query and transaction logic. When developers use EJB entity beans, there is generally little or no direct use of the OracleAS TopLink API.

Runtime The OracleAS TopLink Foundation Library provides the OracleAS TopLink runtime component. Access the runtime component either directly through the OracleAS TopLink API, or indirectly through a J2EE container when using EJB entity beans. The runtime engine is not a separate or external process; instead, it is embedded within the application. Application calls invoke OracleAS TopLink to provide persistence behavior. This function allows for transactional and thread-safe access to shared database connections and cached objects.

Metadata OracleAS TopLink metadata is the bridge between the development of an application and its deployed runtime. Capture the metadata using the OracleAS TopLink Mapping Workbench and OracleAS TopLink Sessions Editor, and pass the metadata to the runtime using deployment `project.xml` and `sessions.xml` files. It is also possible to hand-code these files using Java and the OracleAS TopLink API, but this approach is more labor-intensive.

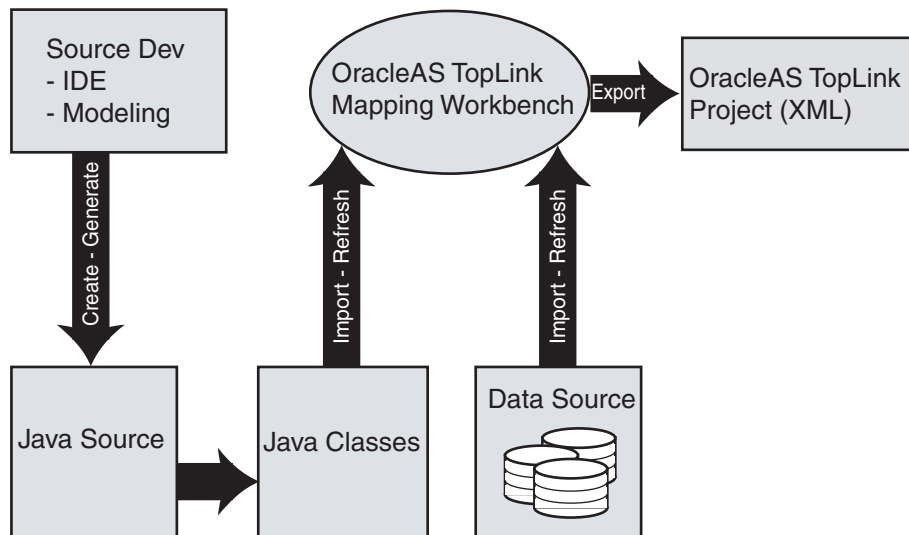
The metadata, encapsulated in the `project.xml` file and the `sessions.xml` file, allows developers to pass configuration information into the runtime environment. The runtime uses the information in conjunction with the persistent entities (Java objects or EJB entity beans), and the code written with the OracleAS TopLink API, to complete the application.

Oracle Application Server TopLink Mapping Workbench

The OracleAS TopLink Mapping Workbench is a graphical development tool that enables developers to map between the object and relational models, and configure many of the OracleAS TopLink Foundation Library features. The OracleAS TopLink

Mapping Workbench creates an OracleAS TopLink project, the primary object in the OracleAS TopLink metamodel. Export the project as a single deployment XML file (the `project.xml` file), which OracleAS TopLink uses in conjunction with the OracleAS TopLink runtime to provide the application-specific persistence capabilities. Figure 2-7 shows how the OracleAS TopLink Mapping Workbench fits into the OracleAS TopLink environment.

Figure 2-7 The OracleAS TopLink Mapping Workbench in a TopLink Environment



The OracleAS TopLink Mapping Workbench can import compiled entity classes (Java objects or EJB entity beans), as well as relational schema through a JDBC driver that the developer configures. Because OracleAS TopLink imports the object and relational models for mapping, developers can develop the two models relatively independently from the O-R mapping phase of project development.

Oracle Application Server TopLink Sessions Editor Most OracleAS TopLink applications include a session configuration file, the `sessions.xml` file, to simplify the application deployment process. The OracleAS TopLink Sessions Editor provides a graphical environment in which to configure the `sessions.xml` file.

Use the `sessions.xml` file to configure one or more sessions for the OracleAS TopLink project, and associate the sessions with the project. This approach allows

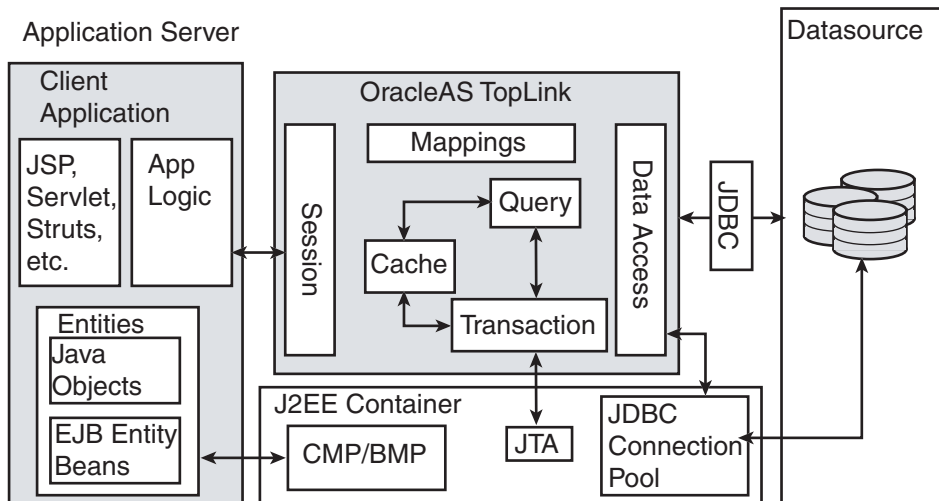
developers to specify individual configurations for each session and to add or modify the following:

- Database (JDBC) login information different from the login information used during development (for example, external data sources for the host application server's connection pools)
- JTA/JTS transaction usage
- Cache synchronization
- Session broker (enables client applications to view multiple databases and projects as a single OracleAS TopLink session)

Oracle Application Server TopLink Foundation Library

The Oracle Application Server TopLink Foundation Library includes a Java library that forms the runtime component of the product. It provides support and the API for the components that make up an OracleAS TopLink application. The API enables developers to interact with OracleAS TopLink to retrieve and modify their application persistent entities. Figure 2-8 shows the interaction of the parts of an OracleAS TopLink application.

Figure 2-8 OracleAS TopLink Application Components



Note: Although this section describes how these components fit into J2EE architectures, note that OracleAS TopLink also supports non-J2EE solutions. The *Oracle Application Server TopLink Application Developer's Guide* describes these solutions in more detail.

Sessions A session is the primary interface between the client application and OracleAS TopLink, and represents the connection to the underlying relational database. OracleAS TopLink offers several different session types, each optimized for different design requirements and architectures. The session manager configures and manages the session as a singleton within the application.

The most commonly-used session is the server session, a singleton session that clients access on the server through a client session. The server session provides a shared cache and shared JDBC connection resources. OracleAS TopLink also supports sessions for two-tier architectures, distributed applications, and multiple databases.

Data Access The OracleAS TopLink data access component provides access to JDBC connections through connection pooling, provided either by OracleAS TopLink or a host application server. This component manages the SQL generation required by the various query operations and reconciles any differences between JDBC drivers and SQL dialects. OracleAS TopLink offers many performance tuning options that optimize its data access capabilities.

Caching OracleAS TopLink supplies an object level cache that guarantees object identity and provides performance enhancement. Developers can configure the OracleAS TopLink cache and maximize the application efficiency by reducing the number of times the application accesses the database. In a clustered environment, developers can configure OracleAS TopLink to synchronize changes with other instances of the deployed application.

Queries The OracleAS TopLink query framework provides developers with the flexibility necessary to manage the complex persistence requirements of enterprise applications. The key features of this query framework include:

- A rich set of query types to allow object retrieval, summary results, and raw data retrieval
- The ability to specify the search criteria using OracleAS TopLink Expressions (for object model based queries), EJB QL, SQL, stored procedures, or query by example

- Configuration options that enable developers to specify how the query is executed, and customize many of its performance optimizing features

Developers can define OracleAS TopLink queries using the OracleAS TopLink Mapping Workbench, in Java code using the OracleAS TopLink API, or, in the case of EJB entity beans, through EJB Finders.

Transactions OracleAS TopLink provides the ability to write transactional code isolated from the underlying database and schema. OracleAS TopLink achieves this functionality through the Unit of Work.

The Unit of Work isolates changes in a transaction from other threads until it successfully commits the changes to the database. Unlike other transaction mechanisms, the Unit of Work automatically manages changes to the objects in the transaction, the order of the changes, and changes that might invalidate other OracleAS TopLink caches. The Unit of Work manages these issues by calculating a minimal change set, ordering the database calls to comply with referential integrity rules and deadlock avoidance, and merging changed objects into the shared cache. In a clustered environment, the Unit of Work also synchronizes changes with the other servers in the cluster.

If an application uses EJB entity beans, developers do not access the Unit of Work API directly, but they still benefit from its features: the integration between the OracleAS TopLink runtime and the J2EE container leverages the Unit of Work automatically.

JTA/JTS Integration: By default, OracleAS TopLink allows the application to create transaction boundaries for all object-level changes. OracleAS TopLink explicitly manages the database transaction, and if it encounters problems, safely rolls back both the database changes and the object-level changes.

In the case of a J2EE application, developers can configure OracleAS TopLink to synchronize with the JTA/JTS subsystem of the host application server. This feature allows an application to use container-managed transactions, rather than the default user-managed transactions.

Note that this functionality is not limited to EJB architectures. Developers can configure any OracleAS TopLink architecture to use container-managed transactions.

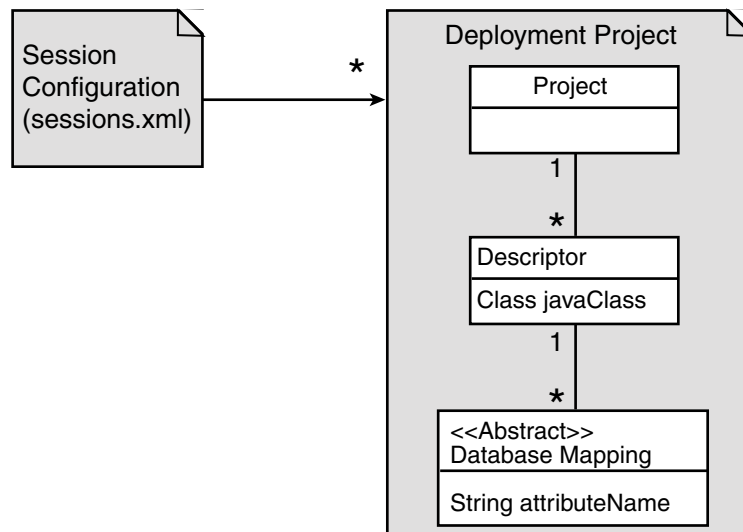
OracleAS TopLink Metadata

The OracleAS TopLink approach to persistence is based on metadata that defines the class structure (objects) and relational schema, along with other configuration information used by OracleAS TopLink at runtime. Developers can use the

OracleAS TopLink Mapping Workbench to define this metadata, and the OracleAS TopLink runtime component uses the metadata to provide the necessary persistence capabilities, using Java's reflective and introspective capabilities.

The TopLink application metadata model is based around the OracleAS TopLink project. The project includes descriptors, mappings, and various policies that customize the runtime capabilities. Figure 2-9 shows this metadata model.

Figure 2-9 OracleAS TopLink Metadata



Sessions XML Use the `sessions.xml` file to configure sessions for the project. Developers can build and edit these files with the OracleAS TopLink Sessions Editor. The session manager uses the `sessions.xml` configuration file during application initialization.

Project The OracleAS TopLink deployment project is the primary container for the metadata. A project generally represents an application and contains the mapping information for all persistent classes and their relationships. Each session (excluding the session broker) in the deployed application references a single project. Although developers can build a project by coding it using the OracleAS TopLink API, we recommend that developers create and manage the project in the OracleAS TopLink Mapping Workbench, and use the OracleAS TopLink Mapping Workbench to generate either an XML or Java source version of the project for use at runtime.

Descriptor A descriptor represents the association between a persistent Java class and a relational table or tables. The descriptor contains configuration information for the class level within a project, as well as a set of mappings for each of its persistent attributes. Many of the more advanced configuration options are set at the descriptor level. The OracleAS TopLink Mapping Workbench supports most of these options, but there are a few that developers must set using the OracleAS TopLink API.

Mappings Mappings describe how the attributes of a mapped class are associated with columns in the database. OracleAS TopLink provides a sophisticated set of flexible and customizable mappings that allow for complex mapping scenarios between the object and relational models.

There are two types of mappings: direct mappings and relationship mappings.

- **Direct Mappings:** Direct mappings relate an attribute or attributes to a column or columns in the relational schema. OracleAS TopLink provides several direct mappings that allow for conversions between the types from the database and the object model's attribute types. Here are the direct mappings and their functions:
 - **Direct-to-field mappings** map a Java attribute directly to a value database column.
 - **Type conversion mappings** explicitly map a database type to a Java type.
 - **Object type mappings** match a fixed number of database values to Java objects.
 - **Serialized object mappings** store large data objects, such as multimedia files and BLOBs, in the database.
 - **Transformation mappings** offer specialized translations between how a value is represented in Java and in the database, such as when developers map multiple fields into a single attribute.
- **Relationship Mappings:** OracleAS TopLink offers sophisticated relationship mapping, which enables developers to represent object relationships based on the database table columns and foreign keys. Here are the relationship mappings and their functions:
 - **One-to-one mappings** represent simple pointer references between two Java objects. The references use any of foreign keys, target foreign keys, or variable classes to define the pointer.

- **Aggregate object mappings** represent the relationship between a given object and a target object. The objects have a strict one-to-one relationship, and all the attributes of the second object are retrievable from the same table as the owning object.
- **Aggregate collection mappings** represent the relationship between a single-source object and a collection of target objects. Unlike one-to-many mappings, in which there must be a one-to-one back reference mapping from the target objects to the source object, there is no back reference required for the aggregate collection mappings, because the foreign key relationship is resolved by the aggregation (object and collection).
- **One-to-many mappings** represent the relationship between a single source object and a collection of target objects.
- **Many-to-many mappings** represent the relationships between a collection of source objects and a collection of target objects. They require an intermediate table for managing the associations between the source and target records.
- **Object-relational mappings** are mappings that leverage databases that support object-relational entity storage within tables.

Application Development with OracleAS TopLink

Using OracleAS TopLink to build an application does not affect the choice of development tools or the creative process. However, OracleAS TopLink does influence how developers approach development. This section highlights some of the key areas in which using OracleAS TopLink affects application development. These areas exist, regardless of whether developers are building an application to support Java objects, EJB entity beans, or both.

Mapping

OracleAS TopLink maps the application's persistent entities to the database, using the descriptors and mappings developers build with the OracleAS TopLink Mapping Workbench. The OracleAS TopLink Mapping Workbench supports several approaches to project development, including:

- Importing classes and tables for mapping
- Importing classes and generating tables and mappings
- Importing tables and generating classes and mappings

- Creating both class and table definitions with mapping creation and model generation

The OracleAS TopLink Mapping Workbench supports all these options; however, the most common solution is to develop the persistent entities using a development tool, such as an integrated development environment (IDE) or modeling tool, and to develop the relational model through appropriate relational design tools. Developers then use the OracleAS TopLink Mapping Workbench to construct mappings that relate these two models.

The OracleAS TopLink Mapping Workbench does offer some facilities for generating persistent entities or the relational model components for an application; however, these utilities are intended only to assist in rapid initial development strategies, rather than complete round-trip application development.

See Also: *Oracle Application Server TopLink Mapping Workbench User's Guide*

Session Management

Sessions are the primary interface between the application and OracleAS TopLink persistence capabilities. When developing an OracleAS TopLink application, developers must ensure that they properly initialize and manage the sessions.

When using EJB entity beans with container-managed persistence (CMP) or bean-managed persistence (BMP), the client code that modifies the entity beans does not access the OracleAS TopLink session directly. Instead, changes occur transparently, through integration with the container or through EJB callbacks.

Well-designed applications that employ Java objects as persistent entities use the session manager provided in the OracleAS TopLink API. This class initializes and manages the singleton session. Developers configure the session manager in the `sessions.xml` file, which allows for easy configuration and customization of the deployed application.

Querying

OracleAS TopLink furnishes several object and data query types, and offers flexible options for query selection criteria, including:

- OracleAS TopLink expressions
- EJB QL
- SQL

- Stored procedures
- Query by example

With these options, developers can build any type of query. We recommend that developers use predefined queries to define application queries. Predefined queries are held in the project metadata and referenced by name. This simplifies application development and encapsulates the queries to reduce maintenance costs.

The OracleAS TopLink Mapping Workbench provides the simplest way to define queries. Developers can also build queries in code, using the OracleAS TopLink API.

If the application includes EJB entity beans, developers can code finders completely using EJB QL, which enables the application to comply with the J2EE specification. Alternatively, developers can use any of the other OracleAS TopLink query options. All querying options are available, regardless of the architecture or persistent entity type.

Transactions

In an OracleAS TopLink application, the Unit of Work ensures that OracleAS TopLink transactions comply with the transactional requirements of the application.

The Unit of Work is one of the most sophisticated and powerful components of the OracleAS TopLink Foundation Library. Although developers that use CMP or BMP entity beans do not use the OracleAS TopLink API to apply transactional changes to their persistent entities, the Unit of Work is used behind the scenes. Understanding how the Unit of Work behaves, and developing simple coding patterns to use it, are the keys to building efficient, maintainable applications.

Packaging and Deployment

Application packaging (for deployment in the host Java or J2EE environment) influences OracleAS TopLink use and configuration. For example, developers package a J2EE enterprise application in an Enterprise Archive (EAR) file. Within the EAR file, there are several ways to package persistent entities within Web Application (WAR) and Java libraries (JAR). How developers configure OracleAS TopLink depends, in part, on how they package the application and how they use the host application server class loader.

Monitoring and Performance Tuning

OracleAS TopLink enables developers to monitor functionality and performance throughout application development, testing, and quality assurance cycles.

OracleAS TopLink offers many textual logging features, as well as the API required to implement custom logging strategies. Developers can use these features to ensure that the application behaves and performs as they expect.

OracleAS TopLink includes a performance profiler feature, available through the OracleAS TopLink Foundation Library API. This runtime feature tracks query execution time, which developers can use for performance analysis. This tool provides the information necessary to identify bottlenecks that hinder application performance.

OracleAS TopLink also offers a rich set of performance enhancement features. Understanding how to configure these features can have a strong influence on application performance, especially in the later phases of application development.

OracleAS TopLink Architectures Overview

OracleAS TopLink is designed to work in both Java and J2EE applications. Since it was first introduced, the flexibility OracleAS TopLink provides has led to its use in many architectural styles. This section introduces the five most common architectures associated with OracleAS TopLink. Although this section describes the architectures in relation to J2EE, OracleAS TopLink continues to fully support non-J2EE and Java applications as well.

Three-Tier

The three-tier (or J2EE Web) application is one of the most common OracleAS TopLink architectures. This architecture is characterized by a server-hosted environment in which the business logic, persistent entities, and the OracleAS TopLink Foundation Library all exist in a single Java virtual machine (JVM).

The most common example of this architecture is a simple three-tier application in which the client browser accesses the application through servlets, Java Server Pages (JSPs), and HTML. The presentation layer communicates with OracleAS TopLink through other Java classes in the same JVM, to provide the necessary persistence logic. This architecture supports multiple servers in a clustered environment, but there is no separation across JVMs from the presentation layer and the code that invokes the persistence logic against the persistent entities using OracleAS TopLink.

EJB Session Bean Facade

A popular variation on the three-tier application involves wrapping the business logic, including the OracleAS TopLink access, in EJB session beans. This architecture provides a scalable deployment and includes integration with transaction services

from the host application server. Communication from the presentation layer occurs through calls to the EJB session beans. This architecture separates the application into different tiers for the deployment.

The session bean architecture can persist either Java objects or EJB entity beans.

EJB Entity Beans with CMP

OracleAS TopLink provides CMP support for applications that require the use of EJB entity beans. This support is available on the leading application servers. OracleAS TopLink CMP support provides the developer with an EJB 1.1 and 2.1 CMP solution transparent to the application code, but still offers all the OracleAS TopLink runtime benefits.

Applications can access OracleAS TopLink-enabled EJB entity beans using CMP directly from the client, or from within a session bean layer. OracleAS TopLink also offers the ability to use regular Java objects in relationships with EJB entity beans.

EJB Entity Beans with BMP

Another option for using EJB entity beans is to leverage OracleAS TopLink BMP in the application. This architecture enables developers to access the persistent data through the EJB API, but is platform-independent.

The BMP approach is portable—that is, after a developer creates an application, you can move it from one application server platform to another.

Two-Tier

A two-tier (or client-server) application is one in which the OracleAS TopLink application accesses the database directly. Although less common than the other architectures discussed here, OracleAS TopLink supports this architecture for smaller or embedded data processing applications.

Oracle JDeveloper

Oracle JDeveloper is a J2EE and XML development environment with end-to-end support for developing, debugging, and deploying business applications and Web services. To maximize developer productivity, JDeveloper provides a comprehensive set of integrated tools to support the complete development life cycle, including the industry's fastest Java debugger and innovative profiler, and CodeCoach tools for code performance analysis and improvement. JDeveloper simplifies J2EE development by providing wizards, editors, visual design tools, and

deployment tools to create high-quality, standard J2EE components, including applets, JavaBeans, JavaServer pages, servlets, and Enterprise JavaBeans.

To simplify the development of scalable, high-performance J2EE applications, JDeveloper offers an open and extensive J2EE framework called Oracle Business Components for Java (BC4J). BC4J is an object-relational mapping tool that implements the Sun J2EE design patterns, allowing developers to quickly build sophisticated J2EE applications.

Oracle JDeveloper Features

Oracle JDeveloper is written entirely in Java and is certified for use with all major platforms, including Windows, Solaris, and Unix. The following are some key features of Oracle JDeveloper.

- **CodeCoach Utility and Profiling Tools:** To assist developers with optimizing Java code, JDeveloper provides a code coaching utility and three integrated profiling tools. With CodeCoach, you can increase the robustness of your code by optimizing system resources. The three profiling modes enable you to create a statistical analysis of the performance of your application with respect to its functionality, both at compile time and runtime. You can also analyze your application's memory use in the Java heap, and the occurrence and duration of various events.
- **Extensible and Pluggable Integrated Development Environment:** JDeveloper provides a public Extension SDK, enabling its development environment to be extended and customized. Using this SDK, customers and partners can develop extensions and integrate them directly into the JDeveloper IDE. Developers can also share the technologies written with the Extension SDK at the JDeveloper Extensions Exchange at:

<http://otn.oracle.com/products/jdev/htdocs/partners/addins/exchange/content.html>

- **Support for SQL, PL/SQL, and XML:** In addition to Java, JDeveloper provides native support for SQL, PL/SQL, and XML. This support includes syntax highlighting and code insight, as well as PL/SQL development, PL/SQL debugging, and SQL tuning. Additionally, JDeveloper provides direct access to the database, allowing you to view, create, modify, and delete tables, views, triggers, indexes, sequences, and more.
- **Debugging Support:** JDeveloper offers robust debugging support for both Java and PL/SQL. Debugging in these two environments is seamlessly integrated when using an Oracle database, providing the ability to step from Java code

directly into PL/SQL code within the same debugging session. Other features include remote debugging, debugging multiple processes on multiple servers, tracking relevant data members in a smart data window, and a count of instantiations per class.

- **Business Intelligence Beans:** To allow the development of highly analytical business intelligence applications, JDeveloper has integrated Oracle Business Intelligence Beans (BI Beans) into its environment. BI Beans enable developers to build applications that take advantage of the new analytic capabilities available in the Oracle database. Using modular components and intuitive wizards, developers can build sophisticated BI applications quickly and easily.

Oracle Business Components for Java

Oracle Business Components for Java (BC4J) is a standards-based, server-side framework for creating scalable, high-performance J2EE applications. The framework provides design-time facilities and runtime services to simplify the task of building, debugging, customizing, and reusing business components. The applications developed with BC4J can be deployed on any J2EE platform and can be accessed from a wide variety of clients, including Web browsers, professional desktop clients, XML clients, and wireless devices.

The J2EE programming model provides developers with a set of core APIs and technologies for developing multi-tier applications that are scalable and portable. In addition to these APIs, developers often find themselves implementing design patterns to address some of the design issues that are common to most multi-tier applications. Understanding these design issues and coming up with solutions for them can be both challenging and time consuming. BC4J steps into this space by providing an application framework for J2EE developers that implements many common design patterns and provides out-of-the-box solutions for tasks like object-relational mapping, validation, data access, network traffic minimization, and user interface binding.

By eliminating the substantial coding and testing work related to these common development issues, BC4J lets application developers focus full-time on implementing business solutions. The benefits of this framework include reduced development cost, lower project risk, and shorter time-to-market.

Additionally, the BC4J framework provides flexibility by allowing application developers to override any of the automatic behaviors the framework provides. Using BC4J, developers can take advantage of built-in behavior while maintaining control over application functionality.

Oracle Business Components for Java Features

The following sections discuss key BC4J features.

Object-Relational Mapping

The Object-Relational (O/R) mapping facilities that BC4J provides make it easy to use Java objects to transparently store and retrieve data in a relational database. Wizards and UML modeling tools make it easy to reverse engineer database tables and views into Java objects. For developers who prefer to work with entity beans, you can also create business components mapped to EJB 2.0 local entity beans.

Validation Model

Enforcing business rules is a critical part of any application, and BC4J provides a framework for consistently developing and enforcing validation rules across your applications. With BC4J, rather than coding business logic into each client, the developer writes it once in a reusable, domain-specific business component. Also, BC4J allows you to perform validation at different levels, depending on the sophistication of your business rules.

Scalability and Performance

The BC4J framework provides features such as resource pooling, network optimization, and disk spillover that are designed to enhance an application's performance and scalability. Over the standard EJB remoting architecture, BC4J implements additional optimizations to keep the number of network roundtrips between the client tier and the application tier to a minimum. Further, for JSP and servlet-based applications deployed as J2EE Web modules, BC4J provides unique performance and scalability features. These include declarative JDBC connection pooling, application module pooling, session state persistence, and disk spillover for large result sets.

Flexible Deployment

BC4J presents a productive programming model for application development that is tier-independent and completely based on J2EE standards. The framework enforces a strict logical separation of the client tier and the business logic tier, making it possible to deploy the same application in multiple configurations without modifying the application code. BC4J supports the standard J2EE deployment modes: J2EE Web module, EJB session bean, and local mode. With BC4J you do not have to choose your deployment platform beforehand, and you can switch platforms easily without making changes to the application code.

Rich User Interface Support

BC4J makes it easy to bind JSP, Java Swing, wireless, and other XML clients to business components. The tier-independent programming model ensures that the same application can be bound to different user interfaces without changing any code in the business logic tier.

For Java client development, JDeveloper provides JClient, a common client framework for building Java applications and applets for business components. JClient uses the model-view-controller architecture to allow you to bind standard Swing controls or any third party model-based controls to BC4J data sources. JClient also provides a number of composite controls, making it easy to design Java clients with richer functionality than the standard Swing controls provide. Using the JClient design-time wizards, you can quickly generate data-aware forms, which you can then customize with the JDeveloper user interface design tools.

For JSP and servlet applications, JDeveloper provides the BC4J Data Tag Library, and set of JSP 1.1 compliant tags for binding JSP pages to BC4J data sources. The BC4J data tags make it easy to design and build JSP database applications with complete control of both the HTML design and the database access operations. When using the data tags, the developer does not have to write copious amounts of Java code to interact with BC4J data or customize the user interface. The client JSP application can mostly be comprised of HTML and JSP tags, giving JSP developers the flexibility to use their favorite visual HTML editors in conjunction with JDeveloper to design the exact appearance of their JSP applications.

Layered Customization Support

Once you have developed and delivered an enterprise application, the person or company installing the software typically needs to tailor it to fit their needs. BC4J offers a simple, innovative way to do this. All framework components have their Java implementation class cleanly separated from their XML component definition. Developers can extend both the Java class and the XML component definition to customize their applications. Further, the extended components can globally replace the original components across the application.

Tight Integration with JDeveloper

Oracle JDeveloper provides integrated design-time support for the BC4J framework. The wizards, editors, and other integrated design tools in JDeveloper make it easy to model, develop, test, deploy, and debug BC4J applications.

Integration with Oracle *interMedia*

In addition to standard object datatypes, BC4J now has the ability to bind to the four Oracle *interMedia* object types to provide multimedia retrieval and upload support. With this integration complete, you can use BC4J to access image, audio, video, and generic media content stored in Oracle *interMedia* databases, and deliver these digital media through UIX applications on Web browsers.

Oracle Application Server Web Services

To enable **e-businesses** to work effectively, the Internet needs to support a standards-based infrastructure that enables companies and their enterprise applications to communicate with other companies and their applications more efficiently. These standards should allow discrete business processes to expose and describe themselves on the Internet, allow other services to locate and invoke them, and provide a predictable response.

Web services drive this transformation by promising a fundamental change in the way businesses function and enterprise applications are developed and deployed. A Web service is a discrete business process that does the following:

- **Exposes and describes itself:** A Web service defines its functionality and attributes so that other applications can understand it. A Web service makes this functionality available to other applications.
- **Allows other services to locate it on the Web:** A Web service can be registered in an electronic Yellow Pages, so that applications can easily locate it.
- **Can be invoked:** Once a Web service has been located and examined, the remote application can invoke the service using an Internet standard protocol.
- **Returns a response:** When a Web service is invoked, the results are passed back to the requesting application over the same Internet standard protocol used to invoke the service.

Web services provide a standards-based infrastructure through which any business can do the following:

- Offer appropriate internal business processes as value-added services that can be used by other organizations
- Integrate its internal business processes and dynamically link them with those of its business partners

Oracle Application Server Web Services Features

Oracle Application Server Web Services provides advanced runtime features and comprehensive support for developing and deploying Web services.

Oracle Application Server Web Services Development Features

Key Oracle Application Server Web Services development features include:

- **Development Environment:** Oracle Application Server Web Services allows application developers to implement Web services using J2EE components. In addition, you can use Java classes or PL/SQL stored procedures to implement Web services. Web services inherit all the runtime and lifecycle management elements of J2EE applications.
- **Development Tools and Wizards:** Oracle Application Server Web Services developers can use the same set of command line utilities to create, package, and deploy Web services that they use for other Oracle Application Server Containers for J2EE (OC4J) applications. In addition, Oracle Application Server Web Services provides the Web Service HTML/XML Streams Processing Wizard that assists developers in creating an EJB whose methods access and process XML or HTML streams.
- **Automatically Generates WSDL:** Oracle Application Server Web Services can generate Web Services Description Language (WSDL) and client-side proxy stubs. This generation occurs when the Web service is assembled using the Web Services Assembly tool or, for a deployed Web service, the first time the WSDL or the client-side stubs are requested. After the first request, the previously generated WSDL or client-side proxy stubs are sent when requested.
- **Registration, Publishing, and Discovery:** Oracle Application Server Web Services provides a standards-compliant UDDI registry where Web services can be published and discovered. The Oracle UDDI registry supports both a private and public UDDI registry and can also synchronize information with other UDDI nodes.
- **Developer Simplicity:** Using Oracle Application Server Web Services, developers do not need to learn a completely new set of concepts. Web services are developed, deployed, and managed using the same programming concepts and tools as J2EE applications.
- **Business Logic Reuse:** Application developers can transparently publish their J2EE applications to new Web services clients with no change in the application itself. The existing business logic developed in J2EE can be transparently accessed from existing J2EE/EJB clients.

- **Common Runtime Services:** Oracle Application Server has a common runtime and brokering environment for J2EE applications and Web services. As a result, Web services transparently inherit various services available with the J2EE container, including transaction management, messaging, naming, logging, and security services.

Oracle Application Server Web Services Deployment and Management Features

Key Oracle Application Server Web Services deployment and management features include:

- **Packaging and Assembly:** The Web Services Assembly Tool assists with assembling Web services and producing a J2EE `.ear` file.
- **Deployment:** Oracle Enterprise Manager provides a comprehensive set of facilities for deploying Web services to Oracle Application Server through Oracle Enterprise Manager Application Server Control. Application Server Control provides a single, consistent *Deploy Applications* wizard for deploying Web services to Oracle Application Server. It accepts a J2EE `.ear` file, and takes you through the steps to specify information about the application you are deploying, and then deploys the application.
- **Register Web Service:** The *Deploy Applications* wizard also provides access to facilities for registering Web services in the UDDI registry.
- **Browse the UDDI Registry:** The Oracle UDDI registry provides the UDDI standards-compliant, pre-defined, hierarchical categorization schemes. Oracle Enterprise Manager can drill down through these categories and look up specific Web services registered in any category.
- **Monitoring and Administration:** Once deployed, Oracle Enterprise Manager provides facilities to deinstall a Web service, and also to monitor Web service performance as measured by response time and throughput, and to monitor status as measured by up time, CPU consumption, and memory consumption. Oracle Enterprise Manager also provides facilities to identify and list all of the Web services deployed to a specific Oracle Application Server instance.

Oracle Application Server Web Services Architecture

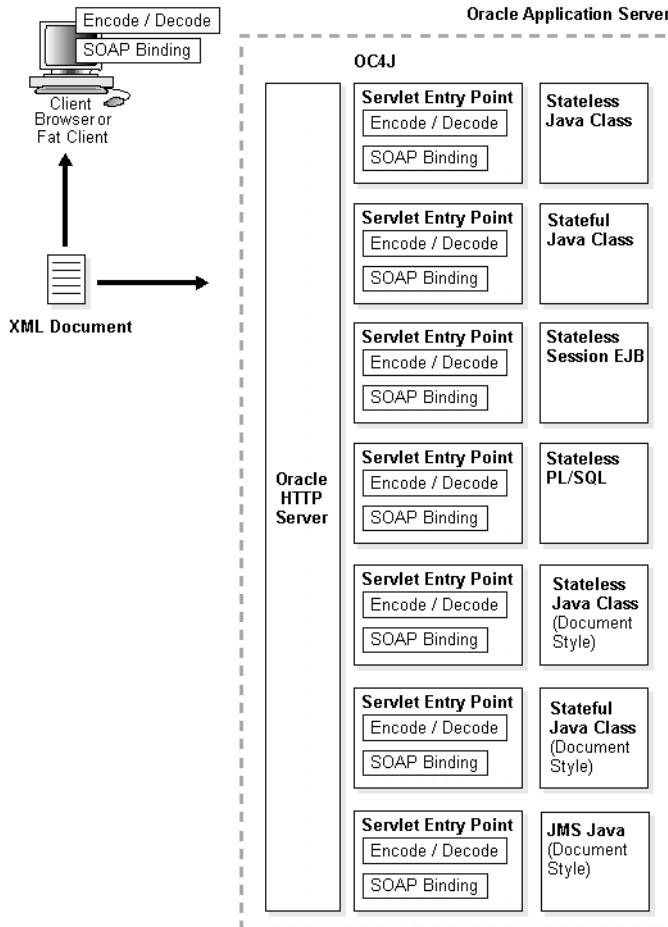
Oracle Application Server Web Services run as **servlets** in the **OC4J servlet container**. This gives Web services the same **scalability**, **availability**, and load balancing facilities that all **J2EE** applications have in Oracle Application Server.

See Also: "Oracle Application Server Containers for J2EE Containers" on page 2-13

Oracle Application Server Web Services support both Remote Procedure Call (RPC) style exchange and message oriented, or Document Style, exchange. Supported RPC Web services include Java classes, stateless session EJBs, and stateless PL/SQL stored procedures. Supported Document Style Web services include Java Class Document Style Web Services and JMS Document Style Web Services.

Oracle Application Server Web Services use a different J2EE standards-compliant servlet for each implementation type to provide an entry point into the Web services implementation. Figure 2-10 illustrates the Oracle Application Server Web Services runtime architecture, including the servlet entry points.

Figure 2–10 Oracle Application Server Web Services Architecture



For a detailed discussion of the Oracle Application Server Web Services architecture, see the *Oracle Application Server Web Services Developer's Guide*.

Web Services Framework

The following Internet standards provide the Oracle Application Server Web Services framework:

- **Simple Object Access Protocol (SOAP)** is a lightweight, XML-based protocol for exchanging information in a decentralized, distributed environment. SOAP supports different styles of information exchange, including Remote Procedure Call style (RPC) and **message-oriented** exchange. RPC style information exchange allows for request-response processing, where an endpoint receives a procedure-oriented message and replies with a correlated response message. Message-oriented information exchange supports organizations and applications that need to exchange business or other types of documents where a message is sent but the sender may not expect or wait for an immediate response.
- **Web Services Description Language (WSDL)** is an XML format for describing network services containing RPC-oriented and message-oriented information. Programmers or automated development tools can create WSDL files to describe a service and can make the description available over the Internet. Client-side programmers and development tools can use published WSDL descriptions to obtain information about available Web services and to build and create proxies or program templates that access available services.
- **Universal Description, Discover, and Integration (UDDI)** is an online electronic registry that serves as electronic Yellow Pages, providing an information structure where various business entities register themselves and the services they offer through their WSDL definitions. There are two types of UDDI registries, public UDDI registries that serve as aggregation points for a variety of businesses to publish their services, and private UDDI registries that serve a similar role within organizations.

Oracle Application Server Forms Services

Oracle Forms applications combine interactive graphical interfaces with strong support for data validation. Forms developers can quickly create applications with powerful data manipulation features.

An Oracle Forms application gives you the power of a desktop application deployed in a browser. It provides a very rich and productive user interface by allowing the following:

- Immediate data validation as **clients** enter data into the form instead of after submitting the form
- Automatic completion and list of value searches for fields that enable users to enter correct information quickly

An Oracle Forms application is analogous to a traditional **HTML** form application. In the traditional application, the user enters data into a form on a Web page and submits the data. The Web **server** processes the data from the form in a **CGI** application. The application then performs any necessary **transactions** and returns feedback to the user's browser. JavaScript and the database perform data validation tasks to ensure that the data from the form is correct and complete.

Oracle Application Server Forms Services deploys Forms applications to **Java** clients in a Web environment. Oracle Application Server Forms Services automatically optimizes class downloads, network traffic, and interactions with the Oracle database. Applications are automatically load-balanced across multiple servers and, therefore, can easily scale to service any number of requests.

Oracle Application Server Forms Services consists of four components:

- **Forms Servlet:** The Forms Servlet is used to establish the initial connection when a user starts a Forms application. It returns to the client the HTML file that contains the Forms Client applet.
- **Forms Client Applet:** A Java applet running in the client browser. It provides the user interface for the corresponding Forms Runtime Process.
- **Forms Listener Servlet:** A **servlet** running in the **OC4J servlet container**. It manages the lifecycle of the Forms Runtime Process and network connections to the runtime process.
- **Forms Runtime Process:** A process running in an Oracle Application Server **instance**. The process maintains the connection between the client applet and the database.

Oracle Application Server Forms Services Features

Oracle Application Server Forms Services provides the following features to allow a robust and standard deployment of Forms applications on the Internet.

Automatic Internet Scalability and High Performance

Oracle Application Server Forms Services is an optimized framework for deploying Oracle Forms applications to a multi-tiered environment. OracleAS Forms Services delivers out-of-the-box functionality and native services to ensure that Forms applications automatically scale and perform over any network. OracleAS Forms Services allows organizations to re-deploy existing applications to the Internet without changing the application code.

Native Services and Out-of-the-Box Functionality With OracleAS Forms Services, the scalability and high performance of Internet deployment do not rely on the quality of the application or on the developers' skills; applications developed with Oracle Forms automatically scale and perform through the use of built-in services that Oracle Forms provides. These services natively deliver the building blocks and the infrastructure of an e-business solution. Business developers do not have to implement complex, low-level infrastructure code and therefore can focus on implementing business requirements. Examples of these native services include record caching and record locking, database management, exception handling, and load balancing.

Application Server Integration As a fully-integrated component of Oracle Application Server, OracleAS Forms Services utilizes the Oracle Application Server Infrastructure. The deployment architecture of OracleAS Forms Services uses Oracle Application Server Containers for J2EE (OC4J). Codeless integration with Oracle Application Server Single Sign-On Server eliminates the need for users to remember multiple passwords. And leveraging the power of Oracle Enterprise Manager Application Server Control means that system administrators can use a single point of control to start, shutdown, monitor, trace, and diagnose OracleAS Forms Services runtime processes.

Network Traffic Optimizations OracleAS Forms Services is highly optimized to reduce network traffic and to minimize database response time. For example, OracleAS Forms Services renders application interfaces as highly-optimized generic Java applets and uses mechanisms such as message diffing, message bundling, and JAR file caching to reduce the number of round trips between the client and the middle tier. At the same time, OracleAS Forms Services leverages the capabilities of the Oracle database to minimize interactions with the database server.

HTTP/HTTPS Support The deployment with OracleAS Forms Services is optimized for corporate intranets, extranets, and for the Internet using socket connections or HTTP/HTTPS communications between the client and the middle tier. The support for HTTP/HTTPS allows Oracle Forms applications to be deployed across industry firewalls in a highly secure and optimized fashion.

Load Management Ensuring that your application has no single point of failure is essential to guaranteeing your application is highly available. The OracleAS Forms Services runtime architecture is based on standard servlets, meaning you can use standard load balancing techniques such as hardware load balancing, reverse proxy servers, and standard load balancing through Oracle Application Server.

Open Access to Internet Standards

Through code-based integration, OracleAS Forms Services gives you the flexibility to leverage emerging standards, including Java and XML, within your Oracle Forms applications. This gives you the advantages of both the application framework and built-in scalability and performance of OracleAS Forms Services, along with the flexibility to leverage Java and XML to extend your applications.

Server-Side Openness OracleAS Forms Services gives you the ability to access components written in Java and make use of them from Oracle Forms applications on the middle tier. Using this feature, OracleAS Forms Services is able to communicate and integrate with other applications or technologies, such as Web services, XML, Enterprise JavaBeans, and non-standard devices.

Client-Side Openness OracleAS Forms Services renders the application presentation as a generic and optimized Java applet. This applet can be extended using Java components (such as JavaBeans and Pluggable Java Components) so that business developers can quickly and easily deliver sophisticated interfaces that maximize end user productivity. The Java client extensibility, along with the new look and feel, leverages the strength of Java without the need for business developers to learn the language.

Oracle Application Server Forms Services Architecture

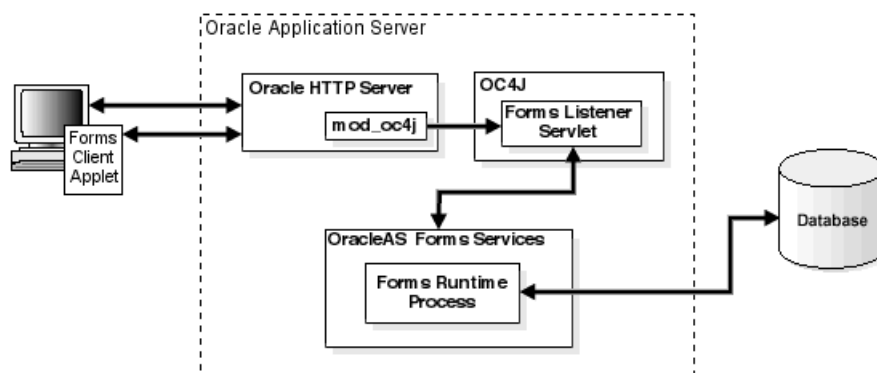
When a client requests an Oracle Application Server Forms application, the request has the following flow:

1. The user launches a browser and navigates to the URL for the Forms application.
2. The HTTP Listener interprets the URL and displays an HTML page containing an `<APPLET>` tag that describes the Forms Java Client to the browser. The URL calls the Forms servlet to create the HTML page dynamically.
3. The client receives the HTML file served by the HTTP Listener. The HTML file supplies the information required to locate the Java class files that make up the Forms Java Client. The HTML file also contains information about the requested Forms application, as well as parameters that get passed to the Forms session, such as login information.
4. The browser downloads the Forms Java Client and the Java applet starts.
5. The Forms Java Client applet sends a request to start a Forms session through the HTTP Listener to the Forms Listener Servlet.

6. After receiving the connection request from the Forms Java Client, the Forms Listener Servlet starts a new Forms Runtime process for the client.
7. The Forms Runtime process allocated to the client loads the module specified in the HTML file and any libraries and menus that are required by that form. All communication between the Forms Java Client and the Forms Runtime process is passed through the Forms Listener Servlet.
8. The user is prompted for database login information, and the connection to the database server is established.
9. The user is now ready to work with the Forms application.

Figure 2–11 illustrates this flow in terms of the Oracle Application Server Forms Services architecture.

Figure 2–11 Oracle Application Server Forms Services Request Flow



Oracle XML Developer Kit

XML makes data portable and interoperable across heterogeneous systems. XML is a metamarkup language that supports markup tags that have been defined by the user to encapsulate and describe data in a Web page. Because it is a self-defined language, it is extremely flexible.

Oracle Application Server includes the **Oracle XML Developer Kit (XDK)** to support development of any applications that use XML. The XDK provides basic XML infrastructure components for manipulating and transforming XML documents. The XDK contains component libraries and utilities that are based on World Wide Web Consortium (W3C) specifications. You can use these components

to generate, manipulate, render, and store XML-formatted data in an Oracle database or to share data between applications written in diverse programming languages.

Oracle XML Developer Kit Tools

The following sections describe the tools included in the Oracle XML Developer Kit.

XML and XSLT Parsers

The XML parser APIs are defined by the W3C specifications so that application developers can use standard programming interfaces. However, if some functionality is not specified, then Oracle Corporation may implement its own enhancements to the specifications.

The Oracle XML and XSLT parsers provide international **character set** and multithreaded support, allow optional validation, and cache document type definitions (DTDs) and stylesheets for performance.

Oracle XML Developer Kit provides the following APIs:

- **Document Object Model (DOM) APIs:** Because an XML document is structured, in the sense that "start" tags have corresponding "end" tags and these tags are nested in an ordered fashion, an XML document can be viewed as a tree whose hosts consist of these tags and information between and corresponding to the tags. When the DOM APIs are used to navigate an XML document, the XML parser parses the document and forms a tree representation of it in memory.
- **Simple APIs for XML (SAX) APIs:** SAX APIs are event-based, meaning that notification of certain events and data encountered during the parsing of an XML document can be reported with callback functions to the application program. One advantage over DOM is that an in-memory representation of the parse tree does not have to be built, thus saving memory and improving performance. When application programs are notified of these events, then they must handle them.
- **Namespace APIs:** These interfaces resolve namespace prefixes that are used to qualify element and attribute names in XML documents. XML document namespaces are identified by uniform resource identifier (URI) references that qualify element or attribute names and locate resources that may be on different machines or in different XML documents. Namespaces make it possible to have identical names for elements and attributes by qualifying them with URIs that differentiate the names.

- **Parser APIs:** Application programmers invoke the parser API to read an XML document and to provide access to its content and structure through DOM or SAX APIs. Typically, initialization and termination functions must also be invoked in association with the parse function. Various flags, such as to discard whitespace and turn on validation, can be set with initialization functions before the application program invokes the parse function.
- **XSLT APIs:** These interfaces read the input stylesheet file and transform the input XML document according to the stylesheet. A command-line interface to the integrated XSLT processor is provided. This allows application programmers to specify the number of threads to use to parse and transform XML documents and other useful options. To enhance performance, the XSLT engines allow **caching** of both stylesheets and Document Type Definitions (DTDs) so they can be reused for multiple XML documents.

XML Schema Processors

Oracle XML schema processors comply with the Structures and Datatypes sections of the W3C XML Schema Working Drafts, except for certain features (such as unique, key and keyref constraints) when the SAX parser is used. XML schemas are a superset of DTDs, except for the support of primitive and complex datatypes in XML schemas. This allows you to validate XML documents, with embedded datatype information, against XML schemas.

XML Class Generators

The XML class generators create a set of **Java** or **C++** classes that create XML documents corresponding to a DTD or to XML schema definitions. This is useful when an application wants to send an XML **message** to another application based on an agreed-upon DTD, or as the back end of a Web form to construct XML documents. You can use the generated classes to programmatically construct XML documents that comply with the DTD or schema definition.

XSQL Servlet

XSQL Servlet is a tool that processes **SQL** queries and outputs the result set as an XML document. This processor uses an XML file with embedded SQL queries as its input. You can use XSQL Servlet to perform the following tasks:

- Build dynamic XML datapages from the results of one or more SQL queries and serve the results over the Web as XML datagrams or **HTML** pages using server-side XSLT transformations
- Receive XML posted to your Web **server** and insert it into your database

XML Transviewer Beans

XML transviewer beans are a set of XML components that constitute XML for Java beans. These are used by Java applications or applets to view and transform XML documents.

These beans are visual and non-visual Java components that are integrated into Oracle JDeveloper to enable the fast creation and deployment of XML-based database applications.

Oracle Application Server PL/SQL Platform

Oracle Application Server provides support for building **PL/SQL**-based applications on the Web. **PL/SQL stored procedures** retrieve data from a database and generate **HTTP** responses containing data and code to display in a Web browser. There are many techniques for coding dynamic pages, but **PL/SQL** is particularly suited for producing dynamic pages based on database processing. Its support for **DML** statements, dynamic **SQL**, cursors, and tight server integration provide both power and flexibility for Web applications. The dynamic pages can contain links and **HTML** forms that call more stored procedures, for example to drill down on the displayed data. The set of interlinked **HTML** pages forms the user interface of the Web application. Producing dynamic content using **PL/SQL** stored procedures gives you the flexibility and interactive behavior of **CGI** programs, without the memory overhead of starting a new **CGI** process each time.

Oracle Application Server PL/SQL Tools

Oracle Application Server **PL/SQL** developers can create applications using either Oracle **PL/SQL** Server Pages or Oracle **PL/SQL** Web Toolkit.

mod_plsql

Mod_plsql is an Apache plug-in that provides support for building **PL/SQL**-based applications on the Web. It maps browser requests into **PL/SQL** stored procedures, which retrieve data from a database and generate **HTTP** responses containing data and code to display in a Web browser.

Oracle PL/SQL Server Pages

To include dynamic content inside Web pages, including the results of **SQL** queries, you can use server-side scripting through **PL/SQL** Server Pages (**PSPs**). You can author the Web pages in a script-friendly **HTML** authoring tool and drop the pieces

of PL/SQL code into place. You may find this technique more convenient than using the HTP and HTF **packages** to write out HTML content line by line.

Because the processing is done on the database server rather than the Web server, the browser receives a plain HTML page with no special script tags, and you can support all browsers and browser levels equally. It also makes network traffic efficient by minimizing the number of server roundtrips.

Embedding the PL/SQL code in the HTML page that you create lets you write content quickly and follow a rapid, iterative development process. You maintain central control of the software, with only a Web browser required on the **client** machine.

Oracle PL/SQL Web Toolkit

The PL/SQL Web Toolkit contains PL/SQL packages that allow you to create applications that generate dynamic HTML. Using the toolkit, you can access data and insert it into Web pages. The packages remove the need for you to know the specifics of HTML syntax and allow you to focus on writing applications.

Oracle Application Server PL/SQL Architecture

Oracle Application Server PL/SQL leverages the `mod_plsql` module to deliver Web content, as follows:

- Visiting a Web page, following a hypertext link, or submitting an **HTML** form causes the database server to run a stored procedure.
- Any choices that a user makes on an HTML form are passed as parameters to the stored procedure. Parameters can also be hardcoded in the **URL** used to invoke the stored procedure.
- The results of the stored procedure are printed as tagged HTML text and are displayed in the browser as a Web page. Web pages generated this way are dynamic: code runs inside the database server, producing HTML that varies depending on the database contents and the input parameters.

This kind of dynamic content is different from dynamic HTML (DHTML). With DHTML, the code is downloaded as JavaScript or some other scripting language, and processed by the browser along with the HTML. A PL/SQL Web application can print JavaScript or other script code in its output to produce complex DHTML that would be tedious to produce manually.

Oracle Content Management Software Developer Kit

Oracle Application Server includes the Content Management Software Development Kit (CM SDK), a robust document lifecycle system that is built in Java and integrated with other Oracle content management API products. Oracle customers and partners have built successful solutions on this Java-based platform that leverages the features and capabilities of the Oracle platform.

Oracle Content Management SDK Features

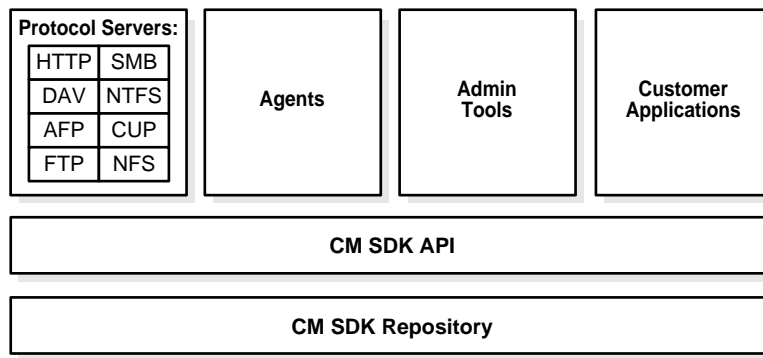
Important features and capabilities of the CM SDK include the following:

- Integration with Oracle Text, Oracle Workflow, Oracle Ultra Search, Oracle *interMedia*, and OracleAS Portal
- Complete document lifecycle API with a variety of options for security, check-in/check-out, versioning, searching, extensible metadata, and other standard content management operations
- Easy extensibility to help you develop and deploy your content management solutions faster
- Managed through Oracle Enterprise Manager Application Server Control
- Synchronous and asynchronous trigger mechanisms to enforce business rules
- Built-in support for standard protocols such as HTTP/WebDAV, SMB/NTFS, FTP, AFP, NFS, IMAP4, and SMTP
- Support for standard Oracle performance, scalability, and high availability features

Oracle Content Management SDK Architecture

Oracle Content Management SDK provides, out of the box, the content and metadata repository, protocol servers, and class libraries that allow developers to quickly create content management solutions.

Figure 2-12 shows the basic architecture of Oracle Content Management SDK. Once installed, developers can immediately start storing content into the repository via the Web DAV, SMP, FTP, or IMAP interface. They can then provide the business logic for their specific requirements and create a Web-based user interface. Also included with the CM SDK installation is a Web starter application that demonstrates how to create Web-based content management applications.

Figure 2–12 Oracle Content Management SDK Architecture

Oracle Application Server MapViewer

The Oracle Application Server MapViewer is a programmable tool for rendering maps using spatial data managed by Oracle Spatial or Oracle Locator. MapViewer provides tools that hide the complexity of spatial data queries and cartographic rendering, while providing customizable options for more advanced users. These tools can be deployed in a platform-independent manner and are designed to integrate with map-rendering applications.

There are five concepts that are fundamental to understanding MapViewer:

- **Style:** A style defines the rendering properties for features that are associated with styles. For example, a text style determines how a feature is labeled on a map, while a line style determines the rendition of a linear feature such as a road.
- **Theme:** A theme is a collection of features (entities with spatial and nonspatial attributes) that are associated with styles through the use of styling rules.
- **Base map:** A base map consists of one or more themes.
- **Mapping metadata:** Mapping metadata consists of a repository of styles, themes, and base maps stored in a database.
- **Map:** A map is one of the components that MapViewer creates in response to a map request. The map can be an image file, the object representation of an image file, or a URL referring to an image file.

When an application uses MapViewer, it applies specific styles, such as colors and patterns, to specific themes, such as cities, rivers, and highways, to render a map.

For example, the application might display a map in which state parks appear in green and restaurants are marked by red stars. A map typically has several themes representing political or physical entities, or both. For example, a map might show national and state boundaries, cities, mountain ranges, rivers, and historic sites. When the map is rendered, each theme represents a layer in the complete image.

MapViewer lets you define styles, themes, and base maps, including the rules for applying one or more styles to each theme. These styles, themes, base maps, and associated rules are stored in the database in map definition tables under the MDSYS schema, and they are visible through metadata views. All styles in a database instance are shared by all users. The mapping metadata that you can access is determined by MapViewer metadata views. The set of map definition objects that a given user can access is sometimes called that user's mapping profile.

Oracle Application Server MapViewer Features

MapViewer includes the following main components:

- A rendering engine (Java class library) that provides cartographic capabilities (a map renderer)
- An XML API that provides a programmable interface to MapViewer

The rendering engine connects to the Oracle database through Java Database Connectivity (JDBC). It also loads the map metadata from the database, and applies it to the retrieved spatial data.

The XML API provides high-level application developers with a convenient interface for submitting a map request to the middle-tier MapViewer and handling its map response.

Oracle Application Server MapViewer Architecture

With MapViewer, the basic flow of action involves two steps, regardless of whether the client requests a map or a MapViewer administrative action.

For a map request, the following occurs:

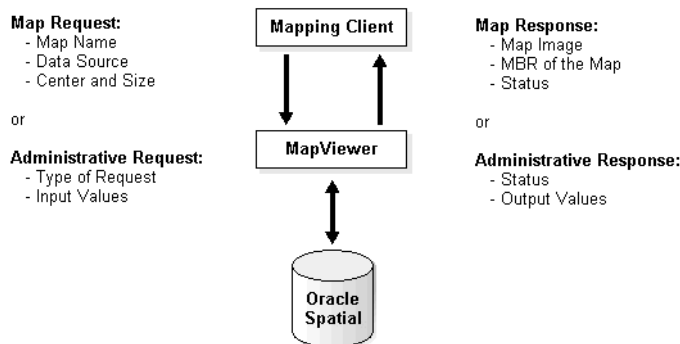
1. The client requests a map, passing in the map name, data source, center location, map size, and optional other data to be plotted on top of a map.
2. The server returns the map image (or a URL for the image), the minimum bounding rectangle (MBR) of the map, and the status of the request.

For a MapViewer administrative request, the following occurs:

1. The client requests a MapViewer administrative action, passing in the specific type of request and appropriate input values.
2. The server returns the status of the request and the requested information.

Figure 2–13 shows the MapViewer architecture, which supports this flow of action.

Figure 2–13 MapViewer Architecture



As shown in Figure 2–13:

- MapViewer is part of the Oracle Application Server middle tier.
- MapViewer includes a rendering engine.
- MapViewer can communicate with a client Web browser or application using the HTTP protocol.
- MapViewer performs spatial data access (reading and writing Oracle Spatial and Locator data) through JDBC calls to the database.
- The database includes Oracle Spatial or Locator, as well as mapping metadata.

Portal Applications

This chapter provides an overview of **Oracle Application Server Portal** features and benefits. The topics include:

- Introduction to Oracle Application Server Portal
- Oracle Application Server Portal Features
- Application Access and Integration
- Oracle Application Server Portal Architecture

Introduction to Oracle Application Server Portal

Portals allow **clients** to access information through a Web browser. This information usually comes from different data sources that the portal makes available through a single entry point. That entry point is known as a page.

Portals also support personalized views, so that each user or user group can customize both the content and the appearance of the portal to suit individual preferences and requirements.

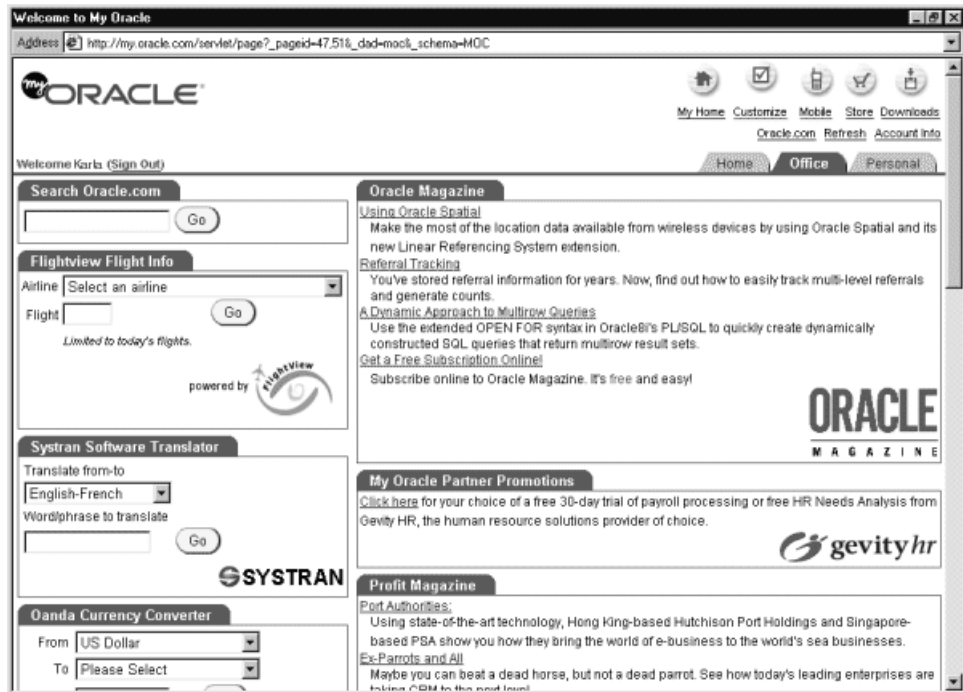
For example, a financial analyst's page would likely include information from real-time Internet-based stock quotes, financial reports from an online repository, and access to legacy financial accounting and banking systems. The data from these systems are independent of each other, but the portal allows them to exist within a single page.

What is Oracle Application Server Portal?

Oracle Application Server Portal is a Web-based tool for building and deploying **e-business portals**. It provides a secure, manageable environment for accessing and interacting with enterprise software services and information resources. A portal

page makes data from multiple sources accessible from a single location. Figure 3-1 shows a sample portal page from <http://my.oracle.com>. Each one of the tabbed areas within the Office tab contains information from a different data source.

Figure 3-1 Sample Portal Page



E-Business Support with Oracle Application Server Portal

The growth of the Internet and Internet technologies creates new opportunities and challenges for businesses. The earlier vision of free-flowing information shared over corporate intranets is meeting roadblocks because of information overload, system management complexities, and conflicting data. Shortcomings in technology and its implementation are limiting the impact of a complete corporate intranet on user's core job functions. As a result, much of the potential revenue, productivity, and efficiency gains are not fully realized.

In response, organizations are turning to portals as key components of their e-business strategy. Portals are emerging as essential problem-solving mechanisms that provide a single source of interaction with all corporate information and the

focal point for conducting day-to-day business. Companies are using portals and supporting applications as follows:

- To provide access to instantly available, personalized, job-based information
- In forums for groups to exchange, analyze, and discuss ideas
- To automate business processes that integrate into daily activities

OracleAS Portal supports e-businesses by doing the following:

- **Providing secure access to existing information, no matter where it is:** Portal developers can organize and structure content in a consistent and logical way. This reduces or eliminates the need for users to hunt for information through a variety of sources.
- **Supporting personalized views:** Users and user communities can organize the information they access in ways that complement their work habits or interactions.
- **Providing self-service to users:** Employees, partners, and suppliers who have important information to share with the community can do so without specific technical skills or help from a technology expert.
- **Enabling single sign-on:** Users can log into the portal once per session and access all internal and external applications without logging into any of them.

Oracle Application Server Portal Features

Key features of Oracle Application Server Portal include the following:

- **An extensible framework:** OracleAS Portal has an extensible framework that integrates Web-based resources such as Web pages, applications, business intelligence reports, and syndicated content feeds within standardized, reusable information components called **portlets**. Within a portlet, these resources are personalized and managed as a service of OracleAS Portal. Companies can create their own portlets for their existing Web resources and can select additional portlets from the growing catalog of third-party portlet providers. Additionally, Oracle Application Server provides many ready-made portlets out-of-the-box that are available for immediate use. The **portal** framework provides additional services including single sign-on, content classification, enterprise search, directory integration, and access control.
- **An easy-to-use, personalized interface:** The OracleAS Portal interface provides an organized, consistent view of the business information, Web content, and applications that each user needs. Portal administrators and designers use a

browser-based control panel to selectively grant access to applications and information by making portlets available only to specific users or user groups.

- **Self-service publishing:** OracleAS Portal self-service publishing features allow authorized users to post and share any kind of document or Web content with other users anywhere in the world. Content contributors use controls for uploading documents, implementing version control, customizing page formatting and display, limiting access, and managing content without any requirement for technical expertise or **HTML** knowledge.
- **A scalable deployment architecture:** The OracleAS Portal architecture is easily configured for departmental, regional, and enterprise-wide deployment. The deployment model supports a variety of configurations, including single **host** and multi-tier, on a broad set of hardware platforms and operating systems.
- **Integration with Oracle Application Server Web Cache:** OracleAS Web Cache provides caching, compression, and assembly features to accelerate the delivery of both static and dynamically-generated Portal content.

Portal Page Creation, Management, and Customization

OracleAS Portal incorporates a portal creation and deployment framework. The framework defines Web information sources as information components, and assembles these components within a portal page. It also supports customization of the Web page to one or more user communities.

Each portal page is divided into either item regions or portlet regions. Item regions allow you to add text, images, and files to a portal page. Portlet regions provide an area where you can place one or more portlets.

A portlet is an **HTML** or **XML** area that summarizes, promotes, or provides basic access to an information resource. The information resources can take on many forms and can serve many purposes.

Items and portlets are the fundamental building blocks of an OracleAS Portal page. Page owners create their own portal pages, and page owners can either maintain their own pages or delegate maintenance responsibilities to other users. Each portal page consists of content presented through one or more items and portlets and links that allow the **client** to navigate to another page or take some action.

Portal Content Publishing and Management

OracleAS Portal provides an integrated set of features for self-service document publishing, file upload, page formatting, and access control. Collaborators and

content publishers no longer need specific technical skills or a Webmaster to publish their content. Instead, they can use an item region on a portal page to publish their content and format content appearance using simple controls.

An item region includes built-in features for publishing, organizing, classifying, cross-referencing, and displaying the content it manages. Key components that make up an item region include:

- **Items that are the pieces of content themselves:** Items are defined by the base content that makes up the item and one or more attributes that describe the item. Item publishers create items by completing a series of steps in a wizard. Additional features, such as item version control, check-in and check-out, expiration, and automatic indexing support collaborative document creation and ease content management tasks.
- **Styles that define how items are displayed:** Item region style properties govern the colors, font properties, size, background images, banners, and other graphical elements for the items and navigation bars in an item region.

Item regions also include components that assist users in navigating or locating content of interest and allow for classifying content with categories and perspectives.

OracleAS Portal provides categories and perspectives as a means of applying a classification to the content (items and portlets) you add to your portal. Categories and perspectives can be used to locate content during searches. Categories are used to describe the type of content you are adding. For example, if you are creating a Human Resources page, you might have categories such as Benefits, Policies, and Payroll. Perspectives are used to describe the type of audience who might be interested in the content, such as managers, supervisors, and non-exempt employees.

Content Searching

OracleAS Portal searches can quickly and easily locate information managed within a portal page. OracleAS Portal supports the following search methods:

- **Basic searches** only match attributes of items in the current portal page against the search criteria. A basic search compares the search criteria to the name, author, description, and keywords of all items within the selected page group.
- **Advanced searches** allow the user to define or restrict the search by specific attributes. Users can specify that their searches do the following:
 - Return matches to any or all search terms

- Search within a specific portal page or across all portal pages
- Restrict searches to a particular category, perspective, item type, or attribute
- **Oracle Ultra Search searches** perform comprehensive searches against portal-managed information. The Ultra Search engine searches through all content that has been indexed using Ultra Search, not just portal metadata. These searches can also return matches based on near matches (terms that appear close together), soundex matches (terms that sound like the search term), and fuzzy matches (terms with similar spellings to the search term).

A search result portlet displays the matches of any search. You can add this portlet to any portal page and customize which attributes it displays.

Content Syndication

Content syndication is the aggregation, exchange, and distribution of information from content providers to syndicators to **subscribers**. The content providers deliver the content, the syndicators send the content, and the subscribers use the content. A content subscriber acquires a content catalog of **subscription** offers from a content syndicator and selects the desired subscription offers. When subscribing to content, the subscriber can choose to search for new information or have it automatically provided to them either when it is updated, or after a specified time interval.

Oracle Application Server Syndication Services delivers any database, legacy file system, or Internet content to any Internet subscriber, and automatically provides any content updates using any standards across any network. This simplifies the process of syndication or automated content exchange. Oracle Application Server Syndication Services provides a comprehensive solution for content aggregation, syndication, and distribution by letting you make available any or all of your content.

Content syndicators can use Oracle Application Server Syndication Services with the following benefits:

- Aggregate content from databases, Web sites, enterprise applications, e-mail repositories, and legacy systems
- Personalize the content to be delivered based on the profiles of their subscribers and deliver content updates based on the delivery policies associated with each subscription
- Automate content delivery by sending content to subscribers when information relevant to them changes, or schedule a content delivery when establishing a

subscription with subscribers, and deliver it to subscribers over multiple communication channels

- Transform any content from any formatted source to a markup language suitable for the subscriber

See Also: *Oracle Application Server Syndication Services Developer's and Administrator's Guide*

Portals and Wireless Devices

In addition to standard Web browsers, wireless clients can also access OracleAS Portal pages. Working with **Oracle Application Server Wireless**, the portal automatically transforms the portal page structure to a format appropriate for the smaller screens of most wireless devices. Only portlets generating Oracle Application Server Wireless XML content display on the wireless device.

OracleAS Portal developers also have access to a set of page design tools that help in creating portal pages that optimize the wireless experience. With these tools, developers can build a distinct portal structure for their wireless users. The wireless pages and portal pages can share portlet instances. This allows clients to reuse portlets on browser and wireless clients without reconfiguring each portlet.

Portal Integration with Oracle Application Server Single Sign-On

Oracle Application Server Portal leverages Oracle Application Server Single Sign-On (SSO) to provide single sign-on capabilities for secure access to portal content and applications. SSO works as a single, unified authentication service to all Oracle Application Server components, applications, and Web pages, storing user information and authenticating users against Oracle Internet Directory.

See Also: Chapter 11, "Security and Identity Management"

Application Access and Integration

Portal clients access **Oracle Application Server Portal** applications through **portlets**. Clients can select the portlets that appear on their page from a list of providers registered with OracleAS Portal. Additionally, developers can use the Oracle Application Server Portal Developer Kit to create their own portlets.

Integrating with Portlet Providers

Applications and information sources, represented as portlets, communicate with the **portal** through a provider. Each portlet only has one provider, and a provider can have one or more portlets that expose an underlying application or information source.

All portlets from portlet providers make use of Oracle Application Server Single Sign-On, regardless of their location. It is not necessary for portlets to be deployed within Oracle Application Server or even on the same hardware. This ensures that only authorized users are able to subscribe to a particular portlet and that authorized users can access all registered portlets by logging into their main portal page.

Oracle Application Server Portlets

Some Oracle Application Server components act as portlet providers to OracleAS Portal. This allows you to easily integrate information from various Oracle Application Server components into a single portal page.

Oracle Application Server Reports Services OracleAS Portal includes a simple report building facility. However, as your reports become more complex, you can import the report into **Oracle Application Server Reports Services** reports. You can deploy any Oracle Application Server Reports Services report as a portlet.

See Also: Chapter 5, "Business Intelligence"

Oracle Application Server Discoverer As a portlet provider, **Oracle Application Server Discoverer** offers worksheet portlets and list of workbooks portlets to OracleAS Portal users. A worksheet portlet contains information from a single Discoverer worksheet. The portlet displays this information in either a table, a graph, or both. The list of workbooks portlet presents a list of available workbooks.

See Also: Chapter 5, "Business Intelligence"

Oracle Content Management Software Development Kit The Oracle Content Management Software Development Kit (Oracle CM SDK) portlet provides a summary of information about your Oracle CM SDK data. Portlet subscribers can see their quota, initiate a file search, list documents or folders, and upload documents. The portlet also links to the Oracle CM SDK Web interface.

See Also: Chapter 2, "J2EE, Web Services, and Internet Applications"

Partner Portlets

In addition to the list of Oracle Application Server components, a growing community of independent software vendors (ISVs) and Internet content providers are creating standard, supported portlets that access their applications and services. For these partners, customers can access the partner's application or service through one or more pre-integrated portlets.

The current catalog of portlets includes the following services:

- Business intelligence and reporting (including Axis Technology and Quest Software)
- Collaboration (including Cubika Internet Technology and SiteScape)
- Document, content, and knowledge management (including Interwoven and STI AS)
- E-business applications, such as **customer relationship management** and enterprise resource management (including Billboard and Droplets)
- News and information sources (including NT-Exchange.com)
- Portal tools (including Curl Corporation and Oracle Application Server Portal Developer Kit)
- Internet searches (including Business Objects and Quest Software)

See Also: <http://portalstudio.oracle.com> for a complete list of Partner portlets

Custom Portlets

The Oracle Application Server Portal Developer Kit (PDK) allows developers to either reuse existing applications as portlets or create new portlets. Developers can write portlets using familiar languages and technologies. These technologies can be as follows:

- **Java** applications
- Oracle **PL/SQL packages**
- Web pages built with any standard technology like Active Server Pages (ASPs) or Perl
- Web services

See Also: <http://portalstudio.oracle.com> for more information on the Oracle Application Server Portal Developer Kit

Additionally, PDK developers can take advantage of the Knowledge Exchange, available through Portal Studio, to leverage portlets created by other Oracle Application Server Portal users.

Oracle Application Server Portal Integration with Oracle Application Server Web Cache

Oracle Application Server Portal is closely integrated with Oracle Application Server Web Cache to improve the overall availability, scalability, and performance of OracleAS Portal. OracleAS Web Cache combines caching, compression, and assembly technologies to accelerate the delivery of both static and dynamically generated Portal content.

OracleAS Portal functions as a Web Cache origin server to take advantage of OracleAS Web Cache features such as fine-grained cache control and load balancing.

See Also: Chapter 9, "Performance and Caching" for more information about Oracle Application Server Web Cache

Oracle Application Server Web Cache Deployment with Oracle Application Server Portal

When you install Oracle Application Server Portal, an OracleAS Web Cache instance is automatically created with pre-defined cache configuration settings. Portal sites can choose from the following deployment options:

- **Co-located:** OracleAS Web Cache runs on the same physical server as the Portal middle tier. This configuration is appropriate for smaller, low-volume sites where the scalability of the middle tier is not a concern.
- **Dedicated:** OracleAS Web Cache is deployed on a dedicated server that sits in front of one or more Portal middle-tier servers. Dedicated deployments are usually preferable to co-located deployments, as there is no risk of resource contention with other server processes. OracleAS Web Cache performs well on commodity hardware, so a dedicated deployment does not have to be costly in terms of hardware expenditure.

For very high-volume sites and to avoid a single point of failure, two or more nodes running OracleAS Web Cache may be deployed behind a third-party network load

balancer. If you have multiple deployments of OracleAS Portal, each Portal site can have its own OracleAS Web Cache server, or one or more sites can share a single Web Cache. Similarly, a Web Provider can share a Web Cache with a Portal site, or a dedicated Web Cache can be deployed in front of the Web server that hosts the Web Provider.

A browser-based console, Oracle Application Server Web Cache Manager, is used to administer all aspects of OracleAS Web Cache, including configuration of caching and load balancing rules, security, manual and automated invalidation, monitoring, and logging.

Oracle Application Server Portal Architecture

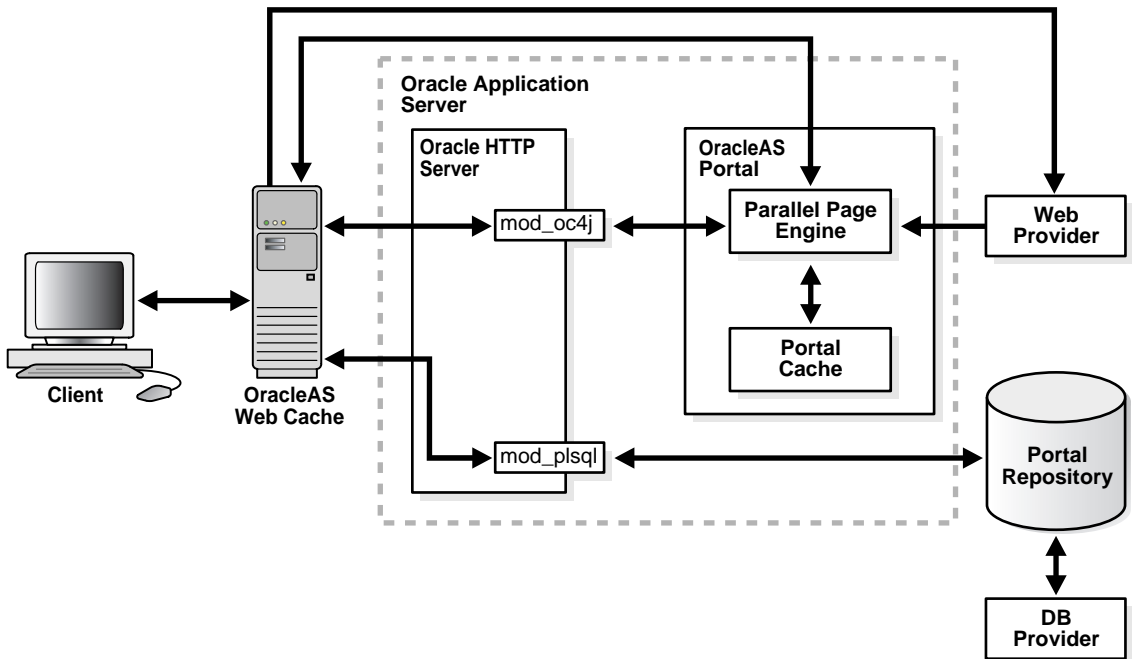
When a **client** requests an **Oracle Application Server Portal** page, many Oracle Application Server components service parts of the request. Requests have the following flow:

1. The client browser requests a **portal** page. **Oracle Application Server Web Cache** receives this request.
2. OracleAS Web Cache forwards the request to the OracleAS Portal Parallel Page Engine (PPE) through Oracle HTTP Server and mod_oc4j.
3. The PPE retrieves the portal page definition. The page definition contains information about the **portlets** on a page and their layout.
 - a. First, it checks if OracleAS Web Cache has a valid, cached copy of the definition.
 - b. Next, it checks if the portal cache has a valid, cached copy.
 - c. Finally, if no cached copy of the definition exists, then the PPE generates a page definition from data in the portal repository. The portal repository is either in the **Oracle Application Server Metadata Repository** or in your **customer database**.
4. The PPE parses the page definition. If a fully cached copy of the page exists, then the page is returned to the client browser through OracleAS Web Cache. If a fully cached copy of the page does not exist, the PPE builds the page from cached and non-cached data with the remaining steps.
5. For each portlet on the page, the PPE checks if a cached copy of the portlet content exists in the portal cache. Then the PPE forwards a request to the appropriate provider through OracleAS Web Cache.

6. Each provider either validates the cached portlet or generates content for the portlet. Web providers return this directly to the PPE using HTTP/S. Database providers return the results to the PPE through Oracle HTTP Server, `mod_plsql`, and OracleAS Web Cache, using HTTP/S or SOAP.
7. The PPE aggregates the content into a single page. This page is sent to OracleAS Web Cache.
8. OracleAS Web Cache returns the final page to the client browser.

Figure 3-2 illustrates this flow in terms of the OracleAS Portal architecture.

Figure 3-2 OracleAS Portal Request Flow



Wireless Applications

This chapter provides an overview of **Oracle Application Server Wireless** features and benefits. The topics include:

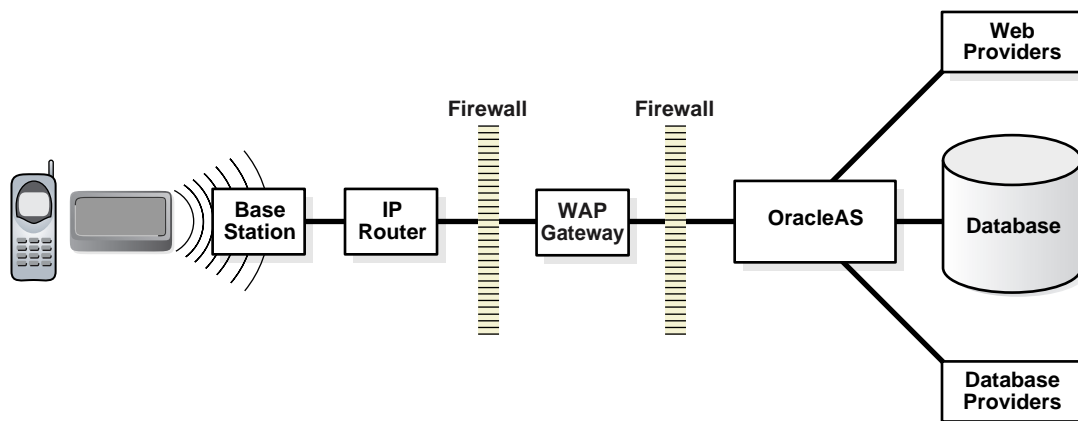
- Introduction to Oracle Application Server Wireless
- Oracle Application Server Wireless Features
- Oracle Application Server Wireless Architecture

Introduction to Oracle Application Server Wireless

Up until now, traditional software applications have not accounted for the fact that employees are mobile. The time spent walking to and from meetings, warehouses, or customers is wasted time without access to key information. Oracle aims to solve this problem with **Oracle Application Server Wireless** by delivering the information that users want, on the mobile device they need it, thereby making them more productive and saving your enterprise money.

Oracle Application Server Wireless Overview

Oracle Application Server Wireless is a component of the Oracle Application Server that helps enterprises and service providers efficiently build, manage, and maintain wireless and voice applications. OracleAS Wireless makes Web and database applications, such as e-mail, news, and directory services accessible to mobile device users without having to rewrite content for every target platform. It transforms content and applications to any markup language supported by any device, such as HTML, WML, HDML, VoiceXML, VoxML, and SMS. Figure 4-1 shows the flow of information between a wireless client and a database application.

Figure 4–1 Oracle Application Server Wireless

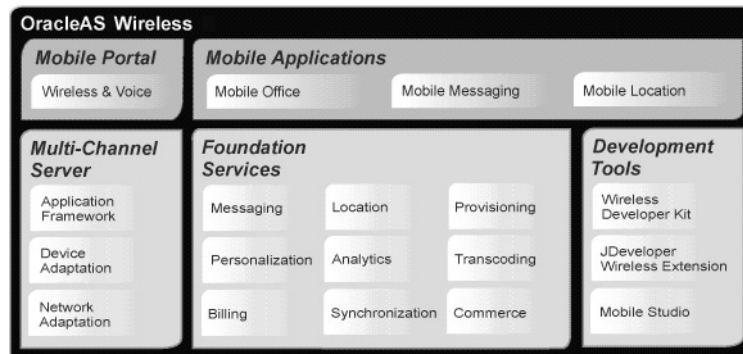
Oracle Application Server Wireless Features

Oracle Application Server Wireless 10g (9.0.4) includes many new features and enhancements, improving the way enterprises and service providers conduct business.

Oracle Application Server Wireless can be split into five component groups:

- **Multi-Channel Server:** The Multi-channel Server detects devices and transforms content and applications to the device.
- **Foundation Services:** The Foundation Services are services to enhance applications and speed development. These are in the form of Java APIs or Web services.
- **Development Tools:** The Development Tools help developers code, test, and debug wireless and voice applications.
- **Mobile Portal:** The Mobile Portal is the end-user wireless portal to access the developed applications and content.
- **Mobile Applications:** The Mobile Applications are out-of-the-box applications to get an enterprise mobile-enabled quickly.

Figure 4–2 shows these five groups with their subcomponents.

Figure 4–2 Oracle Application Server Wireless Overview

Multi-Channel Server

At the core of Oracle Application Server Wireless is the Multi-Channel Server, which enables application access through multiple delivery methods, such as SMS, voice access, WAP, and Pocket PCs. The Multi-Channel Server greatly simplifies and reduces the cost of development by acting as an intelligent wireless proxy for mobile applications. The magnitude of mobile devices and networks are relieved from developers' concerns. Developers can now focus on creating mobile applications for any channel in one, future-proof open standards language. The new Multi-Channel Server extends the existing multi-channel capabilities of previous Oracle Application Server Wireless releases.

Applications written in XHTML are passed through the Multi-Channel Server and translated for any device and network. For example, an XHTML application passed through the Multi-Channel Server is translated to VoiceXML if a phone is accessing the application, and is translated to WML if a WAP phone is accessing the application. The stylesheets used to transcode are maintained and regularly updated by Oracle.

Also new to the Multi-Channel server are Multimedia Adaptation Services. Oracle Application Server Wireless Multimedia Adaptation Services provide device-specific adaptation of images, ringtones, voice grammars, and audio/video streams. Devices support different image formats and have different screen sizes and color depths. As part of the content adaptation performed by OracleAS Wireless in responding to a request, images are dynamically adapted to suit the device. Ringtone adaptation allows for conversion of ringtone data to formats supported by the most popular phones, such as RTTTL, iMelody, and MIDI. The

flexible framework for ringtone adaptation allows developers to easily add support for new ringtone formats.

J2ME Support

Java 2 Micro Edition (J2ME) provides a lightweight operating system for mobile devices, enabling client-side development using open standards. With the large number of J2ME-enabled phones on the market, vendors need a method for efficiently building, managing, and delivering J2ME applications to the right mobile users. Oracle Application Server Wireless 10g (9.0.4) includes complete, end-to-end support for building J2ME applications and delivering them to mobile devices. OracleAS Wireless J2ME support includes the J2ME Developer's Kit and the J2ME Provisioning System.

There is a restriction on the complexity of J2ME applications because of the limited computing power of mobile devices. The more complicated the J2ME application is, the less usable the application will be on a mobile device. One way to create compelling J2ME applications is to use Web services. Applications are able to push some of the CPU-intensive logic to the Web services residing on the server side. However, even the call to Web services from J2ME devices is too CPU intensive. Oracle J2ME Developer's Kit offers the ability to extend Web services to J2ME devices in an optimized manner for mobile devices. Using the J2ME Developer's Kit, J2ME application (MIDlet) developers can make Web services calls through the Oracle Application Server J2ME proxy server using a client stub. Additionally, MIDlet developers can utilize built-in features optimizing communication, such as request and response caching, if the network is unavailable. The calls can automatically resume when the network connectivity resumes.

OracleAS Wireless streamlines the deployment, management, and delivery of J2ME applications with a provisioning system. The application management Web-based tool allows users to upload J2ME applications for management and secure storage. A byte-code inspector verifies the application for any malicious content. The OracleAS Wireless support for over-the-air (OTA) efficiently delivers applications to target users or devices. Digital rights management adds a digital layer around J2ME applications to support business logic that provides full control over the application. The digital wrapper supports billing strategies and application lifespan control.

Notifications and Multimedia Messaging

Oracle Application Server Wireless 10g (9.0.4) further enhances intelligent messaging with functionality for actionable alerts, message adaptation, and failover

delivery control. Also new are multi-media messaging (MMS) features that allow for richer messaging experiences. Existing messaging capabilities have been improved to include more flexible message templates, security to prevent message spoofing, support for message prioritization, and more flexibility in handling volume alerts.

OracleAS Wireless supports multi-media messaging (MMS) for rich mobile messages, including graphics, videos, and audio. MMS messages can be authored natively in SMIL or in open standards XHTML. Messages that are authored in XHTML are automatically adapted for wireless devices by OracleAS Wireless. The power of adaptation allows a message to be written once and automatically optimized for any target device.

Notifications are improved by allowing messages to be sent, and responded to, using the new actionable alerting capabilities, enabling further action from a sent alert. For example, a stock alert can prompt a user to take an action and sell when a target price is hit. Location can now also play a role in alerting. Location-based alerts generate and deliver alert messages based on a mobile user's current location. For example, a field service coordinator receives an alert when a service engineer is within two miles of a customer with an urgent service request.

Asynchronous applications enable messaging devices, such as SMS and email devices, to access applications. A mobile user can maintain a session with an application via SMS or email. To invoke an application, the user sends a message with the message body containing the name of the application and any inputs. A separate message is sent back, by the application, with the results.

Wireless Development Kit

The Oracle Application Server Wireless Development Kit is a small footprint Oracle Application Server Wireless development environment for developing wireless and voice applications. This speeds the development process by giving extra flexibility to fit any development process using any IDE, development tool, Web service, and device simulator. The Wireless Development Kit can be used on any PC or laptop, connected or disconnected, to build and test wireless and voice applications. It is no longer necessary to have a full installation of Oracle Application Server to build and test wireless applications. The Wireless Development Kit supports development for voice, mobile browser, J2ME, and messaging applications.

Oracle offers a version of the Wireless Development Kit specifically for JDeveloper called the JDeveloper Wireless Extension. JDeveloper users can utilize the JDeveloper Wireless Extension for complete wireless development with code

templates, wizards, code insight, and automatic deployment to Oracle Application Server.

Web Clipping

The Wireless Web Clipping Server allows clipping and scraping of existing Web content to create wireless applications that reuse your existing PC browser-based applications. The Wireless Web Clipping Server is used to create many applications, each of which represents Web content that has been clipped and scraped from one or more Web sites scattered throughout a large organization.

To create a Wireless Web Clipping application, the user simply uses a Web browser to navigate to the Web page containing the desired content, then selects the portion of the page to clip and scrape. The user then sets some attributes, exposes input parameters if the Web clipping uses form-based submission, saves the application, and tests the application. The following are some of the features that the Wireless Web Clipping Server supports:

- Navigation through various styles of login mechanisms, including form-based and JavaScript-based submission and HTTP Basic and Digest Authentication with cookie-based session management.
- Fuzzy matching of clippings. If a Web clipping gets recorded within the source page or if its character font, size, or style changes, it will still be identified correctly by the Wireless Web Clipping Server and be delivered as the Wireless Web Clipping application content.
- Reuse of a wide range of Web content, including basic support of pages written in HTML 4.0.1, JavaScript, applets, and plug-in enabled content, retrieved through HTTP `GET` and `POST` functions (form submission).

All Wireless Web Clipping application definitions are stored persistently in the Oracle Application Server infrastructure database. Any secure information, such as passwords, is stored in encrypted form, according to the Data Encryption Standard (DES), using Oracle encryption technology.

Location Services

Oracle Application Server Wireless Location Services give access to the full Location Based Service (LBS) functionality, such as positioning, geocoding, mapping, driving directions, and business directory lookup in an open standards manner. Any application or generic client can use the included WSDLs to invoke the LBS Web services. In addition, OracleAS Wireless instances can use LBS features more conveniently by using the "Web service" provider proxy. This allows you to switch

LBS providers without having to make modifications to the applications using LBS features.

For 10g (9.0.4), the LBS features have been made available through the Web-based tools in addition to being available through APIs. The LBS features allow mobile positioning, to provide the user's current location, and privacy management, to control when and to whom a mobile user's location is available. Both mobile positioning and the caching of the location information can be enabled or disabled by the system or by individual users. Users can grant mobile positioning access to other users or groups of users (communities) for a certain date range and for specified time windows.

This release also allows a mobile user/device to send the current location, which is usually provided by a GPS receiver, to OracleAS Wireless. The current location can be subsequently queried through the existing mobile positioning and privacy management framework. Users can also choose to position themselves manually using the location mark feature. A location mark can be either a point location specified by an address or a region specified by a city, state, or even a country.

In the previous release, users could configure multiple content providers for geocoding, mapping, driving directions, and business directory services. A provider was selected based on static ordering or its availability region. This release adds the ability to monitor the performance and reliability of providers and dynamically adjust the selection criteria. It also logs performance statistics that help administrators in managing their systems.

Mobile Office Applications

Enterprises can get up and running on wireless or voice applications quickly by deploying the mobile applications that are shipped with Oracle Application Server Wireless. The Mobile Office suite of applications provides wireless browser and voice access to email, calendars, address books, tasks, directories, and files. These applications are fully integrated, enabling the most convenient user experience through such features as directory-based or address book-based recipient selection while composing email messages.

All Mobile Office applications are based on standard protocols, allowing a simple integration into existing environments. Mobile Email gives access from any mobile device to any IMAP or POP3 server, including Microsoft Exchange and Lotus Domino. Mobile Directory connects to any LDAP directory server. And finally, Mobile Calendar integrates with Microsoft Exchange and Lotus Domino servers, and through published interfaces Oracle enables easy customization to support any calendar server.

Oracle Application Server Wireless Architecture

In order to set up an Oracle Application Server Wireless environment, you must have the following:

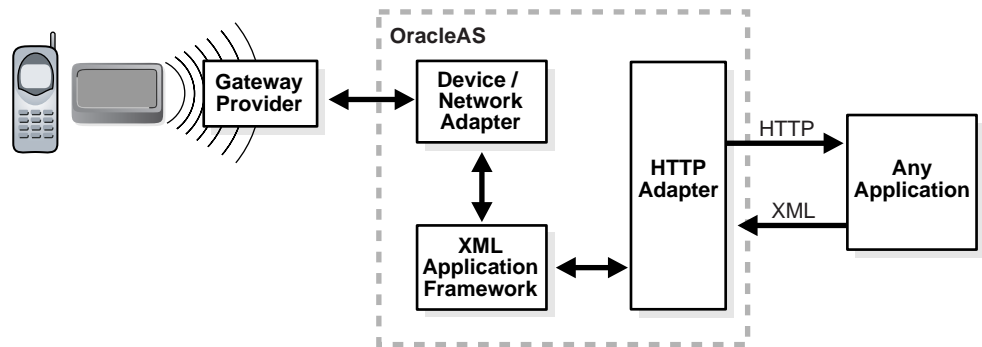
- An application to provide content
- An application server
- A Web site (URL)
- A network provider (ISP or ASP)
- Provisioning and security
- A wireless server
- A gateway (WAP, SMS)
- A wireless network provider
- Wireless devices
- Wireless subscribers

When users request wireless service, the following steps occur:

1. The wireless device connects to the Gateway Provider.
2. The provider provides subscriber, device identification, and user preferences to the Device/Network Adapter.
3. The Device/Network Adapter invokes the XML Application Framework and HTTP Adapter.
4. The HTTP Adapter forwards the request to the application.
5. The HTTP Adapter retrieves the XML result from the application.
6. The result is forwarded to the **client**.

Figure 4–3 illustrates this flow in terms of the **Oracle Application Server Wireless** architecture.

Figure 4–3 Oracle Application Server Wireless Architecture



Oracle Application Server Wireless Adapters

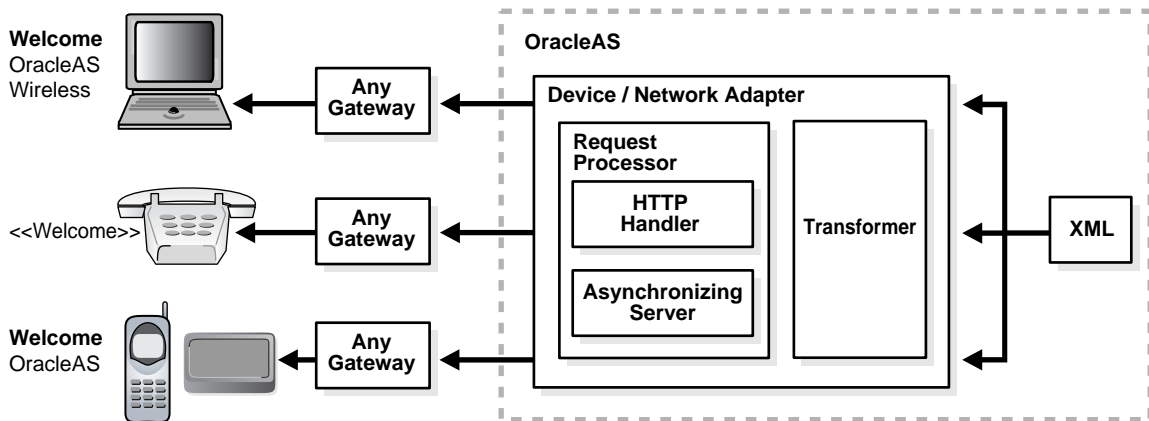
Oracle Application Server Wireless adapters pass application content to the core for processing. The adapter retrieves content from a wide variety of sources, including existing Web sites, databases, existing applications, and legacy systems, and converts the content into Mobile XML.

Each adapter can securely access content from any source, depending on the output of the source. For example, the HTTP Adapter can access content from any source that outputs XML. To render this content to a device, a developer points the HTTP Adapter to the application with a URL. The adapter converts the content to Mobile XML, the passes it to OracleAS Wireless transformers that processes the Mobile XML and deliver the result of the query to the wireless device.

Oracle Application Server Wireless Transformers

Oracle Application Server Wireless transformers complete the conversion of application content from the original format to the target format. Result transformers convert the Mobile XML documents created by the adapters into Simple Result format. Device transformers then convert the Simple Result documents to the markup language appropriate for the requesting wireless device. Figure 4–4 illustrates this conversion process.

Figure 4-4 Oracle Application Server Wireless - Adapters and Transformers



The Device/Network Adapter automatically transforms and optimizes the application content for any wireless device and network. It supports the following mobile technologies:

- 2-way pagers for asynchronous services (SMTP/SMS)
- WAP devices
- Voice for access through regular phone lines
- PDA devices

OracleAS Wireless provides three types of transformers:

- **Generic service transformers:** These are languages such as WML. Generic service transformers convert the Mobile XML to a generic WML format that works on any WAP-compliant wireless phone.
- **Device-specific transformers:** These are optimized for a specific device. For example, instead of using the generic WML transformer, you can use a device-specific transformer that exploits the device characteristics of a specific phone. The Wireless initial repository includes transformers for several target formats, including CHTML, HDML, MML, VoiceXML, and VoxML.
- **Custom transformers:** You can create custom transformers to target new device platforms and optimize content presentation for specific devices. Wireless publishes device transformation rule files so that anyone can create support for any type of device and markup language.

Business Intelligence

This chapter provides an overview of Oracle Application Server Business Intelligence features and benefits. The topics include:

- Introduction to Oracle Application Server Business Intelligence
- Oracle Application Server Reports Services
- Oracle Application Server Discoverer
- Oracle Application Server Personalization
- Integrating Oracle Application Server Business Intelligence

Introduction to Oracle Application Server Business Intelligence

In an increasingly competitive business world, **business intelligence** empowers decision makers to act on information to keep their businesses and services competitive. Complete business intelligence about the operation of an enterprise and its customers can be used to assess opportunities as well as potential risks.

Automated business processes and electronic interactions with customers and suppliers have yielded an abundance of valuable business intelligence data. This data tracks the fundamentals and operations of the business, and presents an opportunity to dramatically improve operational efficiency, reduce costs, and increase overall productivity.

With an **e-business**, traditional information sources such as enterprise resource planning (**ERP**) systems and customer relationship management (**CRM**) systems can be correlated with information that comes directly from the usage patterns of companies' Web sites. Once this information is gathered, it can be analyzed and in turn used to provide a more personalized service to users. This enables companies to more closely analyze consolidated business data, secure existing business relationships, forge new revenue generating opportunities, and assist with the formulation of strategies to improve operating efficiency.

In order for an e-business to successfully gather business intelligence information about Web site usage and provide meaningful data analysis, it must be able to do the following:

- Collect data generated from Web site activities and transform the data into business intelligence information
- Integrate Web traffic and business **transaction** data, then condense this into a meaningful report
- Enable business users at all levels of the organization to perform ad-hoc queries of stored business intelligence data
- Provide a deployment platform for creating multi-source, end-user reports that are available online or on paper
- Provide real-time analysis and recommendations for users of the Web site

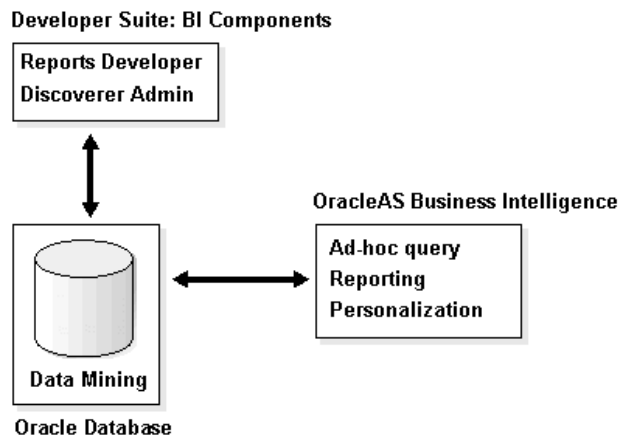
Oracle Application Server Business Intelligence provides a complete, integrated solution that addresses all of these business intelligence requirements. Oracle Application Server Business Intelligence consists of the following components:

- **Oracle Application Server Reports Services** provides a powerful deployment platform for publishing high-quality, dynamically generated end-user reports in a scalable, secure environment. You can build and publish sophisticated reports from any source, with unlimited data formatting, and deploy them seamlessly on Oracle Application Server.
- **Oracle Application Server Discoverer** enables business users at all levels of the organization to gain immediate access to information from data warehouses, and online transaction processing (OLTP) systems. Users create ad hoc queries, reports and graphs, drill up and down on their result set, pivot, and sort data. Users also create their own calculations, conditions, and totals to generate custom reports.

- **Oracle Application Server Personalization** provides real-time analysis and recommendations for e-business sales channels, such as Web stores, application hosting environments, and business call centers. Oracle Application Server Personalization provides an integrated real-time recommendation engine built on predictive models that are automatically created within the Oracle Database Server using embedded data mining technology. Oracle Application Server Personalization delivers powerful, scalable real-time personalization for customer areas of interest. This enables businesses to deliver tailored, one-to-one customer experiences for Web site users or any touch-point that can call a Java API for real-time recommendations.

Figure 5-1 shows an overview of Oracle Application Server Business Intelligence components.

Figure 5-1 Oracle Application Server Business Intelligence Components



Oracle Application Server Reports Services

In a business intelligence environment, data is generated by multiple sources in various formats. Publishing this business intelligence data should be fast and easy, without requiring multiple, convoluted steps to manipulate and consolidate the data into a single format.

Introduction to Oracle Application Server Reports Services

Oracle Application Server Reports Services provides a robust deployment platform for creating high-quality, dynamically generated end-user reports in a scalable, secure environment. You can build and publish sophisticated reports from any data source, in any format, and deploy them anywhere with Oracle Application Server.

OracleAS Reports Services is part of Oracle Reports. Oracle Reports also includes Oracle Reports Developer, a component of the Oracle Developer Suite. Using Oracle Reports Developer you can build a complex data model and share it between an existing high quality paper layout and an improved high quality Web layout using **servlets** and **JSP** technology.

Developers can publish sophisticated, high quality reports from any data source, in any data format, and deploy them anywhere on Oracle Application Server. OracleAS Reports Services can combine data from multiple data sources into a single report, including the Oracle database, **XML** feeds and **JDBC**-enabled data sources. Developers can even add custom data sources using the open **API** for **Pluggable Data Sources**.

Reports can be generated on demand or scheduled for a specific time or interval. Once formatted, a report can be distributed to a standard Web browser, email, printer, or **Oracle Application Server Portal**, be stored on the middle tier for rapid retrieval, or be stored in a custom destination via the **Pluggable Destinations API**.

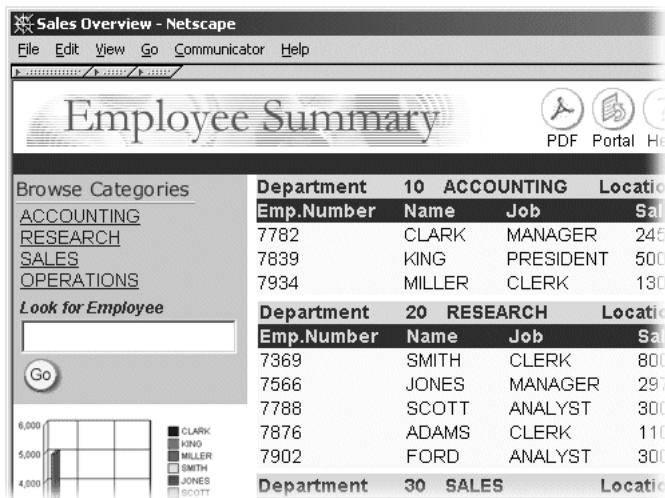
OracleAS Reports Services enables you to do the following:

- Access reports dynamically from any browser on demand
- Leverage middle-tier load balancing to provide high volumes of reports, without excessive demands on limited resources
- Generate reports in **HTML** for browser viewing, **PDF** for quality viewing, **XML** to communicate data to Web sites, **RTF** and delimited output for import into Microsoft Office, and **PCL** for printing
- Optionally secure distribution of reports by allowing only specific users to access the report and grant secure access
- Link to published reports and provide **portlet** support using OracleAS Portal

See Also: Chapter 3, "Portal Applications"

Figure 5–2 shows an example of an Oracle report.

Figure 5–2 Oracle Reports Example



Note: Oracle Reports Developer is available with Oracle Developer Suite.

Oracle Application Server Reports Services Features

Oracle Application Server Reports Services provides the following features:

- Event-Based Reporting
- Extensible Architecture
- Enhanced Report Bursting and Distribution

Event-Based Reporting

OracleAS Reports Services enables you to run a report that is triggered by database events. You can execute a report using a set of **PL/SQL stored procedures** that can be called from a database trigger. The trigger invokes OracleAS Reports Services and the event data is submitted to the **server**. For example, when an employee submits an expense report, new data is being inserted into the database. The insert event causes a database trigger to fire, which sends a report to the employee manager on a **portal** page or in an e-mail message asking for expense report approval.

In addition to database events, you can trigger reports from within your Java application using the Reports Web Service or from within your business process using Oracle Workflow and the Reports Workflow activity.

Extensible Architecture

OracleAS Reports Services enables you to customize your access to data sources, runtime engines, security, notification, distribution, and cache, as follows:

- **Write your own data access methods for data that is specific to your business:** A Pluggable Data Source (PDS) is a set of **Java APIs** that provides openness to OracleAS Reports Services by enabling data input from any source. PDSs allow OracleAS Reports Services to combine data from various data sources including multiple Oracle database connections – each potentially having a different connection. When written to the PDS API, you can specify the icons to be displayed in the data model and Report Wizard of Oracle Reports Developer.
- **Use OracleAS Reports Services functionality with your own runtime engine or job type:** The OracleAS Reports Services engine is the default implementation provided by Oracle. However, you can take advantage of the security, notification, distribution, and **caching** features of OracleAS Reports Services using your own engine code. An example of runtime engine code is a Java class that uses an operating system call to package a series of reports into a zip file and then distributes that file using e-mail to a defined set of users.

- **Create your own security mechanisms for user-authentication against the OracleAS Reports Server:** The OracleAS Reports Services architecture provides a complex security mechanism. The OracleAS Reports Server authenticates users and performs the necessary security checks to ensure the user is able to process the selected report. The out-of-box interface allows the administrator to easily enter the OracleAS Reports access control information from within a Web browser. All of the access control data is then stored within the Oracle Application Server Portal repository, which is checked each time a user runs a report request.
- **Configure OracleAS Reports Services to submit notifications upon success or failure of a job:** Besides using the default implementation of sending an e-mail message, you can now implement your own notification mechanisms.
- **Distribute to various destinations such as the Oracle AS Reports Services cache, e-mail messaging, and printers:** You can add your own destinations such as fax machine and FTP servers using the pluggable destination API.
- **Replace the default caching with your own caching mechanism for the output of the OracleAS Reports Services using the pluggable caching API:** The pluggable caching APIs are custom engines that use Java APIs to pass jobs to the OracleAS Reports Server.

Enhanced Report Bursting and Distribution

Oracle Application Server Reports Services enhanced bursting and distribution enables you to distribute a report to different media, such as paper, e-mail messages, Web pages, or even delimited output like a spreadsheet.

You can burst on repeating data within a report. Bursting enables you to create multiple personalized reports out of a single report model. For example, department-based reports can be distributed to employees within that department. This ensures that each user only receives information that is relevant and appropriate for them. For example, with a salary report, the manager of a sales department will get an e-mail with the salary information for employees in that department only; the manager of the purchasing department will only be e-mailed the salary information for employees in that department; the CEO of the company will receive a printout containing a summary of each department's information.

The report runs only once, and the output is sent to different sets of users according to requirements. You only have a single set of data to fetch and format in order to distribute the report.

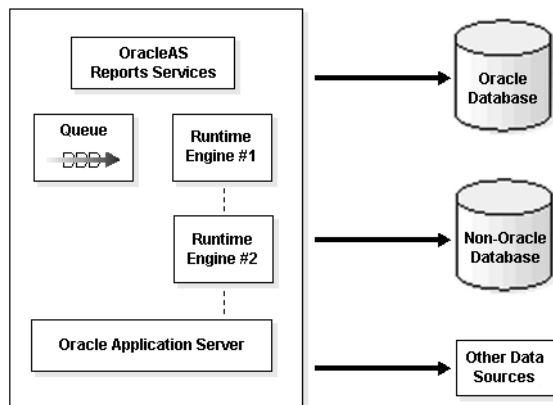
Oracle Application Server Reports Services Architecture

Figure 5–3 shows how Oracle Application Server Reports Services handles **client** requests. OracleAS Reports Services runs reports by entering all requests into a job queue. When one of the server's runtime engines becomes available, the next job in the queue is dispatched to run.

As the number of jobs in the queue increases, the server can start more runtime engines until it reaches the maximum limit specified in your server configuration. OracleAS Reports Services runtime engines are shut down automatically after having been idle for a period of time that you specify.

OracleAS Reports Services keeps track of all jobs submitted in the server, including jobs that are running, scheduled to run, or finished. The Reports Queue Manager (Windows), the Reports Queue Viewer (UNIX), or the `show jobs` command (Web) enable you to view information on when jobs are scheduled, queued, started, and finished, as well as the job output and the final status of the report.

Figure 5–3 Oracle Application Server Reports Services Architecture



OracleAS Reports Services job **objects** are persistent. This means that if the server is shut down and then restarted, all jobs are recovered, not just scheduled jobs.

Oracle Reports Developer enables you to embed a report within a larger existing Web page. This technology enables you to open and save HTML, JSP, and XML files that contain report definitions. When a report is saved as a JSP file, the data model is embedded using XML tags. The entire report can also be defined using XML tags and saved as an XML file.

You can also use Oracle Reports Developer to take retrieved data, using the data model, and embed it into an existing Web page. This provides tremendous flexibility in creating reports that meet business demands by completely integrating multiple sources of information within a single Web page. If you choose to create your own JSP, Oracle Reports Developer supplies templates that can be used to build your report.

Developers can easily create a JSP layout using the Reports Block Wizard to generate the necessary **JSP tags** in Oracle Reports Developer. Alternatively, they can add the tags themselves manually for more precise control.

See Also: *Oracle Application Server Reports Services Publishing Reports to the Web*

Note: Oracle Reports Developer is available with Oracle Developer Suite.

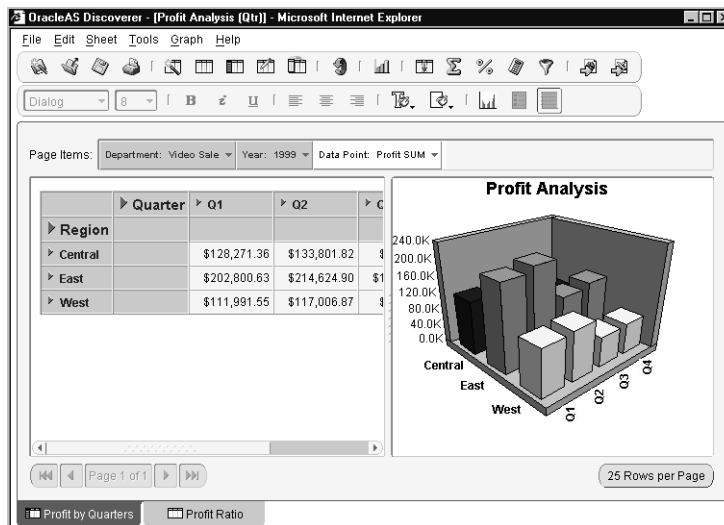
Oracle Application Server Discoverer

Oracle Application Server Discoverer is an intuitive ad-hoc query, reporting, analysis, and Web publishing tool. Using OracleAS Discoverer, business users at all levels of the organization gain immediate access to information from data warehouses and OLTP systems.

Introduction to Oracle Application Server Discoverer

Using OracleAS Discoverer you can easily access information, drill down, pivot, and sort data. You can perform simple calculations, such as sub-totals and percentages, and also make use of advanced analytic functions that are available in the Oracle database.

Figure 5-4 shows an example of an OracleAS Discoverer worksheet.

Figure 5–4 Oracle Application Server Discoverer Worksheet Example

OracleAS Discoverer enables you to present the information you want to see in a familiar spreadsheet format. Using OracleAS Discoverer, you can do the following:

- Obtain information from a database without knowing how the database is structured, or how to use a database programming language
- Obtain the latest information without waiting for reports to run overnight, or for programmers to write reports for you
- Analyze data interactively using the powerful analysis tools of OracleAS Discoverer to drill, pivot, and graph information
- Create worksheets in tabular and cross-tab formats using OracleAS Discoverer wizards
- Create totals, calculate percentages, and create custom calculations on your data
- Perform advanced SQL analytic calculations using the OracleAS Discoverer easy-to-use analytic function templates

For example, a manager might view a summary report of sales that shows the top performing products for the quarter, and drill down to view results by region or by sales person. The manager could also use Oracle Application Server Discoverer to perform a quick comparison with previous quarters, or to calculate the three-month moving average of the organization's sales.

The end result is a collection of worksheets and graphs that persuasively support your business decisions and help you identify business opportunities.

Oracle Discoverer Tools

Oracle Application Server Discoverer provides the following tools:

- Oracle Application Server Discoverer Viewer
- Oracle Application Server Discoverer Plus
- Oracle Application Server Discoverer Portlet Provider
- Oracle Discoverer Desktop
- Oracle Discoverer Administrator

Oracle Application Server Discoverer Viewer

Oracle Application Server Discoverer Viewer enables business users to run, view, and graph reports created by users of Oracle Application Server Discoverer Plus. OracleAS Discoverer Viewer is a true lightweight business intelligence tool, requiring no other software than a simple Web browser. Its Web interface is familiar to anyone who regularly uses a browser to view pages on the Internet, and it uses the same high performance, scalable architecture as other OracleAS Discoverer solutions. OracleAS Discoverer Viewer reports are live, enabling users to interact with their data by changing layouts and specifying parameters.

OracleAS Discoverer Viewer breaks large worksheets into pages in the background and displays the first page. Additional pages are displayed on demand. This reduces network traffic between the **client** and the OracleAS Discoverer server. The OracleAS Discoverer server monitors client activity and ends inactive sessions after a specified time-out limit. This frees up valuable resources for other users.

Oracle Application Server Discoverer Plus

Oracle Application Server Discoverer Plus enables business users to query, graph, and create reports. OracleAS Discoverer Plus runs as a **Java** client. You can perform drill and pivot analysis, and publish results to the World Wide Web.

Similar to OracleAS Discoverer Viewer, OracleAS Discoverer Plus breaks large worksheets into pages in the background and displays the first page. Additional pages are displayed on demand, reducing network traffic between the client and the OracleAS Discoverer server.

Oracle Application Server Discoverer Portlet Provider

Integration between Oracle Application Server Discoverer and **Oracle Application Server Portal** through the OracleAS Discoverer Portlet Provider enables you to publish reports or a list of reports to OracleAS Portal. This enables other users in the organization to easily access the information by subscribing to these **portal** pages. OracleAS Discoverer is a **portlet** provider to OracleAS Portal. There are two types of OracleAS Discoverer portlets: a list of workbooks portlet, and a worksheet portlet. Including a list of workbooks portlet in an OracleAS Portal page enables users to select a workbook and display the workbook in OracleAS Discoverer Viewer. Including the worksheet portlet in an OracleAS Portal page displays the information from that OracleAS Discoverer worksheet (as data, a graph, or both) along with a link to open the worksheet in OracleAS Discoverer Viewer.

See Also: Chapter 3, "Portal Applications"

Oracle Discoverer Desktop

Oracle Discoverer Desktop provides functionality similar to that of OracleAS Discoverer Plus. However, unlike OracleAS Discoverer Plus, Oracle Discoverer Desktop runs in a **client/server** architecture. Workbooks created with Oracle Discoverer Desktop can be opened in Oracle Application Server Discoverer Plus and Oracle Application Server Discoverer Viewer.

Note: Oracle Discoverer Desktop is available with Oracle Developer Suite.

Oracle Discoverer Administrator

The Oracle Discoverer manager uses Oracle Discoverer Administrator to perform a variety of tasks, including maintenance of business areas, folders, summary table creation, and end user access. Oracle Discoverer Administrator provides extensive defaulting capability to enable managers to rapidly implement their end user environment. The Oracle Discoverer Administrator (a component of Oracle Developer Suite) is required to support Oracle Application Server Discoverer Plus, Oracle Application Server Discoverer Viewer, and Oracle Application Server Discoverer Desktop.

Oracle Discoverer Administrator is used for the initial setup and ongoing maintenance of the Oracle Discoverer End User Layer (EUL). The Oracle Discoverer EUL is a server-based, low maintenance, powerful metadata repository and query management engine for data warehouses and OLTP systems. It automatically performs query redirection to summary tables. The Oracle Discoverer EUL is

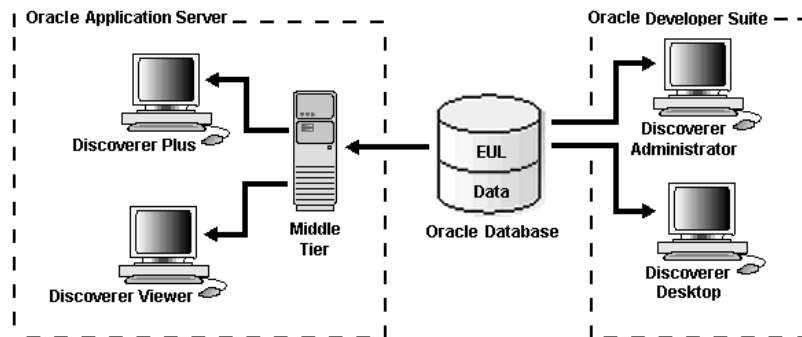
centrally stored in the RDBMS for scalability and ease of maintenance, and is shared by both the client and server.

Note: Oracle Discoverer Administrator is available with Oracle Developer Suite.

Oracle Application Server Discoverer Architecture

Figure 5–5 shows how data flows through and is processed by Oracle Application Server Discoverer.

Figure 5–5 Oracle Application Server Discoverer Architecture



OracleAS Discoverer uses an efficient cache that is completely transparent to both the end user and the administrator. The cache enables rapid analysis without re-querying the database. In a Web environment, this cache is located on a middle tier server and is associated with the user's session. This enables end users to ask subsequent questions about the data without having to re-execute the query on the database server. Pivoting of data and many calculations are handled on the middle tier to provide exceptional performance. As users drill for more detail, OracleAS Discoverer fetches only the additional information requested and incorporates this into the existing cache, reducing both database and network traffic.

The first time that a computer is used to connect to OracleAS Discoverer, the OracleAS Discoverer Plus applet is downloaded from the OracleAS Discoverer Services tier and cached on the client computer. The OracleAS Discoverer Plus applet provides the OracleAS Discoverer Plus user interface and functionality for analyzing business intelligence data. When the user logs on subsequently, the

OracleAS Discoverer Plus applet runs from the local cache and does not need to be downloaded.

The OracleAS Discoverer servlet generates HTML pages from HTTP requests using an XML/XSL processor. The OracleAS Discoverer Plus servlet provides the connection pages for OracleAS Discoverer Plus and OracleAS Discoverer Viewer. The OracleAS Discoverer Plus servlet forwards requests from the OracleAS Discoverer Plus applet to the OracleAS Discoverer servlet. The OracleAS Discoverer Plus servlet adds a URL parameter to the request indicating that the OracleAS Discoverer Plus applet was the originator of the request.

The OracleAS Discoverer Portlet Provider servlet provides a user interface for publishing OracleAS Discoverer worksheets and links to OracleAS Discoverer workbooks on an Oracle Application Server Portal page.

The OracleAS Discoverer database tier consists of:

- OracleAS Discoverer workbooks used to store reports and charts
- The End User Layer (EUL) that provides an easy-to-understand view of the data
- The business intelligence data that users want to analyze

The OracleAS Discoverer EUL enables you to access the same metadata and share reports regardless of client type.

See Also: *Oracle Application Server Discoverer Plus User's Guide* and *Oracle Application Server Discoverer Configuration Guide*

Oracle Application Server Personalization

Today's e-businesses must compete by maintaining personal, one-to-one relationships with their Web customers, including both registered customers and anonymous Web visitors. Companies must provide e-customers with customized product recommendations, ratings of the likelihood that the customer will "like" the recommendations, and improved site navigation based on their interests and profiles.

Introduction to Oracle Application Server Personalization

Oracle Application Server Personalization provides real-time analysis and recommendations for Web stores, application hosting environments, and business call centers. Oracle Application Server Personalization provides an integrated real-time recommendation engine that is deployed with Oracle Application Server.

By delivering real-time personalization, Oracle Application Server Personalization delivers powerful, scalable real-time personalization for customer areas of interest. This enables e-businesses to provide e-customers with customized product recommendations, rank the likelihood that a customer will like the recommendations, and improve site navigation based on customer interests and profiles.

OracleAS Personalization is used by business-to-consumer customers, **business-to-business** customers, and Application Service Providers (ASPs) who need to personalize their customer "touch points" to support business-critical operations to achieve a competitive advantage.

Oracle Application Server Personalization provides recommendations and answers to questions such as these:

- Which items is the customer most likely to buy or like?
- Customers that bought or like this item are likely to buy or like which other items?
- How likely is it that the customer will buy or like this item?
- Which items is the customer most likely to buy or like given that the customer likes or is buying another item?

Oracle Application Server Personalization Features

Oracle Application Server Personalization provides the following features:

- Real-Time Recommendation Engine Architecture
- Data Mining Technology
- Single Administrative Interface

Real-Time Recommendation Engine Architecture

OracleAS Personalization dynamically serves personalized recommendations (such as products, content, and navigational links) in real time based on a registered customer's or anonymous visitor's explicit input (**transactions**, purchases, ratings, and demographic data) and implicit information (mouse clicks, pages visited, and banners viewed).

Data Mining Technology

Powerful data mining technology embedded in the Oracle Database Server automatically discovers individualized behavior patterns to generate highly accurate personalized recommendations in real time. OracleAS Personalization uses data mining to build data mining models. The models predict what the Web site visitor will probably like or buy. The predictions are based on the data collected for that Web site visitor in previous sessions, in the current session, and on demographic, purchasing, and ratings information.

OracleAS Personalization works in conjunction with existing Web applications and other applications that can make Java API calls. The applications ask OracleAS Personalization to capture certain visitor or customer activities, such as navigational clicks, adding items to a shopping cart, or providing demographic information. This data is both cached by OracleAS Personalization and saved into an Oracle database **schema**. At user-scheduled times, OracleAS Personalization mines the collected data, builds predictive models with rules for making recommendations, and populates its recommendation engines. The calling application can then ask OracleAS Personalization to generate real-time, individualized recommendations to display to each visitor or customer. OracleAS Personalization uses the rules to retrieve a list of items about which the visitor is most likely to be interested.

Single Administrative Interface

OracleAS Personalization enables you to build, configure, manage, and deploy many recommendation engines throughout your enterprise from a single administrative interface.

Additionally, the administrative interface supports deployment of multiple recommendation strategies for different campaigns or time periods (such as holidays). You can also capture and model behavior for specific events using the events scheduler.

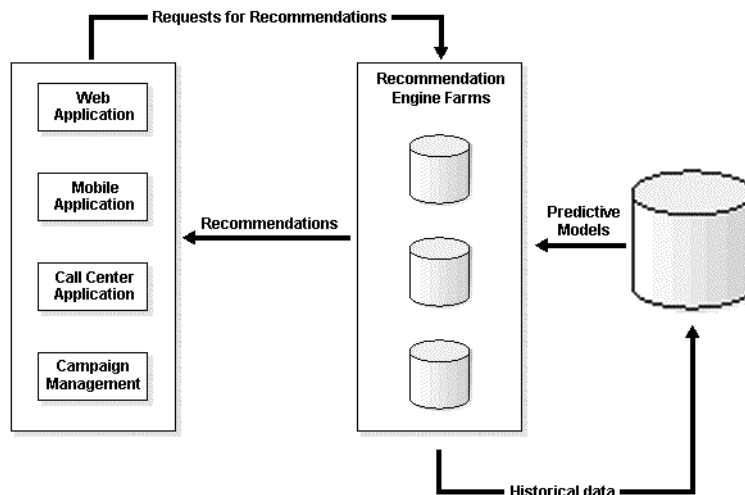
Oracle Application Server Personalization Architecture

Figure 5–6 shows how data flows through and is processed by Oracle Application Server Personalization.

OracleAS Personalization incorporates visitor activity, through Web, mobile, call center, and campaign applications, into recommendations. This activity data is saved for that visitor in an Oracle database, where OracleAS Personalization uses the data to build predictive models. Future visits to the Web site add to the set of data for a customer. The new data allows for more accurate predictions.

OracleAS Personalization uses **SQL** queries for obtaining scores, which can be executed in real-time or in batch mode. Recommendation engines serve OracleAS Personalization recommendations to Web sites across the enterprise.

Figure 5–6 Oracle Application Server Personalization Architecture



OracleAS Personalization predictive models may be rebuilt on a periodic basis (e.g., daily, weekly, monthly) and deployed to the recommendation engines when completed. OracleAS Personalization enables users to create "recommendation

engine farms" that are comprised of many recommendation engines serving customized recommendations to the Web site.

The architecture is extremely scalable for high-traffic sites. OracleAS Personalization stores the predictive models in memory to handle the high traffic and speed requirements associated with e-commerce sites. Data mining algorithms find hidden patterns and customer profiles that drive personalized recommendations.

OracleAS Personalization uses the Oracle Application Server Personalization Administrator to define a **package** that contains information needed to build predictive models using data mining technology in the database, as well as information about the database connections. The OracleAS Personalization Administrator creates and manages schedules for building the packages and for deploying the packages to the recommendation engines that will produce the recommendations. Recommendation engines with the same package are grouped together in recommendation engine farms.

See Also: *Oracle Application Server Personalization User's Guide*

Integrating Oracle Application Server Business Intelligence

Oracle Application Server Business Intelligence provides a complete, integrated, and open solution to meet every company's needs.

Tight integration between Oracle Application Server, Oracle Developer Suite, and the Oracle Database delivers performance and scalability. Oracle Application Server Business Intelligence provides the technology that makes data visible across all parts of the business. The company's decision makers are enabled to gather, analyze, and share the information quickly with stakeholders such as internal employees, external customers, suppliers, and partners. Deploying Oracle Application Server Business Intelligence leads to more accurate, significantly faster, and more cost effective decision making.

In turn, businesses benefit from such decisions by developing differentiated product offerings, improving operational efficiency, and building profitable customer relationships.

Oracle Application Server Business Intelligence provides the capability to:

- Consolidate disparate sources of data
- Turn the data into information via ad hoc query, analysis, and reporting
- Share the information with internal and external stakeholders

- Create a custom business intelligence application or extend an existing application to meet specific business requirements

Oracle Application Server Business Intelligence provides a fully integrated e-business intelligence environment. The following sections discuss Oracle Application Server Business Intelligence in relation to other Oracle Application Server components.

- Leveraging Single Sign-on Functionality
- Leveraging Oracle Application Server Portal Functionality
- Creating an E-Business Dashboard Using Oracle Application Server Portal
- Using Oracle Enterprise Manager Application Server Control for Management
- Implementing Enterprise Reporting with Oracle Reports Builder
- Improving Performance with Oracle Application Server Web Cache

Leveraging Single Sign-on Functionality

A single sign-on service provides a single authentication mechanism that allows users to identify themselves securely to multiple applications through a single authentication step. Web-based e-businesses can use **single sign-on** functionality for deployment of business intelligence applications to employees, customers, and partners.

Oracle Application Server Single Sign-On is a component of Oracle Application Server that enables users to log in to multiple Web-based applications, such as expense reports, e-mail, and benefits information, using a single user name and password. As such, OracleAS Single Sign-On serves as the security gateway for all Oracle Application Server Business Intelligence features.

With OracleAS Single Sign-On, each user maintains only one identity and password for all applications they access. OracleAS Single Sign-On creates an enterprise-wide user to access multiple accounts and Oracle Application Server applications.

Both the Reports and Discoverer components of Oracle Application Server Business Intelligence leverage OracleAS Single Sign-On functionality to provide a seamless point of access to all of your business intelligence users.

See Also: "Oracle Application Server Single Sign-On" on page 11-4

Leveraging Oracle Application Server Portal Functionality

Oracle Application Server Portal is a browser-based, data publishing and developing solution that offers Web-based tools for publishing information on the Web and building Web-based, data-driven applications.

OracleAS Portal is tightly integrated with Oracle Application Server Reports Services to create a robust and secure data publishing environment. OracleAS Portal provides easy-to-use wizards for setting up Oracle Application Server Reports Services security. These include wizards for defining user access to reports, Oracle Application Server Reports Servers, printers, output formats, and report parameters.

Once you define access control information, it's stored in the OracleAS Portal repository. As an OracleAS Portal user, you can then, optionally, publish registered resource description frameworks (RDFs) and JavaServer Pages (JSPs) to an OracleAS Portal page. As with all OracleAS Portal functionality, using OracleAS Portal to deliver your reports is not required. You can deliver reports through command lines and still benefit from the access control features available to you through OracleAS Portal.

Access to OracleAS Reports Services security features is not dependent on whether you also use OracleAS Portal to publish report links or report content. Even if you don't publish via OracleAS Portal, you can still take advantage of the OracleAS Reports Services security features available in OracleAS Portal to control user access to all of your reports.

See Also:

- *Oracle Application Server 10g Security Guide*
- *Oracle Application Server Reports Services Publishing Reports to the Web*

Creating an E-Business Dashboard Using Oracle Application Server Portal

By integrating **Oracle Application Server Portal** and Oracle Application Server Business Intelligence features, you can create an E-Business Intelligence dashboard.

All Oracle Application Server Business Intelligence components can be integrated into the portal as **portlets**. A portlet is a reusable component and basic building block of a **portal** page. Portlets create part of a Web page in either **HTML** or **XML**. Using Oracle Application Server Business Intelligence components with OracleAS Portal you can do the following:

- Integrate with Oracle Application Server Personalization applications to create portlets
- Create and distribute **Oracle Application Server Reports Services** reports as portlets
- Create **Oracle Application Server Discoverer** List of Workbooks Portlets or Worksheet Portlets

See Also: Chapter 3, "Portal Applications"

Using Oracle Enterprise Manager Application Server Control for Management

Oracle Application Server Business Intelligence integration with Oracle Enterprise Manager Application Server Control provides a centralized configuration management tool, enabling managers to view and configure Oracle Application Server services in the middle tier. Application Server Control provides easy-to-use graphical interfaces for managing Oracle Application Server Business Intelligence services. From Application Server Control you can:

- Administer Oracle Application Server Business Intelligence services
- Manage public connections
- View and manage ports from a single intuitive interface
- View and search Oracle Application Server log files
- Switch user defined connections on or off
- Set default locale for connections
- Set connections to Oracle Application Server Single Sign-On
- Examine real-time performance metrics and resource usage

See Also: *Oracle Enterprise Manager Administrator's Guide*

Implementing Enterprise Reporting with Oracle Reports Builder

Oracle Application Server Discoverer provides seamless integration with Oracle Reports Builder, the enterprise reporting tool available from Oracle Developer Suite. OracleAS Discoverer users export workbooks with full fidelity. Information about layout, formatting, exceptions, parameters, and other features is passed via XML, enabling OracleAS Discoverer workbooks to be extended using the full power

of Oracle Reports Builder. This also provides users access to the full suite of publishing capabilities of OracleAS Reports Services, such as batch scheduling and PDF support. Refer to the Oracle Developer Suite Reports Builder online help for more information.

See Also: *Oracle Application Server Reports Services Publishing Reports to the Web*

Note: Oracle Reports Builder is available with Oracle Developer Suite.

Improving Performance with Oracle Application Server Web Cache

To boost performance over the Internet or extranet, Oracle Application Server Business Intelligence leverages Oracle Application Server Web Cache. Oracle Application Server Discoverer uses OracleAS Web Cache to speed up the response time for the most common requests. Additionally, you can use OracleAS Web Cache as a router to balance the load across all available application servers.

See Also: *Oracle Application Server Web Cache Administrator's Guide*

Oracle Application Server Integration

This chapter provides an overview of Oracle Application Server Integration features and benefits.

This chapter contains these topics:

- Introduction to Oracle Application Server Integration
- What is Oracle Application Server ProcessConnect?
- Oracle Application Server ProcessConnect Features
- Oracle Application Server ProcessConnect Architecture
- Oracle Application Server ProcessConnect Summary

See Also: *Oracle Application Server ProcessConnect User's Guide* and *Oracle Application Server InterConnect User's Guide* for detailed information on Oracle Application Server Integration

Introduction to Oracle Application Server Integration

Enterprise Integration solutions are designed to allow customers to address two primary issues: data consistency between systems and automating business processes spanning multiple systems. The need for Enterprise Integration has traditionally been driven by three factors:

- Corporate mergers and acquisitions that lead organizations to have a variety of different information technology (IT) systems that need to be rationalized
- Use of packaged applications in various lines of business that need to be connected to support organization-wide business processes
- Business process re-engineering efforts that need processes to be streamlined and optimized for greater efficiency

The shift to e-business further accelerated these trends and also created new business needs that have driven additional requirements for organization-wide integration, such as the following:

- Virtual supply chains require organizations to connect their supply chains with those of their trading partners to get better visibility and optimizations across the supply chain.
- Customer relationship management requires organizations to knit their customer-facing systems with those of their distribution channel partners in order to better serve customer needs.
- Corporate self-service requires organizations to streamline their own organization-wide business processes for greater efficiency.

Driven by these different factors, organizations want Enterprise Integration solutions that address the two primary integration issues: data consistency and automating business processes. Data consistency solutions essentially allow organizations to synthesize a consistent view of information from the fragmented data they have stored in fragmented data stores. For instance, a financial institution may have customer information in many different databases, but want its customer service representatives to have a single view of the customer across all the different financial products and services the customer purchases. A data consistency solution not only allows an organization to synthesize such information from the different data stores, but also provides the ability to synchronize data across these different systems for consistency.

Business process automation solutions essentially allow organizations to design, monitor, and optimize business processes that span multiple systems and applications within an organization and with trading partners. For instance, a manufacturing company may want to optimize how quickly and efficiently orders taken through its e-commerce storefront are passed into the following:

- Financial systems (to track orders and bill customers)
- Supply chain planning systems (to generate a bill of materials)
- Suppliers (to procure the necessary parts cost effectively and just-in-time)

A business process automation solution not only allows an organization to design a business process and automate it across different systems, it also allows an organization to monitor the events that drive the business process and optimize the business process.

Oracle Application Server offers two integration solutions: Oracle Application Server InterConnect and Oracle Application Server ProcessConnect.

Oracle Application Server InterConnect is a fast message broker that is useful for integrating data between endpoints such as databases, applications, and trading

partners. It provides integration both within an enterprise and between enterprises. However, logging and monitoring capabilities are limited to keep speed maximized.

Oracle Application Server ProcessConnect is a standards-based solution designed to integrate processes within and between enterprises. It offers process monitoring, activity monitoring, and process optimization, and allows you to see a common view of all your events and processes.

This chapter discusses Oracle Application Server ProcessConnect in further detail. For more information about Oracle Application Server InterConnect, see the E-Business Integration section in the *Oracle Application Server 10g Documentation Library*.

What is Oracle Application Server ProcessConnect?

Oracle Application Server ProcessConnect is a single, standards-based, event-driven business process automation solution that is designed to integrate data, applications, and business processes within the enterprise (enterprise application integration or EAI) and between companies (business-to-business or B2B integration). It is built as a fundamental component of the Oracle Application Server and uses a number of standards to integrate systems with each other, including:

- Java 2 Environment, Enterprise Edition (J2EE)
- Java Messaging Services (JMS)
- J2EE Connector Architecture (JCA)
- Web services
- RosettaNet
- EDIX12
- EDIFACT

A fundamental element of its design is the ability to integrate systems with each other using a common data format (or syntax) and common semantics associated with the business process that spans the systems being integrated. Further, this common syntax and semantics is defined to be independent of the data and process semantics of any system (or end point) being integrated. By isolating the design of the business process from the syntax and semantics of any system being integrated, organizations have the ability to integrate existing applications and trading partners together into corporate business processes while retaining the flexibility to easily extend these processes as more applications, trading partners, and B2B protocols become available for integration.

A second fundamental element of the Oracle Application Server ProcessConnect design is that it uses a model-based approach to integration as compared to a programmatic approach. Analysts use a set of integration concepts to model the integration. This enables analysts to focus on what needs to be done without needing to specify how to do it. Specifically, analysts use three design concepts to model the different elements associated with an integration:

- Events
- Roles and processes
- Agreements

Analysts use a single visual tool to do the following tasks:

- Model events, roles and processes, and agreements
- Add the necessary data translations, transformations, and correlations to complete the definition of the business process
- Save the design in a design-time repository
- Validate the complete business process design
- Deploy valid designs to a runtime repository

The designs created using the visual tool are captured in a single design-time repository schema, making it easy to back up and reuse integrations that have been designed. A sophisticated set of validation rules allows analysts to validate a design prior to deployment, avoiding many runtime errors.

The integration manager (or runtime) of Oracle Application Server ProcessConnect receives valid business process definitions and executes them by invoking a number of services to do the following:

- Manage events
- Carry out translations and transformations
- Validate and route information flowing through the system
- Correlate events

The integration manager writes all state data associated with existing business processes into a runtime repository. By querying this runtime repository, Oracle Application Server ProcessConnect provides sophisticated facilities to monitor events and business processes and to optimize business processes. Further, all data and events in the system are translated (syntactically converted) and transformed (semantically converted) into a common representation that is independent of the format of any system. This enables organizations to have a much more efficient way to find enterprise-wide business process intelligence.

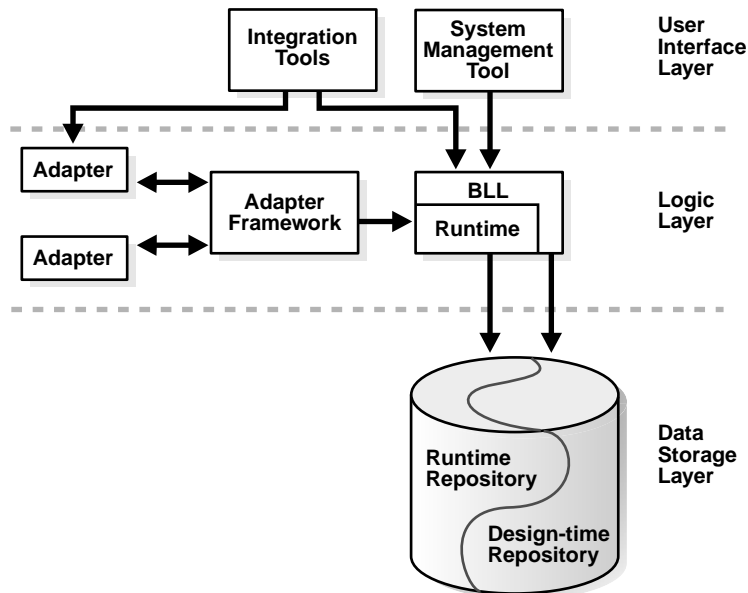
Finally, Oracle Application Server makes integration design more efficient and productive by delivering integration solutions that can be efficiently deployed and customized. Oracle Application Server ProcessConnect is also built to leverage the enterprise capabilities of the Oracle9i Database and Oracle Application Server to support both limited scope integration projects within organizations and corporate-wide integration projects that have sophisticated scalability, availability, and security requirements.

Oracle Application Server ProcessConnect Features

Oracle Application Server ProcessConnect supports the enterprise integration needs of e-business by providing the following features:

- A single, comprehensive visual design toolset
- Design-time and runtime repositories
- A runtime integration manager
- Data management facilities for translation, transformation, and validation
- A standards-based adapter framework and a comprehensive set of adapters
- Business process monitoring and business activity monitoring facilities
- Systems monitoring and management facilities

The various components of Oracle Application Server ProcessConnect are illustrated in Figure 6-1. This section discusses each of the features in greater detail.

Figure 6–1 Oracle Application Server ProcessConnect Components

Before discussing the specific features of Oracle Application Server ProcessConnect, it is important to understand the three concepts that provide the foundation of the model and methodology of Oracle Application Server ProcessConnect:

- Parties
- Events
- Roles and processes

A party is the abstraction defining the sources and targets that need to be integrated. A trading partner or an enterprise application (such as Oracle e-Business Suite or SAP) are two examples of a parties.

Events are caused by or consumed by parties and are managed within Oracle Application Server ProcessConnect. Events are indications of state changes within parties that require a reaction from other parties. For example, creating a purchase order is a state change and expresses a specific intent, namely to purchase a product.

Processes define how events are executed, for example sending an event from a trading partner to a back-end application, and requiring user approval along the way. Events are often referred to as business activities. These three concepts and the

other elements associated with specifying an integration are defined using a single, HTML-based visual toolset.

Visual Design Toolset

Oracle Application Server ProcessConnect provides a single, HTML-based visual toolset for modeling, monitoring, and administration. Specifically, the modelling facilities enable analysts to do the following:

- Design or model all the elements associated with defining an integration (or business process) from a single user interface
- Add profiles that define the way in which the designed integration communicates with the parties being connected

The monitoring facilities enable analysts to monitor the state of events (business activities), integrations (business processes), and agreements in the system. The administration facilities enable analysts to configure various facilities within the design-time environment and the runtime environment. The specific features provided by the visual toolset include:

- **Modeling events and business processes:** The modeling facilities allow analysts to define data types, events, roles and business processes, translation maps (or syntactic conversion of data types and events), and transformation maps (or semantic conversion of data types and events). It also provides facilities to model additional constructs such as correlations, namespaces, and interactions. All elements that are modeled using the visual toolset are captured as metadata in the design-time repository.
- **Modeling profiles and agreements:** Profiles are used to model the way in which parties being integrated such as trading partners, packaged applications, or legacy systems communicated with the business process or integration. Each party that is being integrated is modeled using the concept of an agreement: a single business process has specific agreements with each of the parties with which it communicates. A party may itself have multiple agreements, different agreements being used for different business processes in which it participates. A business process coupled with a set of agreements constitutes a configuration. The integration manager receives valid configurations and executes them.
- **Deploying to the runtime environment:** Once an integration is designed, analysts use a set of deployment screens to deploy the design to the runtime repository. Two steps are involved in deploying an integration:
 - Validation invokes a set of rules to determine whether the integration that has been modeled is valid both syntactically and semantically. Validation

occurs prior to deployment and Oracle Application Server ProcessConnect uses a number of sophisticated validation rules to allow analysts to correct their designs before deployment.

- Once validated, Oracle Application Server ProcessConnect creates the necessary metadata and objects in the runtime repository that are then interpreted by the integration manager to execute the integration.
- **Monitoring business activities and processes using reports:** When accepting events and executing business processes, the Integration Manager writes all state information associated with processing these events and processes to a runtime repository. By querying this repository, Oracle Application Server ProcessConnect provides facilities to monitor the state of events, business processes, and agreements. These facilities are described in greater detail later in this section.
- **Configuring the system for administration:** Finally, Oracle Application Server ProcessConnect is also integrated with Oracle Enterprise Manager for systems monitoring and administration purposes. Oracle Application Server ProcessConnect provides facilities to configure the runtime environment, communication protocols, adapters, and logging facilities. Oracle Application Server ProcessConnect can also be used to start, stop, and restart instances of the Integration Manager and to monitor the system for status, faults, performance, and resource consumption.

Design-Time and Runtime Repositories

Oracle Application Server ProcessConnect captures the integration modeled using the visual modeling toolset as metadata in a design-time repository. The design-time repository allows analysts to easily modify and reuse designs, making them more productive. Once an integration has been designed and validated, the act of deployment results in the creation of the necessary metadata in the runtime repository for the runtime engine or Integration Manager to execute the integration. Separating the design-time and runtime repositories allows analysts to modify business process definitions without affecting the runtime behavior until a new business process definition is actually deployed. Additionally, it also allows the runtime system to work against a repository that has been designed for optimal performance and scalability. Following are the major facilities provided by the design-time and runtime repositories:

- **Design-time repository:** The design-time repository consists of metadata that captures the definition of an entire integration including events, roles and business processes, parties and agreements, translations and transformation maps, event validation rules, and communication protocols and semantics. All modeling data is captured in the design-time repository in a single normalized

relational schema providing two important benefits: versioning and life cycle management. The design-time repository allows the user to do the following:

- Reuse existing designs in new integrations
- Negotiate agreements with parties
- Validate and check the integration prior to deployment

Once the design has been validated, it is deployed. The act of deploying a business process signifies the end of design and modeling an integration configuration. The versions of the metadata in the configuration are frozen after successful deployment. Any change to this metadata in the configuration are frozen after successful deployment. Any change to this metadata will create a new version, enforcing model consistency.

- **Runtime repository:** The deployment of a business process results in the creation of the necessary metadata in the runtime repository for the Integration Manager to execute the integration. The runtime repository consists of a complete set of metadata necessary for a given integration. Any change to the business process configuration must be made at design time and then re-deployed. This approach ensures runtime consistency because every runtime change by definition has an equivalent design-time change. Once deployed, the Integration Manager captures all runtime state including instance data (associated with event instances and process instances) and execution state in the runtime repository. The runtime repository can then be queried using reporting facilities described below for the purposes of business activity monitoring and business process monitoring.

Runtime Integration Manager

The integration manager or runtime environment provides the technology stack with the different integration components required to execute the modeled integration.

The integration manager coordinates the execution of business processes that span parties. When a party sends an event to another party, the information is received from the adapter framework within Oracle Application Server ProcessConnect in the form of an Oracle Record.

After receiving an Oracle Record from the adapter framework, the coordination described in TABLE occurs:

Table 6–1 Integration Manager Coordination

Key	Description
1	The event map manager is invoked to determine the correct event type.
2	The event manager creates a representation of the event in the runtime repository.
3	Validation is invoked before the event instance is created to prevent the processing of invalid events. If the event instance is invalid, it is rejected within the same synchronous invocation of the adapter. If the native event instance is valid, it is stored in the runtime repository for further processing (at the same time, providing auditing).
4	The integration manager detects a new event instance in the runtime repository and informs the role manager. The role manager must either initiate a new role (or process) or continue with an existing one.
5	The correlation manager is invoked to determine if this event instance is correlated to an event instance used in any of the already-initiated roles. If this is the case, the event is passed to the existing instance for processing. If not, a new role or process is initiated to process this new event.
6	In the translation binding role, a translation step translates the syntax of the event into the common syntax defined for the business process.
7	In the transformation binding role, a transformation step processes the event further to transform its semantics into the common semantics associated with the business process. After transformation, the role manager completes processing the event.
8	The business process manager is then invoked to process the business process. After processing the business process, the same architectural behavior applies in the outbound direction.
9, 10	The party manager and agreement manager look up which party and which role to use in order to apply the necessary transformation and translation when passing the event to the receiving party.

Adapters and Adapter Framework

Adapters provide the facilities to connect the integration manager or runtime with the parties that are being integrated. Oracle Application Server ProcessConnect provides an adapter framework and a comprehensive set of adapters to access a variety of different kinds of systems.

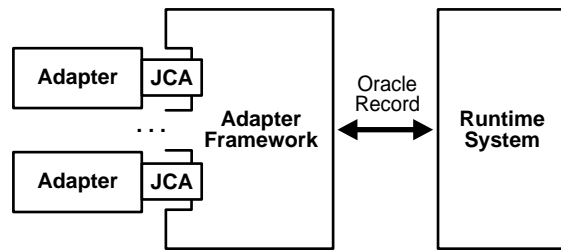
Adapters

The interfaces provided by all the parties involved in an integration vary considerably. Parties can expose many different interfaces, such as synchronous APIs, asynchronous APIs, databases, screen scraping, transactional behavior, and

JCA. Adapters know about one or more interfaces that belong to one or more specific parties. There are three types of adapters:

- Application adapters know how to connect to a specific application and exchange wire messages with it. Examples of application adapters are an SAP adapter or a Siebel adapter.
- Technology adapters know how to connect to a specific technology. Examples of technology adapters are a database adapter and an Advanced Queuing (AQ) adapter.
- B2B protocol adapters are general-purpose adapters that can connect to a trading partner using standard B2B protocols, such as RosettaNet and EDI. Figure 6–2 shows the adapter framework and several adapters. In the inbound case, an adapter is responsible for receiving the wire message and using the JCA to pass information in a standard format called an Oracle Record to the adapter framework. In the outbound case, the opposite must be accomplished.

Figure 6–2 Adapter Framework, Adapters, and Integration Manager (Runtime)



Adapter Framework

The adapter framework is the process that manages the adapters and is responsible for getting information (in the form of Oracle Records) into and out of the integration manager. The key benefit of this architecture is that the integration manager receives every Oracle Record in exactly the same way, while the adapters deal with the specific differences associated with different kinds of parties (such as different protocols, synchronous and asynchronous communication, and so on). The adapter framework is completely stateless, which enables recovery after fail over. Multiple adapter framework instances can also run in parallel to provide scalability. Figure 6–2 shows the adapter framework in detail.

Data Management

The integration manager invokes a set of data management services to translate datatypes and event instances, transform them, and validate the structure and semantics of datatypes and events. The data translation and data transformation facilities provide the following capabilities:

- **Translation or syntactic conversion:** Data translation services convert datatypes and events from their native representation in the format of the party being integrated into an XML representation that is specific to the integration platform. Oracle Application Server ProcessConnect provides translators to translate XML datatypes represented as XML schemas. Oracle Application Server ProcessConnect also provides a scripting language called the data definition description language (D3L) for defining translators to translate any other binary datatype including ASCII and EBCDIC.
- **Transformation or semantic conversion:** Data transformation services convert the semantics of the datatypes and events from the semantics of the party being integrated into a common semantic format that can be defined by the user. Oracle Application Server ProcessConnect provides several prebuilt transformation functions, including both simple functions (such as copy, concatenate, split, substr, and replace operations) and complex functions (such as nested objects, arrays, iterators, and multiple rollups). Boolean conditions determine which transformation map to apply. Domain value mapping facilities and extensibility through facilities to call out to custom coded rules in Java and ASL-T all combine to provide a sophisticated transformation service to convert datatypes and events into a common semantic model. By combining the translation and transformation facilities, datatypes and events can be syntactically and semantically converted from their representation in a specific party's format into a common model chosen by the user.
- **Validation of events and datatypes:** While designing a business process and prior to deployment, analysts can use the validation facilities of the integration manager to validate the structure and semantics of both datatypes and events. Oracle Application Server ProcessConnect provides a variety of prebuilt validation functions, such as range checking and lookup patterns, and extensibility through custom-coded rules in Java or XPath. Validation rules can be applied when agreements are imported, business process definitions are exported, and business processes are deployed to the integration manager.

Business Process and Business Activity Monitoring

The integration manager writes all state information, including event instances and business process state, into a runtime repository. The runtime repository has an optimized schema design for better performance. You can use a single set of business process and activity monitoring screens to design a set of reports that query the runtime repository for information about the state of specific events, alerts, processes, collaborations, and agreements. Oracle Application Server ProcessConnect provides three types of reports:

- Status reports that provide the status of an event or process
- Historical reports that combine the statuses of a number of events and business processes over a period of time
- Aggregate reports that provide aggregate conditions (such as average, sum, greater than, and less than) across a set of events or business processes

To summarize, Oracle Application Server ProcessConnect provides three types of monitoring facilities:

- **Event reports:** Event reports enable analysts to monitor business activities. This enables analysts to determine the state of one or more events based on the type of event, the originating and terminating parties, the business process integrating these parties, the content of the events, and the time period during which the event may have occurred. For instance, this enables an analyst to track the state of all purchase orders greater than \$50,000 sent from an Oracle11i order management application to an SAP general ledger during the previous five days.
- **Business process reports:** Business process reports similarly enable analysts to monitor business processes to determine the state of one or more business processes based on the parties being connected (the profiles), the specific agreements that govern how the business process communicates with these parties, and the time period during which the business process may have originated.
- **Agreement reports:** Agreement reports enable analysts to determine which trading partners or applications have valid agreements and which business processes can communicate using these valid agreements.

Both the event and business process monitoring facilities provide the ability for users to track domain alerts and to define specific business processes to process these alerts. For instance in enable analysts to do the following:

- Define an alert if there are more than ten purchase order over \$50,000 waiting to be processed
- Alert an administrator if such a condition occurs
- Define a specific business process that can be invoked to expedite the processing of these purchase orders

Systems Monitoring and Management

While Oracle Application Server ProcessConnect provides business activity and business process monitoring facilities targeted at business analysts, it also provides systems level monitoring and management facilities targeted to systems administrators. The components of Oracle Application Server ProcessConnect are all monitored from a systems management and monitoring point of view using Oracle Enterprise Manager. Oracle Enterprise Manager provides a number of important facilities, including:

- **Systems monitoring facilities:** Oracle Enterprise Manager provides facilities to monitor all components of Oracle Application Server ProcessConnect, including the following:
 - Runtime and all its services
 - Adapter framework and adapters
 - Runtime repository for status (up/down), faults, performance, and resource usage (CPU and memory consumption)

In addition to raising domain alerts, Oracle Application Server ProcessConnect also provides facilities to raise system alerts. System alerts such as a notification that an adapter has failed can be raised to Oracle Enterprise Manager or passed to a pager or e-mail system to alert system administrators. Configurable error logging enables administrators to configure only specific components to generate error logs and to browse these logs efficiently.

- **System management and administration:** Oracle Enterprise Manager also provides administrators with a complete set of facilities to create one or more instances of the integration manager, to configure them, to start and stop one or more instances and their components (such as stopping just a specific adapter), and to optimize the instances for overall latency and throughput of events through the system.

Business to Business (B2B) Connectivity and Prepackaged Integrations

A specific type of adapter provided by Oracle Application Server ProcessConnect is the B2B connectivity facility (or B2B adapter). The B2B adapter is designed with an extensible and layered architecture that manages several functions:

- **Transport:** The B2B adapter can use a variety of different wire protocols to transport documents between trading partners. With RosettaNet, Oracle provides facilities to communicate using RosettaNet over HTTP.
- **Packaging and signing:** Packaging and signing facilities can package a B2B message with different envelope formats and security surrounding the document payload. With RosettaNet, Oracle provides facilities to add secure multipurpose Internet mail extensions (S/MIME) packaging to the message.
- **Document exchange:** Document exchange services can format the payload into an appropriate B2B format such as RosettaNet business object documents (BODs).
- **Collaboration:** Facilities to represent collaborations are provided where a sequence of activities and events, such as messages sent and acknowledgements received can be modeled. For instance, these collaboration facilities represent the public processes between two trading partners communicating using RosettaNet partner interface processes (PIPs).

Further, the Oracle Application Server ProcessConnect model-driven facilities also enable analysts to develop prepackaged integrations or business flows that can then be easily customized. For instance, analysts or consultants can design a business process that automates the flow of a purchase order from an Oracle order management application to an SAP general ledger. The predefined business process involved the definition of the business process and the associated translation and transformation maps. These prebuilt definitions can be efficiently captured as models in the design-time repository. The models can be easily exported and transferred to a customer site where they can be efficiently customized. For instance, Oracle prepackages three business processes to process purchase orders (for RosettaNet PIPs 0A1, 3A4, and 3A6) with Oracle Application Server ProcessConnect. These involve the business process definitions and the associated transformation and translation maps to map both the public process and private process (for Oracle Applications 11.5.8) and the necessary translation and transformation maps to map RosettaNet BODs to open applications group (OAG) definitions.

Oracle Application Server ProcessConnect Architecture

Oracle Application Server ProcessConnect provides a components-based, scalable, and reliable transactional architecture that implements the facilities introduced in the "Oracle Application Server ProcessConnect Features" section. The architecture follows the traditional layered approach of a user interface layer, a logic layer, and a data storage layer.

Oracle Application Server ProcessConnect is designed to support a variety of different deployment topologies to integrate parties with each other, including the following:

- Point-to-point integration, where only two parties are being integrated
- Hub-and-spoke model integration, where several different parties can be integrated with each other
- Hosted model integration, where multiple parties are connected with a hosted facility that serves as an integration hub between these parties

By providing a single model-driven facility to define integrations and a methodology independent of the syntax and semantics of any party being integrated, Oracle Application Server ProcessConnect enables an integration to be defined once and deployed in a variety of architectures, independent of the deployment topology being used.

Further, Oracle Application Server ProcessConnect also leverages the facilities provided by Oracle Application Server for scalability. The integration manager is a Java process that is stateless (all state being written to the runtime repository). Multiple instances of the integration manager can be started and used to increase event throughput through the system. These instances can be clustered together for load balancing, scalability, and fail over (for high availability). In addition, the adapter framework is designed for scalability. The adapters are multithreaded, and one or more instances of both the adapter framework and the adapters can be started with appropriate facilities for load balancing and fail over. Finally, Oracle Application Server ProcessConnect leverages the security facilities of Oracle Application Server for encryption, secure packaging and signing, and auditing and nonrepudiation.

See Also: *Oracle Application Server ProcessConnect User's Guide* for additional information on these Oracle Application Server ProcessConnect facilities

Oracle Application Server ProcessConnect Summary

To summarize, organizations need business process automation solutions to design, monitor, and optimize business processes spanning multiple systems and applications within an organization and with trading partners. Oracle Application Server ProcessConnect provides a single, standards-based, event-driven business process automation solution that is designed to integrate data, applications, and business processes within the enterprise (EAI) and between companies (B2B integration). Oracle Application Server ProcessConnect is built as a fundamental component of Oracle Application Server and uses a number of standards to integrate systems with each other. A fundamental Oracle Application Server ProcessConnect design element is the ability to integrate systems using a common data format and syntax, and common semantics that are associated with the business process and are independent of the data and process semantics of any party being integrated. A second fundamental element of the Oracle Application Server ProcessConnect design is that it uses a model-based approach to integration as compared to a programmatic approach. Analysts use three concepts to model the different elements associated with an integration:

- Events
- Roles and processes
- Agreements

Analysts use a single visual design tool to do the following:

- Model these three elements
- Add the necessary data translations, transformations, and correlations to complete the definition of the business process
- Save the design in a design-time repository
- Validate the complete business process design
- Deploy valid designs to a runtime repository

The integration manager (or runtime) receives valid business process definitions and executes them by invoking a number of services to do the following:

- Manage events
- Carry out translations and transformations
- Validate and route information flowing through the system
- Correlate events

The integration manager writes all state data associated with existing business processes into a runtime repository. Oracle Application Server ProcessConnect provides a number of sophisticated monitoring tools to monitor events (or business activities) and business processes. Finally, Oracle Application Server ProcessConnect provides users with four important benefits:

- A single integrated product for integration within the enterprise and with trading partners
- A model-driven approach that enhances the productivity of analysts integrating systems
- The ability to maintain and upgrade integrations easily
- The facilities to comprehensively monitor business processes and business activity

Oracle Application Server Infrastructure

This chapter provides an overview of the Oracle Application Server Infrastructure. The topics include:

- Introduction to Oracle Application Server Infrastructure
- Oracle Application Server Infrastructure Architecture

Introduction to Oracle Application Server Infrastructure

Oracle Application Server provides an industry standards-based application deployment platform and security and management facilities to simplify all aspects of application deployment. Oracle Application Server Infrastructure is a comprehensive deployment platform designed to streamline application deployment by leveraging a single security, directory, and product metadata framework for all applications.

Oracle Application Server Infrastructure includes the Oracle Identity Management infrastructure, which provides security life cycle management for network entities such as users, devices, processes, and applications. It provides a comprehensive, integrated security framework to support all Oracle Application Server components, as well as third party and custom applications deployed on Oracle Application Server. The framework is based on Oracle Application Server Single Sign-On for authentication, Oracle Internet Directory for authorization, user provisioning, password policy and delegated administration services, and directory integration and synchronization, and Oracle Application Server Certificate Authority to manage X.509v3 certificates supporting PKI-based (strong) authentication.

By providing an integrated infrastructure, Oracle Application Server reduces the time required to develop Internet applications, makes these applications more reliable when deployed, and lowers the total deployment cost.

What is Oracle Application Server Infrastructure?

Oracle Application Server Infrastructure is an installation type that provides centralized product metadata and security services, configuration information, and data repositories for middle tier installations. The middle tier instances typically use the Infrastructure for two main services:

- **Product Metadata Service:** All of the product metadata required by the Oracle Application Server middle tier instances is bundled as part of the infrastructure. Product metadata is not accessed directly by customer applications. The Product Metadata Service is provided by Oracle Application Server Metadata Repository, which middle tier instances can use as a centralized component repository and leverage for product metadata lookups.
- **Identity Management Services:** Identity Management Services provide a consistent security and identity management model for all Oracle Application Server applications. It also provides a single source of security metadata containing all administration and user privileges. Middle tier components use the Identity Management Services to increase security, centralize the authentication services, and manage passwords. The Identity Management Services are provided by the Oracle Identity Management infrastructure and its components.
- **Management Service:** The Management Service in the Oracle Application Server 10g (9.0.4) Infrastructure is used to support the Distributed Configuration Management (DCM) tool. DCM stores information in the metadata repository.

Oracle Application Server Infrastructure Components

Oracle Application Server contains the following components:

- Oracle Application Server Metadata Repository
- Oracle Identity Management
 - Oracle Application Server Single Sign-On
 - Oracle Internet Directory and its components
 - Oracle Application Server Certificate Authority

In addition to these components, dedicated Oracle HTTP Server and Oracle Application Server Containers for J2EE (OC4J) instances are also installed with Oracle Application Server Infrastructure. These instances are used by the various components of the Infrastructure to service requests.

Oracle Application Server Metadata Repository

Oracle Application Server Metadata Repository is an information store that enables both infrastructure and middle tier instances to manage and configure their components in an optimal way. Oracle Application Server Metadata Repository can be installed into either a new or an existing database. When you install the Metadata Repository into a new database, infrastructure installs an Oracle9i Enterprise Edition database server that contains the demo data, schemas, and metadata required by most of the Oracle Application Server middle tier instances. You can also choose to install the Metadata Repository and its associated data, schemas, and metadata into an existing database, using the Repository Creation Assistant tool.

There are two general types of data that can be stored in a database: customer or application data, and metadata. Customer or application data is user data created by a client application. It is accessed directly by the client application.

Metadata, by comparison, includes component-specific information that is accessed by the Oracle Application Server middle tier or Infrastructure components as part of their application deployment. The end user or the client application does not access this data directly. For example, a Portal application on the middle tier accesses the Portal metadata as part of the Portal page assembly aggregation. Metadata also includes demo data for many Oracle Application Server components, such as data used by the Order Management Demo for BC4J.

The Oracle Application Server Metadata Repository stores three main types of metadata:

- Management metadata
- Identity Management metadata
- Product metadata

Table 7-1 shows the Oracle Application Server components that store and use these types of metadata during application deployment.

Table 7-1 Metadata and Infrastructure Components

Type of Metadata	Infrastructure Components Involved
Management metadata	Distributed Configuration Management (DCM)
Identity Management metadata	Oracle Application Server Single Sign-On (SSO), Oracle Internet Directory, Oracle Application Server Certificate Authority

Table 7–1 Metadata and Infrastructure Components

Type of Metadata	Infrastructure Components Involved
Product metadata (includes demo data)	Oracle Application Server Metadata Repository

Using Oracle Application Server Infrastructure with Middle Tier Installations

Oracle Application Server provides three middle tier install options. Oracle Application Server Metadata Repository is required for all installation types except for J2EE and Web Cache.

- J2EE and Web Cache:** Installs Oracle HTTP Server, Oracle Application Server Containers for J2EE, Web Cache, Web Services, BC4J, UDDI, and Oracle Enterprise Manager Application Server Control. Installing Oracle Application Server Metadata Repository is optional, but recommended. Installing Oracle Application Server Infrastructure allows you to create database-managed OC4J clusters, and enables you to use Single Sign-On and other identity management protections available through the Infrastructure.
- Portal and Wireless:** Installs all components of J2EE and Web Cache, plus Portal, Syndication Services, Ultra Search, and Wireless. Installing Oracle Application Server Metadata Repository is required.
- Business Intelligence and Forms:** Installs all components of J2EE and Web Cache, Portal and Wireless, plus Forms, Reports, Discoverer, and Personalization. Installing Oracle Application Server Metadata Repository is required.

Oracle Application Server Integration components, such as Oracle Application Server ProcessConnect, Oracle Application Server InterConnect, and Workflow are installed on top of any of these middle tier installation options.

Oracle Content Management Software Developer Kit (CM SDK) is also installed separately in addition to one of the standard middle tier installations.

The Distributed Configuration Management (DCM) component enables middle tier management, and stores its metadata in the Metadata Repository for both the Portal and Wireless and the Business Intelligence and Forms install options. For the J2EE and Web Cache install type, by default DCM uses a file-based repository. If you choose to associate the J2EE and Web Cache install type with an Infrastructure, the file-based repository is moved into the Metadata Repository, enabling database-managed Oracle Application Server clustering.

Oracle Application Server Metadata Repository Contents

Within Oracle Application Server Metadata Repository, there is metadata for many Oracle Application Server components. Oracle Application Server Metadata Repository contains metadata for the following components:

- Distributed Configuration Management (DCM)
- Oracle Internet Directory (OID)
- Oracle Application Server UDDI Registry (for Web Services)
- Oracle Business Components for Java (BC4J)
- Oracle Application Server Portal
- Oracle Application Server Syndication Services
- Oracle Ultra Search
- Oracle Application Server Single Sign-On (SSO)
- Oracle Application Server Wireless
- Oracle Application Server Discoverer
- Oracle Application Server Integration
- Oracle Workflow
- Oracle Application Server Web Services
- Online Analytical Processing (OLAP)
- Oracle Application Server Certificate Authority

For information related to the metadata for each component, please see the *Oracle Application Server 10g Administrator's Guide*.

Oracle Identity Management

Identity management is the process of managing the security life cycle for network entities in an organization, and most commonly refers to the management of an organization's application users. Oracle Identity Management is an integrated infrastructure that Oracle products use for distributed security. The Oracle Identity Management infrastructure includes the following components:

- Oracle Application Server Single Sign-On
- Oracle Internet Directory and its components

- **Oracle Application Server Certificate Authority**

The following sections contain brief overviews of each Identity Management component. For more information on the Identity Management infrastructure, please see Chapter 11, "Security and Identity Management".

Oracle Application Server Single Sign-On

Oracle Application Server Single Sign-On enables users to access multiple Oracle Application Server applications with a single password. Using Single Sign-On, users can log in to Oracle Application Server and gain access to all applications for which they are authorized, without requiring them to re-enter a user name and password for each application. Oracle Application Server Single Sign-On retrieves user information from Oracle Internet Directory, and LDAP v3 compliant directory.

Oracle Internet Directory

Oracle Internet Directory (OID) is the Oracle implementation of Lightweight Directory Access Protocol (LDAP), version 3. Application server instances, components, and infrastructures store security and management information in the directory. Oracle Internet Directory serves the Oracle Application Server environment by providing authentication and a centralized user provisioning model whereby you can create and manage users on an enterprise scale. It provides a single source of access to administration information such as Oracle Application Server instance objects, Oracle Application Server instance configuration, Oracle Application Server component schema mappings, application group information by components (such as Portal). When users log in, they are authenticated once by Oracle Application Server Single Sign-On against their OID credentials, and afterwards can access multiple applications seamlessly.

Directory Integration and Provisioning Directory Integration and Provisioning (DIP) is a component of Oracle Internet Directory. It permits synchronization between Oracle Internet Directory and other directories; user repositories and automatic provisioning services for Oracle components; applications; and third-party applications through standard interfaces. Typically, provisioning an application means creating and managing separate user accounts and their privileges.

Delegated Administration Services Delegated Administration Services (DAS) is a component of Oracle Internet Directory. It allows users and application administrators to perform trusted proxy-based administration of directory information. You can assign administrative responsibilities according to business requirements, and control security policies for different components of the enterprise.

Oracle Application Server Certificate Authority

The Oracle Application Server Certificate authority manages and publishes X.509v3 certificates to support PKI-based (strong) authentication methods. OracleAS Certificate Authority also serves as an assertion services, since the certificates it generates are assertions about a network's identity and its entitlements.

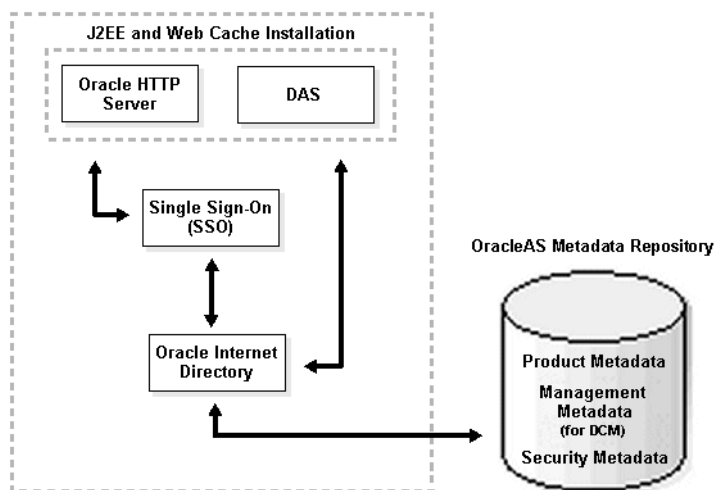
See Also: *Oracle Identity Management Concepts and Deployment Planning Guide*

Oracle Application Server Infrastructure Architecture

Oracle Application Server Infrastructure provides centralized product metadata and security, configuration information, and data repositories for middle tier installations.

Figure 7-1 shows how the Oracle Application Server Infrastructure components work closely together to provide these services to the middle tier Oracle Application Server instances.

Figure 7-1 Oracle Application Server Infrastructure Components



- Oracle Application Server Metadata Repository contains three types of metadata:

- Product Metadata for middle tier instances
- Management Metadata for Distributed Configuration Management
- Security Metadata for Oracle Internet Directory and Oracle Application Server Single Sign-On
- Oracle Application Server Single Sign-On uses Oracle HTTP Server, and the Delegated Administration Service (DAS) uses the Infrastructure OC4J instance, which are part of the J2EE and Web Cache instance that is embedded in Oracle Application Server Infrastructure.
- Oracle Application Server Single Sign-On stores user information in Oracle Internet Directory.

Oracle Application Server Infrastructure has several deployment architectures, which makes it easy to fit it into an existing enterprise deployment methodology. Some of the most frequently used deployment topologies for both Oracle Application Server Infrastructure itself and for applications which use Oracle Application Server Infrastructure are discussed in Chapter 12, "Recommended Topologies".

Part II

Deployment

Part II discusses topics related to application deployment, and contains the following chapters:

- Chapter 8, "Scalability, High Availability, and Recovery"
- Chapter 9, "Performance and Caching"
- Chapter 10, "System Management"
- Chapter 11, "Security and Identity Management"

Scalability, High Availability, and Recovery

This chapter provides an overview of high availability and scalability solutions provided by Oracle Application Server. The topics include:

- Scalability
- High Availability
 - Application High Availability
 - Oracle Application Server Middle Tier High Availability
 - Oracle Application Server Infrastructure High Availability
- Recovery Solutions

Scalability

Scalability is the ability of a system to provide throughput in proportion to, and limited only by, available hardware resources. A scalable system is one that can handle increasing numbers of requests without adversely affecting response time and throughput.

The following sections describe Oracle Application Server components and features you can use to increase the scalability of your system.

Web Cache Clusters

You can configure multiple **instances** of OracleAS Web Cache to run as independent caches, with no interaction with one another. However, to increase the **availability** and **scalability** of your Web Cache, you can configure multiple instances of OracleAS Web Cache to run as members of a cache cluster. A cache cluster is a

loosely related set of Web cache instances working together to provide a single logical cache.

See Also: *Oracle Application Server 10g Performance Guide*

Oracle Application Server Containers for J2EE (OC4J) Routing

The `mod_oc4j` module of the Oracle HTTP Server (OHS) is an intelligent router that routes requests to J2EE instances. It generally deals with stateless requests, since stateful requests are forwarded to the OC4J instance that served the previous request.

There are eight different intelligent routing algorithms that you can use, depending on the type and complexity of routing you need. Table 8-1 describes each type of routing and the algorithms available.

Table 8-1 Intelligent Routing Algorithms Matrix

Routing Style	Simple Algorithm	Algorithm with Local Affinity	Algorithm with Routing Weight
Round Robin	All OC4J processes (remote and local) are placed in an ordered list. Oracle HTTP Server chooses an OC4J process at random for the first request. For each subsequent request, OHS forwards requests to another OC4J process in round robin style.	There are two lists of requests, one each for local and remote OC4J processes. OHS always chooses a local process, using the simple random mechanism, until the local process list is empty. Then it selects a process from the remote processes list in a simple round robin style.	Each machine is assigned an integer that represents its ability to service requests. OHS uses this routing weight to decide which machine to route requests to, and chooses an OC4J process in round robin style from the OC4J processes on that machine.

Table 8–1 Intelligent Routing Algorithms Matrix

Routing Style	Simple Algorithm	Algorithm with Local Affinity	Algorithm with Routing Weight
Random	All OC4J processes (remote and local) are placed in an ordered list. For every request, OHS chooses an OC4J process at random and forwards the request to that instance.	There are two lists of requests, one each for local and remote OC4J processes. OHS always chooses a local process, using the simple random mechanism, until the local process list is empty. Then it selects a process from the remote processes list in a simple random style.	Each machine is assigned an integer that represents its ability to service requests. OHS uses this routing weight to decide which machine to route requests to, and chooses an OC4J process at random from the OC4J processes on that machine.
Metric-based	All OC4J processes (remote and local) are placed into an ordered list. Through an asynchronous heartbeat, OC4J processes communicate to OHS how busy they are. OHS uses this information as a routing weight, and sends more requests to the OC4J processes that are less busy.	There are two lists of requests, one each for local and remote OC4J processes. OHS always chooses a local process, using the simple metric mechanism, until the local process list is empty. Then it selects a process from the remote processes list using the simple metric algorithm.	NA

Choosing an Intelligent Routing Algorithm

Each of these intelligent routing algorithms has advantages and drawbacks. Choosing which one to use will depend on the type of environment you are using. Use the following guidelines to help determine which algorithm is best for your enterprise:

- For a loose cluster setup (multiple identical machines with OHS and OC4J), the round robin with local affinity algorithm is preferred. In this setup, an external router distributes requests to multiple machines running OHS and OC4J. In this case OHS gains little by routing to other machines, except in the extreme case that all OC4J processes on the same machine are dead.

- For a tiered deployment (where one tier of machines contains OHS and another contains OC4J), the preferred algorithms are simple round robin and simple metric-based. To determine which of these two is best in a specific setup, you may need to experiment with each and compare the results. This is required because the results are dependent on system behavior and incoming request distribution.
- For a heterogeneous deployment (where the different OC4J machines vary in power), a weighted round robin algorithm is preferred. You may need to tune the number of OC4J processes to achieve the maximum benefit. For example, a machine with a weight of 4 gets 4 times as many requests as a machine with a weight of 1, but may not contain 4 times as many OC4J processes. If the metric computation provides an effective way of modelling the application load, you may also consider using the simple metric-based algorithm.

Enterprise JavaBean (EJB) Client Routing

In EJB client routing, EJB classes take on the routing functionality that `mod_oc4j` provides for Oracle HTTP Server. Using the Active Components for Java (AC4J) architecture, EJBs can interact in a loosely-coupled fashion. This provides support for reliable asynchronous, disconnected, one-way request and response interactions, without the complexity of JMS programming. It automatically routes service requests to the appropriate service provider, and provides automatic security context propagation, authorization and identity impersonation. It also provides automatic exception routing and handling, which is integrated into the EJB framework.

High Availability

The **availability** of a system or any component in that system is defined by the percentage of time that it works normally. For example, a system that works normally for twelve hours per day is 50% available. A system that has 99% availability is down 3.65 days per year on average. Critical systems may need to meet exceptionally high availability standards, and experience as little as four to five minutes of downtime per year.

Oracle Application Server is designed to provide a wide variety of high availability solutions, ranging from load balancing and basic clustering to providing maximum system availability during catastrophic hardware and software failures.

The underlying high availability paradigm for Oracle Application Server is clustering. You can create different types of clusters depending on which Oracle Application Server components are involved.

High Availability for Planned Maintenance

Every system requires maintenance, and minimizing downtime during maintenance and upgrade operations is important for ensuring high availability. Oracle Application Server provides the following capabilities to minimize planned downtime for these activities.

Rolling Upgrade

Oracle Application Server supports the upgrade of Infrastructure and middle tier instances from Oracle9i Application Server Release 2 (9.0.2) or (9.0.3) to Oracle Application Server 10g (9.0.4) with minimal operational impact. A new archiving capability enables one system configuration to be captured and then re-applied to another system. This allows one system to be upgraded and tested before its configuration is cloned. Similarly, Web Cache clusters allow incremental update of configuration to the clustered instances.

Cloning

You can clone an existing Oracle Application Server instance, creating an additional instance with the exact same configuration. With cloning, you reduce the possibility of introducing configuration errors during installation and setup of the new instance.

Application High Availability

Oracle Application Server provides several features for ensuring application-level high availability. You can design your applications to take advantage of these capabilities when deployed on Oracle Application Server. This section discusses session replication, session persistence, and distributed caching.

Session Replication

Session replication is a high availability solution that is implemented in Oracle Application Server using Oracle Application Server Containers for J2EE (OC4J) islands.

On the client side, different HTTP requests from the same client can be grouped together with a cookie. The cookie contains enough information to provide continuity between client requests. However, if the server experiences a failure, the state associated with a request or cookie may be lost. There are three ways to guard against this:

- State safe applications save the state in a database or other persistent storage system, avoiding the loss of state when the server goes down. However, there is a performance cost to continually writing this information.
- Stateless applications do not have a state that needs to be carried between requests, and so are not impacted by the server going down. Any other server is able to handle the request.
- Stateful applications benefit from OC4J session state replication, which replicates the session state across all islands in the cluster automatically. This replication takes place when the session is updated.

During OC4J session state replication, if one OC4J process fails, another OC4J process that has the session state replicated to it takes over the application request. When an OC4J process fails during a stateful request, Oracle HTTP Server forwards the request in the following order:

1. If another OC4J process is active within the same application server instance, OHS forwards the request to this process.
2. Otherwise, OHS forwards the request to an OC4J process in another application server instance in the cluster.

Session Persistence

Working with EJBs, you can have session persistence, which means that the EJB's state exists beyond the lifetime of the application or the server process. Entity beans, which represent business objects in a persistent storage mechanism, can have either bean-managed persistence or container-managed persistence. For more information on EJBs and persistence, see Chapter 2, "J2EE, Web Services, and Internet Applications".

Distributed Cache

Distributed cache is a high availability solution that is implemented using Oracle Application Server Java Object Cache. Java Object Cache is an "in-process" cache of Java objects that can be used on any Java platform by any Java application. It allows

applications to share objects across requests and across users, and coordinates the life cycle of the objects across processes.

Java Object Cache in OC4J enables data replication among processes even if they do not share an island, instance, or cluster relationship. This type of replication enhances performance by caching large, shared Java objects, regardless of which application produced them. It also improves availability in the event that the sources of these objects becomes unavailable. Java Object Cache also supports object versioning, allowing different applications to have different versions of an object available.

Oracle Application Server Middle Tier High Availability

You can make Oracle Application Server middle tier instances highly available using a variety of solutions, each leveraging the availability features of single instances as well as the power of Oracle Application Server clustering.

Oracle Application Server Single Instance Availability Features

An Oracle Application Server Instance (also called an application server instance) is the set of processes required to run the configured components within an application server installation. There can be only one application server instance per application server installation. The terms installation and instance are sometimes used interchangeably; however, it is important to note that an installation is the set of files installed into an Oracle home, while an instance is a set of processes associated with those files.

Each Oracle Application Server instance has features that enhance availability and facilitate the implementation of other high availability solutions. Such features include:

A single instance is the minimum unit that can be used in any type of Oracle Application Server clustering. You can have a single instance associated with an infrastructure, a file-based repository, or neither.

- **Process death detection and restart:** Oracle Process Management and Notification (OPMN) provides process monitoring, process death detection, and process restarting for monitored processes.
- **Configuration cloning:** Distributed Configuration Management (DCM) uses a metadata repository for configuration information, allowing you to manage distributed Oracle Application Server instances.

- **Data replication:** Oracle Application Server Containers for J2EE (OC4J) islands within an instance provide Web-application-level stateful session replication. An OC4J island is a group of OC4J processes within an OC4J cluster that replicate session state among one another. These processes must have the same configuration, which is enforced by having all of the processes be part of the same OC4J instance. You can also replicate data across processes within an Oracle Application Server instance using EJB session replication.
- **Intelligent routing:** Oracle Application Server Web Cache and Oracle HTTP Server (through `mod_oc4j`) provide configurable and intelligent routing for incoming requests. By communicating with OPMN, `mod_oc4j` routes requests only to processes and components that it determines to be alive.

Oracle Application Server Clusters

An Oracle Application Server **cluster** is a set of application server **instances** configured to act in concert to deliver greater **scalability** and **availability** than a single instance can provide. While a single application server instance can only leverage the operating resources of a single **host**, a cluster can span multiple hosts, distributing application execution over a greater number of CPUs. While a single application server instance is vulnerable to the failure of its host and operating system, a cluster continues to function despite the loss of an operating system or host, hiding any such failure from **clients**.

Clusters leverage the combined power and **reliability** of multiple application server instances while maintaining the simplicity of a single application server instance. For example, **browser clients** of applications running in a cluster interact with the application as if it were running on a single **server**. With appropriate front-end load balancing, any instance in an application server cluster can serve client requests. This simplifies configuration and deployment across multiple instances and enables fault tolerance among clustered instances.

Types of Oracle Application Server Clusters

Oracle Application Server clusters are able to propagate configuration information across all cluster instances instantly when you change the configuration of any member instance. The propagation is handled through the built-in distributed configuration management (DCM) system.

You can manually configure Oracle Application Server instances to act as a cluster. However, Oracle Application Server provides two ways to manage clusters more effectively.

Centrally-managed Oracle Application Server Clusters propagate configuration information across all application server instances within the cluster, which simplifies configuration and deployment. There are two ways you can choose to manage this type of cluster: using a database repository or using a file-based repository.

Oracle Application Server Clusters Managed Using a Database Repository Oracle Application Server Clusters managed using a database repository store their metadata and configuration information in the database. This type of cluster requires the Oracle Application Server Infrastructure, since the metadata and configuration information is stored in a metadata repository that resides on an Infrastructure system.

Oracle Application Server Clusters Managed Using a File-Based Repository Oracle Application Server Clusters managed using a file-based repository designate an application server instance as the repository host. The repository host uses its file system to store the metadata repository that contains the metadata and configuration information for the cluster.

Benefits of Oracle Application Server Clustering

There are three main benefits of using clusters: scalability, availability, and manageability. Oracle Application Server clustering provides scalability by unifying multiple application server instances spread over multiple hosts to collectively serve a single group of applications, making it possible to serve increasing numbers of concurrent users after the capacity of a single piece of hardware is exhausted.

Oracle Application Server clustering enables high availability by removing the single point of failure that a single host poses. It introduces redundancy and **failover** into the system, and protects against the loss of session state in case of process failure. It ensures that configuration information remains synchronized across instances, and enables intelligent routing with `mod_oc4j` and session replication through islands.

Oracle Application Server clusters can be managed using Oracle Enterprise Manager Application Server Control. Oracle Application Server managed clustered instances synchronize their configurations automatically, relieving the administrator of the responsibility to manually update each individual instance. Oracle Application Server cluster management simplifies the tasks of creating and administering clusters and reduces the chance of human error corrupting the system.

Oracle Application Server also enables you to create manually configured Oracle Application Server clusters that do not require a metadata repository, and therefore have no database dependency. Manually configured clusters provide scalability and availability, but not manageability. In a manually configured cluster, the application server administrator has the responsibility for synchronizing the configuration of the application server instances.

Improving Availability with an External Load Balancer

You can use an external load balancer to improve the availability of both clustered and non-clustered Oracle Application Server instances.

Clients access the cluster through a load balancer, which hides the cluster configuration. The load balancer can send requests to any application server instance in the cluster, as any instance can service any request. An administrator can raise the capacity of the system by introducing additional application server instances to the cluster.

You can also use a load balancer to increase the availability of non-clusterable Oracle Application Server instances, such as Portal and Wireless. As long as the load balancer is configured to serve a set of instances, it will route requests accordingly.

Types of External Load Balancers

There are three main types of external load balancers you can use with Oracle Application Server instances: hardware load balancers, network load balancers, and Oracle Application Server Web Cache.

Hardware Load Balancer Hardware load balancing involves placing a hardware load balancer (such as Big-IP or Alteon) in front of a group of Oracle Application Server instances. The hardware load balancer routes requests to the instances in a client-transparent fashion.

Windows Network Load Balancer With some Windows operating systems, you can use the features of your operating system to perform network load balancing. For example, with Microsoft Advanced Server, the NLB functionality allows you to send requests to different machines that share the same virtual IP or MAC address. The servers themselves do not need to be clustered at the operating system level.

Leveraging Web Cache as an External Load Balancer OracleAS Web Cache supports content-aware load balancing and failover detection. These features ensure that cache misses are directed to the most available, highest-performing application

server. For more information on OracleAS Web Cache, see Chapter 9, "Performance and Caching".

Benefits of External Load Balancing

There are three main benefits of using clusters: scalability, availability, and manageability. Load balancing improves scalability by providing an access point through which requests are routed to one of many available instances. Instances can be added to the group that the load balancer serves to accommodate additional users.

Load balancing improves availability by routing requests to the most available instances. If one instance goes down, or is particularly busy, the load balancer can send requests to another active instance.

Load balancing improves manageability by routing application deployment and system configuration requests to the most available instances. If one instance goes down, or is particularly busy, the load balancer sends requests to another active instance.

Improving Availability with Operating System Clusters

This type of cluster involves installing Oracle Application Server on a hardware cluster created through the operating system or other clustering system solutions, such as Veritas. Operating system clustering is now supported in Oracle Application Server 10g (9.0.4). However, using operating system clusters with Oracle Application Server does not have a significant impact on system availability. Leveraging operating system clusters is similar in effect to using a simple external load balancer across a system that does not use operating system clusters.

Oracle Application Server Infrastructure High Availability

This section discusses the Oracle Application Server Infrastructure and strategies for improving Infrastructure availability.

High Availability Requirements for Oracle Application Server Infrastructure

The Oracle Application Server Infrastructure product metadata service, security service, and management service are provided by the following components, which must all be available to guarantee the availability of the Infrastructure:

- Oracle Application Server Metadata Repository

- Oracle Net listener
- Oracle HTTP Server (OHS)
- For Oracle Identity Management:
 - Oracle Internet Directory and Oracle Internet Directory monitor
 - Oracle Application Server Single Sign-On
 - Oracle Application Server Certificate Authority
 - Oracle HTTP Server for use by Single Sign-On
 - OC4J instance for use by Oracle Delegated Administration Services
- For Oracle Management Services:
 - Oracle Enterprise Manager Application Server Control

For the Infrastructure to provide all essential services, all of these components must be available. Any high availability solution must be able to detect and recover from any software failures of any of the processes associated with the Infrastructure components. Solutions must also be able to detect and recover from any hardware failures on the hosts that are running the Infrastructure.

In Oracle Application Server 10g (9.0.4), all of the Infrastructure processes, except the database and its listener, are started, managed, and restarted by the Oracle Process Management and Notification (OPMN) framework. This means that any failure of an OPMN-managed process is handled internally by OPMN. However, database process failures or database listener failures are not handled by OPMN. Also, failure of any OPMN processes leaves the Infrastructure in a non-resilient mode if the failure is not detected and resolved.

See Also: Chapter 7, "Oracle Application Server Infrastructure" for a detailed discussion of the Oracle Application Server Infrastructure

Oracle Application Server Cold Failover Clusters

Oracle Application Server Cold Failover Clusters are a high availability solution for Oracle Application Server Infrastructure that protects against system failure. All of the Infrastructure components are installed on a set of shared disks that can be mounted on any of the nodes in a hardware cluster. If the Infrastructure on the

active node fails, another node in the cluster restarts the Infrastructure automatically.

In a typical Cold Failover Cluster scenario, instead of the middle tier pointing to a specific node in the cluster, it is configured to point to a virtual hostname. Each node in the cluster has its own private hostname as well, and the active node carries the virtual hostname for the entire cluster.

Oracle Application Server Infrastructure is only active on one node in the cluster at any time. When the Infrastructure on the active node goes down, the cluster software brings up the Infrastructure on one of the inactive nodes, with the same virtual hostname as the failed node. Although there will be some minimal downtime, this allows for faster recovery times on the middle tier, as it does not need to be reconfigured to point to a new Infrastructure. From the perspective of middle tier applications, the new active node in the Cold Failover Cluster is identical to the node that failed.

Note that the inactive nodes in the hardware cluster are only inactive with respect to Oracle Application Server Infrastructure. The inactive nodes can be used for other purposes, such as running middle tier instances or other applications.

Oracle Application Server Middle Tiers in a Cold Failover Cluster Environment

Oracle Application Server 10g (9.0.4) middle tier instances can be installed on the local disks of each node in a hardware cluster. Requests across middle tiers are distributed using an external load balancer. Although this configuration does not provide middle-tier failover, it does increase availability because if a node fails, the middle tier instances on the remaining nodes will continue to function. The Infrastructure will still fail over as designed.

Oracle Application Server Active Failover Clusters

In the initial release of Oracle Application Server 10g (9.0.4), **Active Failover Cluster** is a **Limited Release** feature. Please check Oracle *MetaLink* (<http://metalink.oracle.com>) for the most current certification status of this feature or consult your Sales Representative, before deploying this feature in a production environment.

Active Failover Cluster configurations are similar to Cold Failover Cluster solutions, except that with Active Failover Clusters the Infrastructure is active on all nodes in the cluster at the same time (also known as hot failover). Active Failover Clusters provide a robust cluster architecture for the Oracle Application Server Infrastructure, and are a more transparent high availability solution than Cold Failover Clusters. Because all of the nodes are active, failover from one node to

another is quick and requires no manual intervention. This active-active setup also provides scalability to the Infrastructure that is deployed on it.

This configuration leverages the Real Application Cluster (RAC) feature of the Oracle database for running the Infrastructure database. Each node in the hardware cluster contains its own `$ORACLE_HOME`, which contains the configuration files and binaries needed to run the Infrastructure on that node. The Infrastructure installation across nodes is accomplished in one process. Additionally, all nodes access a set of shared files on the RAC database. The middle tier instances are configured to point to a load balancer in front of the hardware cluster.

Installing the Metadata Repository into an Existing RAC Database

During Infrastructure installation, you can choose to install the Metadata Repository into an existing Oracle database, rather than installing a new database. You can install the Metadata Repository into an existing RAC database to ensure that the Metadata Repository is highly available. However, this only provides high availability for the Metadata Repository; other Infrastructure components must be dealt with separately.

Recovery Solutions

In addition to failing over components to increase availability, once a failure has occurred in your system, it is important to restore the failed component or process as quickly as possible. There are three main types of recovery solutions that you can use, depending on the type and severity of the failure.

Restarting Processes

Recovering from almost all types of failures requires restarting one or more failed processes in your system. There are three process restart scenarios:

- Automatic restart of processes: The failed processes are automatically restarted. No manual intervention is required.
- Manually restart an individual process: This implies that the process failure does not affect any other middle-tier or infrastructure processes, and can be restarted individually.
- Manually restart all processes.

Many types of failures in both the middle tier and the infrastructure only require a process restart solution. Such failures include the death of OPMN, a metadata repository instance failure, or an Oracle Enterprise Manager Application Server

Control crash. For details about specific failure types and how to recover, see the *Oracle Application Server 10g High Availability Guide*

Cold Backup and Restore

Some failures require more involved recovery scenarios than simply restarting processes. In some cases, you will have to perform restoration operations based on backup procedures you have previously implemented. Cold backup and restore operations can be done for both the middle tier and the infrastructure.

- **Middle tier cold backup and restore:** Restoration of the entire Oracle Application Server middle tier, including the Oracle Home, configuration files, and database files, which were backed up after completing a clean and normal shutdown of all Oracle Application Server infrastructure processes and the metadata repository.
- **Infrastructure cold backup and restore:** Restoration of the entire Oracle Application Server infrastructure instance, including the Oracle Home, configuration files, and data base files, which were backed up after completing a clean and normal shutdown of all Oracle Application Server infrastructure processes and the metadata repository.

Failures that require the cold backup and restore solution for recovery include node failure where the node needs to be completely replaced, and the deletion or corruption of Oracle software or binary files. Failures that require this type of recovery solution also then require the manual restart of all processes. For details about specific failure types and how to recover, see the *Oracle Application Server 10g Administrator's Guide*

Online Backup and Restore

Depending on the type of failure your system is experiencing, you may need to restore your system from an online backup. There are four types of online backup and restore scenarios:

- **Middle tier online backup and restore:** Restoration of the Oracle Application Server configuration files, which were backed up while processes were up and running on the middle tier. This also includes restoring a stamped image, which may require additional steps to complete the restoration.
- **Infrastructure online backup and restore:** Restoration of the Oracle Application Server infrastructure configuration files, which were backed up after completing a proper online backup of the Oracle Application Server instance and metadata repository.

- Infrastructure metadata repository online backup and restore: Restoration of the Oracle Application Server infrastructure metadata repository taken from a proper online backup. Complete recovery is required of the database component.
- Infrastructure configuration files online backup and restore: Restoration of the Oracle Application Server infrastructure configuration files taken from an online backup.

Failures that require online backup and restore solutions for recovery include data failure in the metadata repository and deletion or corruption of Oracle Application Server component runtime configuration files. Failures that require this type of solution also then require one or more processes to be restarted. For details about specific failure types and how to recover, see the *Oracle Application Server 10g Administrator's Guide*.

Disaster Recovery

Disaster recovery refers to how a system recovers after a catastrophic site failure. Catastrophic failures include earthquakes, tornadoes, floods, and fires. On the most basic level, disaster recovery involves replicating an entire site, not just pieces of hardware or subcomponents.

The service-level requirements for disaster recovery depend on the business applications you are running. Some applications may not have any disaster recovery requirements. Others may simply have backup data tapes from which they would rebuild a new working site over a period of time. Still others may have requirements to begin operations with a few days or hours after the disaster. The most stringent requirement is to keep the services running despite the disaster.

The Oracle Application Server disaster recovery solution consists of two identically configured sites. Both sites may be dispersed geographically, and if so, they are connected via a network. When the primary site becomes unavailable due to disaster, the secondary site can become operational within a reasonable amount of time.

Client requests are always routed to the site in the production role. After a failover or switchover operation occurs due to an outage, client requests are routed to another site that assumes the production role. Each site contains identical middle tier servers, which are also identical between the two sites.

The site that is in the production role contains a production backend customer database and production Oracle Application Server Metadata Repository configured using the Cold Failover Cluster Infrastructure high availability solution

to protect from host failure. The site in the standby role contains a physical standby of the Oracle Application Server Metadata Repository managed by Oracle Data Guard. Database switchover and failover functions allow the roles to be traded between sites.

The Oracle Application Server Cold Failover Cluster high availability solution and Oracle Data Guard provide the basis of the Oracle Application Server Disaster Recovery solution. Using an Oracle Data Guard physical standby database provides disaster recovery and prevention against user error and data corruption. In the Oracle Application Server Disaster Recovery solution, Oracle Data Guard provides automatic log transport services, managed recovery, and role management between the production and physical standby databases containing the OracleAS Metadata Repository.

The Oracle Application Server Disaster Recovery solution is restricted to identical site configurations to ensure that processes and procedures are kept the same between sites, making operational tasks easier to maintain and execute. Identical site configuration also allows for a higher success rate for manually maintaining the synchronization of Oracle Application Server component files between sites.

The sites are configured in an active-passive configuration. This configuration has one primary site used for production and one secondary site that is initially passive. The secondary site is made active only after an application fails or switches over to it. Since the sites are symmetric, after an application fails or switches over, the application does not necessarily need to switch back to the original production site.

Distributed Configuration Management Archiving Feature

The Distributed Configuration Management (DCM) archiving and recovery feature allows you to take a snapshot of your system configuration. Taken at a time when everything is working properly and optimally, you can restore the system to this previous configuration in the event of a failure. In response to a catastrophic failure, the snapshot can be restored to a system in a remote location.

Performance and Caching

This chapter provides an overview of Oracle Application Server performance and **caching** features and benefits. The topics include:

- Introduction to Performance
- Overview of Caching Solutions
- Introduction to Oracle Application Server Web Cache
- Oracle Application Server Web Cache Features
- Oracle Application Server Web Cache Deployment Architecture
- Additional Caching Components

Introduction to Performance

Increasing the performance of your Web site and increasing the speed of your applications without redesigning or rebuilding the Web site are common goals. To maximize Oracle Application Server performance, all components need to be monitored, analyzed, and tuned. Performance must be built in to an application deployment; you must anticipate performance requirements during application analysis and design, and balance the costs and benefits of optimal performance.

The overall performance of an application is determined by these factors:

- How many resources are available?
- How many clients need the resource?
- How long must they wait for the resource?
- How long do they hold the resource?

The following concepts are fundamental to understanding performance:

- **Response time:** The **response time** is equal to the **service time** plus the **wait time** for a task to complete. You can increase response time performance by reducing the service time, the wait time, or both. For example, you can decrease wait time by implementing parallel processing with multiple resources, such that more resources are available to the incoming tasks. Oracle HTTP Server processes requests in this way, allocating client requests to available `httpd` processes.
- **System throughput:** System **throughput** is the amount of work accomplished in a given amount of time. You can increase throughput with a combination of reducing service time and reducing the overall response time by increasing the amount of scarce resources that are available. For example, if the system CPU is bound, then adding CPU resources should improve performance.
- **Wait time:** While the service time for a task may stay the same, the wait time will lengthen with increased **contention**. If many users are waiting for a service that takes one second, the tenth user must wait nine seconds. Reducing contention should improve performance.
- **Critical resources:** Resources such as CPU, memory, I/O capacity, and network bandwidth are key to reducing service time. Adding resources increases throughput and reduces response time.

As the number of requests rises, the time to service completion increases if the number of resources stays the same. To improve performance, you can either limit the demand rate to maintain acceptable response times, or you can add resources.

Performance Methodology

Achieving optimal effectiveness in your system requires planning, monitoring, and periodic adjustment. The first step in performance tuning is to determine the goals you need to achieve, and then design effective usage of available technology into your applications. After implementing your system, you must periodically monitor and adjust your system.

Performance Targets

Whether you are designing or maintaining a system, you should set specific performance goals so that you know how and what to optimize. If you alter parameters without a specific goal in mind, you can waste time tuning your system without significant gain.

An example of a specific performance goal is an order entry response time under three seconds. If the application does not meet that goal, identify the cause and take corrective action. During development, test the application to determine if it meets the desired performance goals.

User Expectations

Application developers, database administrators, and system administrators must be careful to set appropriate performance expectations for users. When the system carries out particularly complicated operations, response time may be slower than when it is performing a simple operation. Users should be made aware of which operations might take longer.

Performance Evaluation

With clearly defined performance goals, you can readily determine when performance tuning has been successful. Success depends on the functional objectives you have established with the user community, your ability to measure whether or not criteria are being met, and your ability to take corrective action to overcome any performance issues.

Ongoing performance monitoring enables you to maintain a well-tuned system. Keeping a history of the application's performance over time enables you to make useful comparisons. With data about actual resource consumption for a range of loads, you can conduct objective scalability studies and from these predict the resource requirements for anticipated load volumes.

Improving Performance

In order to improve the performance of your applications, you have to consider the various factors that influence performance and make changes to your system as necessary.

Factors in Improving Performance

Performance spans several areas:

- **Sizing and configuration:** Determining the type of hardware needed to support your performance goals
- **Parameter tuning:** Setting configurable parameters to achieve the best performance for your application

- **Performance monitoring:** Determining what hardware resources are being used by your application and what response time your users are experiencing
- **Troubleshooting:** Diagnosing why an application is using excessive hardware resources, or why the response time exceeds the desired limit

Countering the Effects of Excessive Demand

Excessive demand increases response time and reduces throughput. If the demand rate exceeds the achievable throughput, then you must determine through monitoring which resource is exhausted, and if possible increase that resource.

Making Adjustments to Relieve Performance Problems

Performance problems can be relieved by making adjustments to the following:

- **Unit consumption:** Reducing the resource consumption of each request can improve performance. This might be achieved by pooling and caching.
- **Functional demand:** Rescheduling or redistributing the work can improve performance in some cases.
- **Capacity:** Increasing or reallocating resources can improve performance in some cases.

Tuning usually involves a series of trade-offs. After you have determined the bottlenecks, you may have to modify performance in some other areas to achieve the desired results. For example, if I/O is a problem, you may need to purchase more memory or more disks. If a purchase is not possible, you may have to limit the **concurrency** of the system to achieve the desired performance. However, if you have clearly defined goals for performance, the decision on what to trade for higher performance is easier because you have identified the most important areas.

Overview of Caching Solutions

Caching is one of the key technologies that promises to alleviate the computational and economic burdens faced by today's overstrained **e-business** infrastructures. Nearly all applications benefit from having Web content cached on **hosts** between the consumers searching for content and the content source itself. When applied to Web applications, caching is essentially a technique for storing partial or complete Web pages, both static and dynamic, in memory closer to the browser to address the problem of slow access to Web sites.

A practical caching solution must do the following tasks:

- serve dynamic content, ensuring freshness
- handle thousands of concurrent users at high sustained rates of throughput
- provide fast response times
- support local and global deployments
- integrate with other caching techniques
- post-process cached content
- provide high gains with low-cost infrastructure

Caching solutions can be employed in different tiers. Each solution targets a specific tier and presents certain capabilities. It is important to note that response time is the cumulation of the times to access different tiers in your architecture. Most often a complete solution is a combination of one or more caching solutions. Caching solutions include browser caching, proxy caching, content delivery network services, and server accelerators.

The following sections describe Oracle Application Server Web Cache, the middle-tier server acceleration and load balancing component of Oracle Application Server.

Introduction to Server Accelerators

A server accelerator is a cache and compression engine that stands in for one or more specific Web servers, rather than working on behalf of a group of browser users. A server accelerator cache, or "reverse proxy" cache, intercepts all requests to the Web servers, caches a copy of the objects served, and then serves those objects when it next receives requests for them. As the server accelerator's cache becomes populated, it is able to serve more of the requested content itself, freeing up processing resources in the application server and the database for other tasks. Server accelerators also help cut costs, as they are implemented on inexpensive platforms and take some of the load off of more expensive back-end content generation systems.

Introduction to Oracle Application Server Web Cache

Oracle Application Server Web Cache is a powerful, state-of-the-art server acceleration and load balancing solution. OracleAS Web Cache offers intelligent caching, page assembly, and compression features that distinguish it from all other Web caching solutions on the market. Unlike legacy **proxy servers**, which cache

only static **objects**, OracleAS Web Cache accelerates the delivery of both static and dynamic Web content, improving response time for feature-rich pages.

OracleAS Web Cache also supports Edge Side Includes (ESI) for performing page assembly in edge servers. OracleAS Web Cache leverages this technology to enable partial-page caching and dynamic page assembly using both cacheable and non-cacheable page fragments. In this way, OracleAS Web Cache optimizes the delivery of rich, personalized content.

Deployed before a farm of application servers or globally at the network edge, OracleAS Web Cache provides load balancing, **failover**, clustering, and surge protection features for application servers.

Oracle Application Server Web Cache Features

Oracle Application Server Web Cache is a powerful solution for accelerating Web-based applications. The key features of Oracle Application Server Web Cache are divided into three categories:

- Compression and Caching
- Workload Management
- End-User Experience Management

See Also: *Oracle Application Server Web Cache Administrator's Guide* for detailed discussions of each of these features

Compression and Caching

Oracle Application Server Web Cache uses compression, caching, page assembly, and invalidation technologies to speed the delivery of dynamically-generated content and make more efficient use of low-cost hardware.

Automatic Content Compression

You can select to have OracleAS Web Cache compress both cacheable and non-cacheable documents. Because compressed documents are smaller in size, they are delivered to browsers faster with fewer round-trips, reducing overall latency. On average, OracleAS Web Cache is able to compress text files by a factor of 10. For example, 60 KB files are compressed down to 6 KB.

Full Page Static and Dynamic Content Caching

OracleAS Web Cache uses cacheability rules to store documents. There are rules for storing static content, and also rules for storing dynamically-generated content created using technologies such as JavaServer Pages (**JSP**). Supporting dynamic content caching allows OracleAS Web Cache to recognize multiple versions of documents with the same URL, cache session-aware pages, and cache pages that contain personalized information. There are also rules for pages that require personalized content assembly of dynamic Edge Side Includes (ESI) fragments.

Cache Invalidation and Expiration

OracleAS Web Cache supports invalidation as a way to ensure that its cache stays valid with respect to the content being served. Administrators and developers can invalidate cache content by either sending an invalidation message to the computer running OracleAS Web Cache or by assigning an expiration time limit to the cached documents.

Partial-Page Caching and Personalized Page Assembly

OracleAS Web Cache provides dynamic assembly of pages with both cacheable and non-cacheable page fragments. It does this by enabling pages to be broken down into fragments of differing cacheability profiles. With partial-page caching, more HTML content can be cached, then assembled and delivered by OracleAS Web Cache when requested.

- **Page assembly components:** The basic structure used to create dynamic content is a template page containing HTML fragments. The template consists of common elements, such as the "look and feel" elements of the page. The HTML fragments represent dynamic subsections of the page. The template page is associated with the URL that end users request. The template page uses Edge Side Includes (ESI) to tell OracleAS Web Cache to fetch and include the HTML fragments. Each individual fragment is a separate object with its own caching policy.
- **Edge Side Includes (ESI):** ESI is a markup language that enables partial page caching for HTML fragments. ESI can be used with HTML, XML, and any Web publishing technology. ESI is an open standard. For more information, see <http://www.esi.org>.
- **Edge Side Includes for Java (JESI):** JESI is a specification and custom JSP tag library that developers can use to automatically generate ESI code using JSP syntax. Even though JSP developers can always use ESI, JESI provides an even easier way for JSP developers to express the modularity of pages and the

cacheability of those modules, without requiring developers to learn a new syntax.

Workload Management

Oracle Application Server Web Cache includes workload management features like load balancing, failover, surge protection, and clustering to enhance application availability and ensure quality of service.

Web Server Load Balancing and Failover

Load balancing and failover allows Web sites to be built with a collection of **servers** for better **scalability** and **reliability**. OracleAS Web Cache sends requests to the application server with the most available load using its *load balancing* feature. When an application server becomes unavailable, OracleAS Web Cache automatically performs backend **failover**. OracleAS Web Cache distributes the load over the remaining application servers and polls the failed application server for its status until it is back online. When the failed server returns to operation, OracleAS Web Cache includes it in the load distribution.

Performance Assurance

To handle performance issues while maintaining cache consistency, OracleAS Web Cache uses built-in performance assurance heuristics that enable it to assign a queue order to documents. These heuristics determine which documents can be served stale and which documents must be refreshed immediately. The queue order of documents is based on the popularity of documents and the validity of documents assigned during invalidation. If the current load and capacity of the application server is not exceeded, the most popular and least valid documents are refreshed first.

Cache Clustering

A cache cluster is a loosely related set of Web cache instances working together to provide a single logical cache. You can configure multiple instances of OracleAS Web Cache to run as members of a cache cluster. This increases the availability and scalability of your caching solution.

End-User Experience Management

Oracle Application Server Web Cache also includes performance monitoring functionality that provides valuable insight into end-user service levels.

End-User Performance Monitoring

The Oracle Application Server 10g (9.0.4) version of Web Cache includes instrumentation for end-user performance monitoring. Administrators can configure OracleAS Web Cache to measure end-user response times for individual URLs, sets of URLs, or entire Web-based applications, regardless of whether the URLs are cached. For each instrumented request, the complete user experience is recorded. The raw measurements are collected in the OracleAS Web Cache access logs.

New for Oracle Application Server Web Cache 10g (9.0.4)

The following sections describe new features introduced in the 10g (9.0.4) release of Oracle Application Server Web Cache.

Support for Client-side SSL Certificates

OracleAS Web Cache now supports applications that require client-side SSL certificates for PKI-based authentication. For HTTPS requests that require client-side certificates, the client browser sends its certificate to OracleAS Web Cache during the SSL handshake. The cache forwards the request to Oracle HTTP Server (OHS) along with the client's certificate information inserted in special HTTP request headers. OHS recognizes the headers and is able to pass user credentials to Oracle Application Server Single Sign-On for authentication.

Flexible Multi-version Caching Rules

Administrators now have better control over the granularity of multi-version caching rules based on user-agent request headers, or browser types. Previously, you could either cache one version of a page for all browsers, or you could cache one version for each browser type and version. Now, you can customize the caching rules to define groups of browsers that will share a cached version of a page. For example, you could cache one page for all versions of the Internet Explorer, one for all versions of Netscape, and one for all other browsers.

Integration with Oracle Process Manager and Notification Server (OPMN)

In addition to managing OHS and OC4J processes, OPMN now manages the cache and administration server processes for OracleAS Web Cache. These include the start, stop, and auto-restart operations. However, standalone OracleAS Web Cache deployments will continue to use the OracleAS Web Cache Control and "watchdog" process management utilities.

Inline Invalidation and Search Key Invalidation

This release of OracleAS Web Cache introduces an inline invalidation mechanism as an additional way to manage content freshness. The inline invalidation model is implemented as part of the OracleAS Web Cache ESI support, and provides a useful way for origin servers to include invalidation messages along with transactional responses sent to Web Cache. The ability to send invalidation messages inline reduces the connection overhead associated with sending invalidations separately.

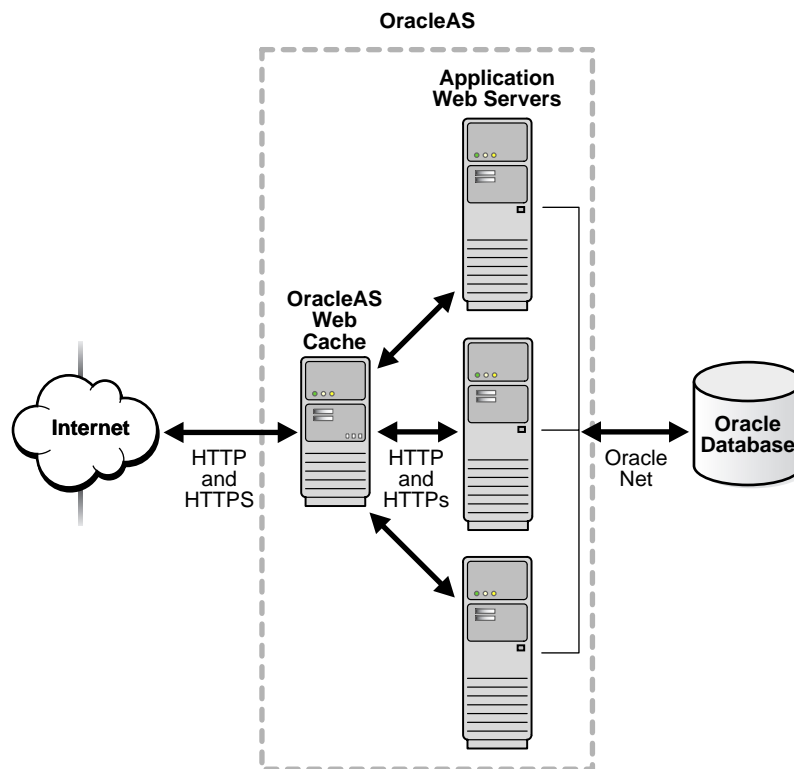
Another new invalidation feature for this release is support for search key invalidation. Previously, a cached document was identified by a URL-based cache key. Invalidation requests needed to specify either exact URLs or a set of URLs and headers matching a regular expression in order to invalidate cached objects. In this release, OracleAS Web Cache invalidation has been extended to support search keys. Cached objects can now be associated with multiple application-specified search keys, with the URL-based key being the primary key. Invalidation can be based on the search keys instead of the primary URL-based key, making invalidation easier for administrators and application developers to use.

Oracle Application Server Web Cache Deployment Architecture

In the simplest of deployment scenarios, OracleAS Web Cache is positioned in front of one or more Web servers to cache and compress content generated by those servers. OracleAS Web Cache then delivers that content to Web browsers. When Web browsers access the Web site, they send HTTP or HTTPS requests to OracleAS Web Cache, which acts as a virtual server for the Web site, masking the existence of the application server farm and the database. If the requested content has changed, OracleAS Web Cache retrieves the new content from the application servers according to the relative load on each server.

OracleAS Web Cache can be deployed on the same **host** as the origin application server (co-located) or on a separate **node** of its own (dedicated). Figure 9–1 shows a dedicated OracleAS Web Cache deployment.

Figure 9–1 Caching Architecture



Because OracleAS Web Cache consumes memory, co-location is only viable if the cache and the application servers do not contend for resources.

Dedicated deployment is often preferable to co-located deployment. In a dedicated scenario, there is no risk of resource contention with other server processes. OracleAS Web Cache also performs well on commodity hardware, so a dedicated deployment need not be a costly one in terms of hardware expenditure. For very high-volume Web sites, and to avoid a single point of failure, two or more hosts running OracleAS Web Cache may be deployed behind a third-party network load balancing device.

Cache hierarchies: OracleAS Web Cache offers hierarchical caching features that enable customers to easily create Content Delivery Networks (CDNs). Many Web-based applications mirror their Web sites in strategic geographical locations. Caching provides a low-cost alternative to mirroring, and can also be used to serve

local markets to shorten response times to these markets and to reduce bandwidth and rack space costs for the content provider. Additionally, in a distributed cache hierarchy, the central cache is aware of the local caches. As a result, any content invalidation messages sent to the central cache automatically propagate to these remote caches. This invalidation propagation ensures content consistency across the CDN and simplifies the management of cache hierarchies.

Using OracleAS Web Cache in heterogeneous environments: While integrated with Oracle Application Server, OracleAS Web Cache is also compatible with third-party application servers, databases, and content management systems.

See Also: *Oracle Application Server Web Cache Administrator's Guide*

Additional Caching Components

Web application developers may also encounter situations when application objects are not HTML or XML fragments; they may have to deal with XML DOM objects or Java serializable objects. There may also be requirements to reuse or post-process cached content, or maintain intermediate results. Oracle Application Server provides two components for dealing with application level caching:

- **Java Object Cache**
- **Web Object Cache**

Java Object Cache

Java Object Cache is a set of Java classes designed to manage Java objects within a process, across processes, and on local disks. Java Object Cache provides a powerful and flexible service that improves server performance by managing local copies of objects that are expensive to retrieve or create. There are no restrictions on the type of object that can be cached or the original source of the object. The management of each object in the cache is easily customized. Each object has a set of attributes associated with it to control such things as how the object is loaded into the cache, where the object is stored, how it is invalidated, and who should be notified when the object is invalidated. Objects can be invalidated as a group or individually.

Web Object Cache

Web Object Cache is a Web-application-level caching facility that is embedded and maintained within a Java Web application. The Web Object Cache is a hybrid cache, both Web-based and object-based. Using the Web Object Cache, applications can

cache programmatically using API calls (for servlets) or custom tag libraries (for JSPs). The Web Object Cache is generally used as a complement to the Web cache. By default, the Web Object Cache uses the Java Object Cache as its repository.

System Management

This chapter provides an overview of Oracle Application Server system management features and benefits. The topics include:

- Introduction to System Management
- Introduction to Oracle Enterprise Manager
- Oracle Enterprise Manager Application Server Control Features
- Oracle Enterprise Manager Application Server Control Architecture

Introduction to System Management

Oracle Application Server provides a set of industry standards-based management facilities to simplify all aspects of Web site administration. It does so by leveraging a single security and directory framework for all applications, and by providing administrators with a management tool to manage, monitor, tune, and troubleshoot across Oracle Application Server **instances**.

Oracle Application Server uses Oracle Enterprise Manager Application Server Control to enable Web site administrators to configure and administer Oracle Application Server instances, to monitor and optimize them for performance and **scalability**, and to help diagnose problems occurring with the application server.

Introduction to Oracle Enterprise Manager

Oracle Enterprise Manager Application Server Control (from here on Application Server Control) provides you with the management tools you need to monitor and administer Oracle Application Server instances. Application Server Control is installed with every instance of Oracle Application Server.

Application Server Control provides Web-based management tools designed specifically for Oracle Application Server. Using Application Server Control, you can monitor and configure components of your application server. You can deploy applications, manage security, and create and manage Oracle Application Server clusters.

Application Server Control consists of the following:

- **The Enterprise Manager Home pages you use to manage Oracle Application Server and its components:** These Web pages provide you with a high-level view of your Oracle Application Server environment. From these pages you can drill down for more detailed information on administration, configuration, and performance monitoring. These pages allow you to administer the application server and its components and deployed applications.
- **The underlying software technologies that keep track of your application server instances and components:** These technologies automatically perform many management tasks. For example, they discover the components of each application server instance, gather and process performance data, and provide access to application configuration information.

Application Server Control is installed as part of the Oracle Application Server installation process.

See Also: "Introduction to Administration Tools" in the *Oracle Application Server 10g Administrator's Guide*

Oracle Enterprise Manager Application Server Control Features

Oracle Enterprise Manager includes the following features that enable you to manage your Oracle Application Server framework:

- Complete Oracle Application Server Administration
- Monitoring Oracle Application Server

Complete Oracle Application Server Administration

Application Server Control provides a full set of features for performing Oracle Application Server administration, with Web-based interfaces for performing operations such as:

- Starting and stopping services
- Managing multiple application server instances on multiple hosts

- Modifying server configuration parameters
- Creating new Oracle Application Server Containers for J2EE (OC4J) instances and adding Java Virtual Machines (JVMs)
- Configuring J2EE resources such as Java Database Connectivity (JDBC) data sources and Java Authentication and Authorization Service (JAAS) providers for J2EE application security
- Deploying J2EE and Web Services Applications
- Managing additional application server components such as Oracle Application Server Business Intelligence
- Creating and managing clusters that speed up the configuration and deployment of your Web applications

Monitoring Oracle Application Server

After you have installed and configured Oracle Application Server, one of your primary tasks as an administrator is to monitor the application server for potential performance issues. The Application Server Control allows you to take a top-down approach to your monitoring activities. For example, you can start by reviewing the basic characteristics of your application server on the Application Server Home page, and then drill down to examine the performance of individual components.

Using this approach, the Application Server Control can help you identify high-level performance issues, such as a high CPU load on your system. You can then drill down to individual components or applications to isolate the root cause of the high-level problem.

See Also:

- *Oracle Application Server 10g Administrator's Guide*
- *Oracle Application Server 10g Performance Guide*

Oracle Enterprise Manager Application Server Control Architecture

Oracle Enterprise Manager Application Server Control provides immediate, out-of-the-box management value with each Oracle Application Server instance you install. Each Oracle Application Server installation includes an Application Server Control for managing that installation. Application Server Control is based on several underlying pieces that comprise the application server management stack,

including Distributed Configuration Management (DCM), Oracle Process Management and Notification Server (OPMN), and Dynamic Monitoring Service (DMS).

Oracle Enterprise Manager Application Server Control Underlying Technologies

Application Server Control relies on various technologies to discover, monitor, and administer the Oracle Application Server environment. These technologies include:

- **Distributed Configuration Management (DCM):** DCM manages configurations among application server instances that are associated with a common Metadata Repository. It enables Oracle Application Server cluster-wide deployment so you can deploy an application to one instance and have it automatically propagated to the entire cluster. You can also make a single host or instance configuration change to one instance and have it propagated across all instances in the cluster. Application Server Control uses DCM to make configuration changes and to propagate configuration changes and deployed applications across the cluster.
- **Oracle Process Manager and Notification Server (OPMN):** OPMN provides process control and monitoring for application server instances and their components. It gathers component status information, and distributes the status information to components that are interested in it. The Application Server Control uses OPMN for such tasks as starting and stopping the components of your application server instance.
- **Oracle Management Agent:** The Oracle Management Agent is a component of Oracle Enterprise Manager that gathers monitoring data and performs administration tasks for the Application Server Control.
- **Oracle Dynamic Monitoring Service (DMS):** The Management Agent leverages another underlying service, the Dynamic Monitoring Service (DMS), to collect performance data. Oracle Application Server components are instrumented with DMS to provide a comprehensive set of built-in performance metrics to automatically measure runtime performance statistics. As a result, the Application Server Control uses this data to monitor the duration of important phases of request processing, as well as status information, such as the number of requests being handled at any given time.

See Also:

- *Distributed Configuration Management Reference Guide*
- *Oracle Process Manager and Notification Server Administrator's Guide*

Security and Identity Management

This chapter provides an overview of Oracle Application Server security solutions. The topics include:

- Introduction to Security
- Security Components and Features
- Security Architecture

Introduction to Security

Oracle Application Server provides a comprehensive integrated security framework supporting all of its components, as well as third party and custom applications deployed on Oracle Application Server. The framework is based on Oracle Identity Management for user **authentication** and **authorization** and user provisioning.

In addition to the components involved in Identity Management, other Oracle Application Server components are also involved in providing security for your online applications. The main components involved are Oracle Application Server Web Cache, Oracle HTTP Server, Oracle Application Server Portal, and Oracle Application Server Java Authentication and Authorization Service.

See Also: *Oracle Application Server 10g Security Guide*

Introduction to Identity Management

Identity management is the process by which the complete security life cycle for network entities is managed for an organization.

Identity management most commonly refers to the management of an organization's application users, where steps in the security life cycle include account creation, suspension, privilege modification, and account deletion. The

network entities managed may also include devices, processes, applications, or anything else that needs to interact in a networked environment. Entities managed by an identity management process may also include users outside of the organization, such as customers, trading partners, or Web services.

Identity management is important to IT deployments because it can reduce administrative costs while at the same time improving security. Identity management benefits include:

- Identity management saves money. For most enterprises, application user administration is a very expensive, laborious, and error-prone process. Identity management centralizes and automates many user administration tasks, reducing costs while improving accuracy and security.
- Identity management enables faster deployments. Typically, provisioning of a new application means creating and managing separate user accounts and their privileges. Identity management enables the new applications to leverage the existing infrastructure for its user management, and thus reduces the time it takes to deploy and manage new applications.
- Identity management improves the end-user experience. An identity management strategy allows new users to gain access to their applications quickly, eliminating wasted employee time. Further, identity management enables a customized application experience, enhancing usability.
- Identity management improves application security. An identity management strategy allows users to have their passwords and security credentials managed centrally. This reduces the temptation for users to write down security information, raising the risk of unauthorized access.

Security Components and Features

The Oracle Application Server security framework involves many components, each of which contributes features that enable you to secure your Oracle Application Server deployment:

- Oracle Identity Management
 - Oracle Application Server Single Sign-On
 - Oracle Internet Directory
 - includes Oracle Delegated Administration Services and Oracle Directory Integration and Provisioning
 - Oracle Application Server Certificate Authority

- Oracle HTTP Server Security
- OracleAS Web Cache Security
- Portal Security

Oracle Identity Management

Oracle Identity Management is an integrated infrastructure that Oracle products rely on for distributed security. The Oracle Identity Management infrastructure includes the following components:

- Oracle Internet Directory, a scalable, robust LDAP V3-compliant directory service implemented on the Oracle9i Database.
- Oracle Directory Integration and Provisioning, part of Oracle Internet Directory, which permits synchronization between Oracle Internet Directory and other directories, user repositories, automatic provisioning services, and applications.
- Oracle Delegated Administration Services, part of Oracle Internet Directory, which provides trusted proxy-based administration of directory information by users and application administrators.
- Oracle Application Server Single Sign-On, which provides single sign-on access to Oracle and third-party Web applications.
- Oracle Application Server Certificate Authority, which generates and publishes X.509v3 certificates to support PKI-based (strong) authentication methods.

While Oracle Identity Management is designed to provide an enterprise infrastructure for Oracle products, it may also serve as a general-purpose identity management solution for user-written and third-party applications. Oracle Identity Management provides a robust and scalable enterprise-wide identity management platform for third-party applications, hardware, and network operating systems. Custom applications may leverage Oracle Identity Management through a set of documented and supported services and APIs, for example:

- Oracle Internet Directory provides LDAP APIs for C, Java, and PL/SQL, and is compatible with other LDAP SDKs.
- Oracle Delegated Administration Services provide a Web-based self-service console that may be customized to support third-party applications. In addition, it provides a number of services for building customized administration interfaces that manipulate directory data.

- Oracle Directory Integration Services facilitate the development and deployment of custom solutions for synchronizing OID with third-party directories and other user repositories.
- Oracle Provisioning Integration Services provide a mechanism for provisioning third-party applications, as well as a means of integrating the Oracle environment with other provisioning systems.
- Oracle Application Server Single Sign-On provides APIs for developing and deploying partner applications that share a single sign-on session with other Oracle Web applications.
- Oracle Application Server Java Authentication and Authorization Service, the Oracle implementation of the JAAS standard, allows applications developed for the Web using the Oracle J2EE environment to leverage the identity management infrastructure for authentication and authorization.

In addition, Oracle works with third-party application vendors to ensure their applications can leverage Oracle Identity Management out of the box.

Oracle Application Server Single Sign-On

An important security feature of Oracle Application Server is support of single sign-on (SSO) to Web-based applications. Oracle Application Server Single Sign-On addresses the problem of "too many passwords." With the rapid growth of the Internet, this problem has become increasingly prevalent, causing users inconvenience that typically results in poor security practices and increased administrative costs.

Oracle Application Server Single Sign-On resolves this problem by enabling users to log in to Oracle Application Server and gain access to those applications for which they are authorized, without requiring them to re-enter a user name and password for each application.

It is fully integrated with Oracle Internet Directory, which stores user information. It supports LDAP-based user and password management through OID.

Oracle Application Server Single Sign-On provides the following functionality:

- It authenticates users and passes their identities securely to partner applications, such as **Oracle Application Server Portal**. It prompts users for a username and password when they access the system for the first time in a given time period, and verifies the password presented by the user.
- It uses **cookies**, which are formatted pieces of information stored on a **browser client** by a **Web server**. Cookies allow Web servers to store and retrieve

information about the client user, effectively maintaining client state information in the otherwise stateless Web environment.

- It supports Public Key Interface (PKI) client authentication, which enables PKI authentication to a wide range of Web applications. By means of an API, Oracle Application Server Single Sign-On can integrate with third-party **authentication** mechanisms such as Netegrity SiteMinder.

With Oracle Application Server Single Sign-On, users typically sign on to a centrally administered Single Sign-On Server through a designated Web **portal**. Once it authenticates the user, Single Sign-On Server displays links to all the applications for that user.

Using a centrally administered Single Sign-On Server has these advantages:

- **Convenience:** The user enters the user name and password only once, at a central corporate Web portal, to access all the needed applications. From the user's perspective, authentication to each application happens transparently.
- **Increased Security:** Fewer user name and password combinations lower the risk of unauthorized access to a user's restricted information.
- **Ease of Administration:** Oracle Application Server Single Sign-On provides centralized provisioning of user accounts, so that administrators can easily create new user accounts. Centralizing the authentication process also makes it possible to support additional authentication mechanisms in a localized manner. For example, you can implement password-based authentication, using Single Sign-On and Oracle Internet Directory, then switch to digital certificate-based authentication using OracleAS Certificate Authority, Single Sign-On, and Oracle Internet Directory, and the change would be localized to the SSO Server.

Partner and External Applications There are two kinds of applications to which Oracle Application Server Single Sign-On provides access:

- Partner Applications
- External Applications

Partner applications are integrated with the SSO Server. They are built upon an Oracle Application Server Single Sign-On API that enables them to delegate authentication to the SSO Server.

External applications are Web-based applications that retain their authentication logic. They do not delegate authentication to the SSO Server and, as such, require a user name and password to provide access. Currently, these applications are limited

to those which employ an **HTML** form for accepting the user name and password. The user name may be different from the SSO user name, and the SSO Server provides the necessary mapping.

The SSO offering in Oracle Application Server is a critical differentiator for users seeking a robust, fully integrated SSO architecture. Oracle Application Server leverages **JAAS**, as well as Oracle Internet Directory, to deliver a comprehensive end-to-end security infrastructure across the entire Oracle Application Server product.

Oracle Internet Directory

Oracle Internet Directory (OID) is a critical component of Oracle Application Server management and security infrastructure. It ensures that user accounts and groups are managed centrally through the LDAP Version 3 standard. Oracle Application Server enables users to be created centrally in OID and shared across all components in Oracle Application Server. When users log in, they are authenticated once by Oracle Application Server Single Sign-On against their OID credentials, and can thereby access multiple applications seamlessly.

Oracle Delegated Administration Services Oracle Internet Directory includes Oracle Delegated Administration Services, which provides trusted proxy-based administration of directory information by users and application administrators. Oracle Delegated Administration Services includes a Self-Service Console, an easy-to-use, Web-based interface which allows end-users and application administrators to search for and manage data in the directory. Through this console, Oracle Delegated Administration Services provides Oracle Application Server with a means of provisioning end-users in the Oracle Application Server environment. OID also enables components of Oracle Application Server to broadcast data about users and group events, so that those components can update any user information stored in their local application instances.

Oracle Directory Integration and Provisioning Oracle Directory Integration and Provisioning enables customers to synchronize data between various directories and Oracle Internet Directory. The Oracle Directory Integration Platform is a set of services and interfaces that make it possible to develop synchronization solutions with third party metadirectories and other enterprise repositories, such as SunONE/iPlanet Directory Server. With Oracle Application Server, Oracle Internet Directory includes connectors for out-of-the-box synchronization with Oracle Human Resources, Active Directory, and SunONE/iPlanet Directory Server 4.2 and 5.0.

Oracle Internet Directory also provides a plug-in framework for applications that require customized functionality, such as referential integrity of data. The plug-in framework is delivered as a highly-flexible **PL/SQL** interface, allowing user-defined operations to be invoked by the **directory server** before, after, or in place of standard LDAP commands.

User and Directory Searches Oracle Internet Directory provides users with directory searches capabilities with sophisticated server-side **caching** capabilities. OID also provides two key features that ensure administrators can deliver seamless directory services to all users:

- **Alias De-reference:** When a user or an application searches on an alias, OID automatically de-references the alias and returns the entry to which it refers. This feature enables administrators to change the names of **objects** in ways that are transparent to users and applications.
- **Enhanced Proxy Capabilities:** Administrators can safely establish performant, auditable middle-tier application access to the directory on behalf of end user communities.

Key Directory Features Oracle Internet Directory provides the following key directory features:

- Native LDAP server complying with all LDAP v2 and v3 **RFCS**, along with the Open Group's "LDAP CERTIFIED" test specifications (VSLDAP test suite 2.2)
- Supports the **X.500** information, naming, and storage model
- Extensible directory schema for online modifications with no downtime
- LDAP APIs in Java, C, and PL/SQL to assist with application development

Using Oracle Internet Directory with Middle Tier Components The middle tier components use Oracle Internet Directory in the following ways:

- Application server instances and infrastructures store security and management information in Oracle Internet Directory. Oracle Internet Directory stores users' information, such as user names and privileges, required for internal operation of the application server.
- Oracle Application Server Java Authentication and Authorization Service stores **realm** and **JAAS policy** in Oracle Internet Directory.
- Oracle Application Server Single Sign-On validates the user name and password against user and group profiles stored in Oracle Internet Directory.

See Also: *Oracle Internet Directory Administrator's Guide*

Oracle Application Server Certificate Authority

Oracle Application Server Certificate Authority (OCA) is a component of the Oracle public key infrastructure (PKI) offering that allows you to create and manage X.509v3 digital certificates for use in Oracle or third-party software. The Certificate Authority is fully standards-compliant, and is fully integrated with Oracle Application Server Single Sign-On and Oracle Internet Directory. Oracle Application Server Certificate Authority provides Web-based certificate management and administration, as well as XML-based configuration. It leverages the identity management infrastructure, high availability, and scalability of the Oracle platform.

Java Authentication and Authorization Service (JAAS)

Oracle Application Server provides an implementation of Java Authentication and Authorization Service (JAAS) that integrates with the Oracle Application Server J2EE security infrastructure to enforce security constraints for Web (**servlets** and **JSPs**) and **EJB** components.

Oracle Application Server Java Authentication and Authorization Service support provides the following benefits:

- Integrates **Java**-based applications with Oracle Application Server Single Sign-On, includes authentication, thereby giving you extensible security for Java-based applications
- Manages access control policies centrally in Oracle Internet Directory, controls access by role, and partitions security **policy** by subscriber
- Supports impersonation of a specific user, allows an enterprise bean, servlet, or JSP to run with the permissions associated with the current client or a specified user

OracleAS Web Cache Security

Oracle Application Server Web Cache provides the following security-related features:

- Restricted Administration
- Secure Sockets Layer (SSL) Support

Restricted Administration

OracleAS Web Cache restricts administration with the following features:

- Password **authentication** for administration and invalidation operations
- Control over ports from which administration and invalidation operations can be requested
- IP and subnet administration **restrictions**

Secure Sockets Layer (SSL) Support

The secure sockets layer (SSL) protocol, developed by Netscape Corporation, is an industry standard for network transport layer security. SSL provides authentication, **encryption**, and data integrity in a public-key infrastructure (PKI). By supporting SSL, OracleAS Web Cache is able to cache pages from HTTPS requests.

Note that HTTPS traffic can be process intensive. If OracleAS Web Cache needs to have traffic travel over the open Internet, then configure OracleAS Web Cache to send HTTPS requests to the application server. If traffic only travels through a LAN in a data center, then the traffic can be sent with **HTTP** so as to reduce the load on the application servers.

SSL interacts with the following entities:

- **Certificate Authority:** A certificate authority (CA) is a trusted third party that certifies the identity of third parties and other entities, such as users, databases, administrators, **clients**, and **servers**. The CA verifies the party identity and grants a certificate, signing it with its **private key**. The OracleAS Web Cache certificate must be signed by a CA. In Oracle Application Server 10g (9.0.4) Oracle provides the new Oracle Application Server Certificate Authority (OCA), which allows you to create and manage X.509v3 digital certificates for use in Oracle or third-party software. For more information on OCA see "Oracle Application Server Certificate Authority" in this chapter.
- **Certificate:** A **certificate** is created when a party's **public key** is signed by a trusted CA. A certificate ensures that a party's identification information is correct, and that the public key actually belongs to that party. A certificate contains the party's name, public key, and an expiration date, as well as a serial number and certificate chain information. It can also contain information about the privileges associated with the certificate. When a network entity receives a certificate, it verifies that it is a trusted certificate, one issued and signed by a trusted CA.

- **Wallet:** A **wallet** is a transparent database used to manage authentication data such as keys, certificates, and trusted certificates needed by SSL. A wallet has an X.509 version 3 certificate, private key, and list of trusted certificates.

Security administrators use the **Oracle Wallet Manager** to manage security credentials on the OracleAS Web Cache server. Wallet owners use it to manage security credentials on clients. Specifically, Oracle Wallet Manager is used to do the following:

- Generate a public-private **key pair** and create a certificate request for submission to a CA
- Install a certificate for the identity
- Configure trusted certificates for the identity

To support HTTPS between browsers and OracleAS Web Cache, configure a wallet on the OracleAS Web Cache server for each supported site. To support HTTPS between OracleAS Web Cache and the application servers, configure a wallet on the application server.

See Also: *Oracle Application Server 10g Security Guide*

Oracle HTTP Server Security

Oracle HTTP Server provides the following security-related features:

- Session Renegotiation Support
- SSL Hardware Acceleration Support
- Port Tunnelling
- OHS to OC4J SSL Support

Session Renegotiation Support

With session renegotiation support, individual directories can be protected by different levels of encryption. Some directories may only need a minimum level of encryption, while others require stronger encryption.

SSL Hardware Acceleration Support

Software-based SSL encryption can sometimes be slow. Oracle HTTP Server supports the option of having dedicated SSL hardware through nCipher. nCipher is a third-party math accelerator that improves the performance of the PKI cryptography that SSL uses.

Port Tunnelling

Port tunneling lets all communication between Oracle HTTP Server and OC4J happen on a single port or a small number of ports. Previously, the firewall configuration had to include port information for several ports to handle communication between Oracle HTTP Server and multiple OC4J instances. With the port tunnel, a daemon routes requests to the appropriate OC4J instances. Using this method, only one port has to be opened through the firewall regardless of the number of OC4J instances involved.

OHS to OC4J SSL Support

Oracle HTTP Server and OC4J can communicate using the AJP protocol over the Secure Sockets Layer (SSL). Previously, OHS and OC4J used the AJP 13 protocol unencrypted, without support for authentication. Now, OHS has been modified to extend support to the AJP13 protocol over SSL.

Portal Security

This section discusses the following security topics:

- User Authentication in OracleAS Portal
- Access Control in OracleAS Portal

User Authentication in OracleAS Portal

Oracle Application Server Single Sign-On provides a single point of validation for **portal** user credentials and governs user access to intranet resources based on employee profiles. When a user logs into a portal page, the single sign-on **server** validates the user name and password against user and group profiles stored in Oracle Internet Directory.

Oracle Application Server Single Sign-On also supports external, partner applications. For these applications, the single sign-on server logs in to the application automatically for the user.

See Also: Chapter 10, "System Management"

Access Control in OracleAS Portal

Most portal elements have an access control list (ACL). This list controls which users and groups may access the element, and to what extent. For example, if you wanted all of the users in a group to be able to see the items on a portal page, in the page's ACL you would grant that group View privileges.

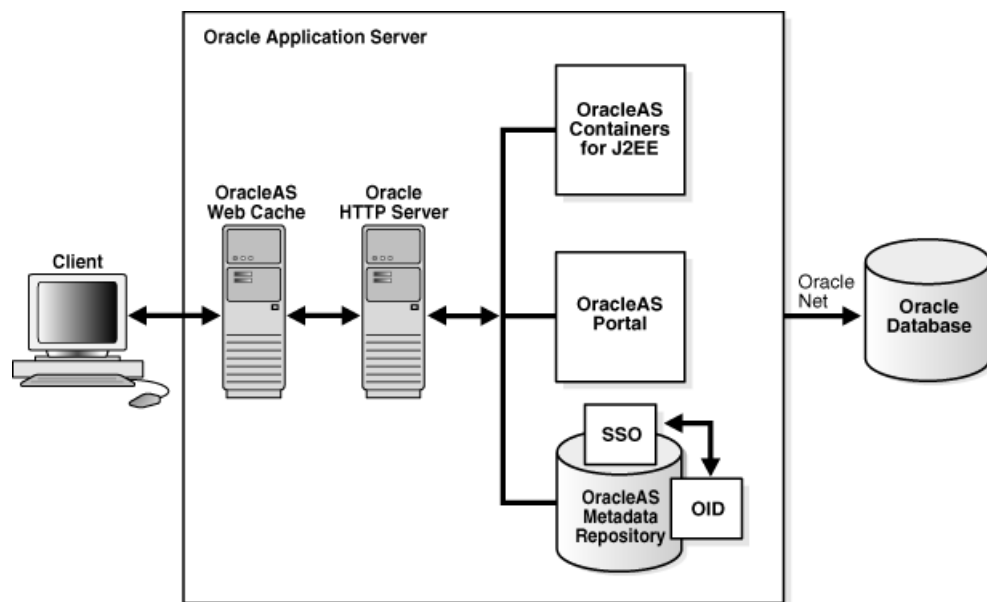
Besides ACLs, you can also use global privileges to grant access to all **objects** of a given type in **Oracle Application Server Portal**. For example, granting the Create privilege for All Pages to a group enables all members of that group to create pages.

Security Architecture

Figure 11-1, illustrates how the elements of Oracle Application Server function together. Following is a list of the functionality of the various components:

- **OracleAS Web Cache** is positioned in front, where it provides efficiency.
- **Oracle HTTP Server** is the front-end Web server for Oracle Application Server. Through **Apache**-based modules as well as modules developed by Oracle, users can access a variety of Oracle Application Server services.
- **OracleAS Portal** provides the infrastructure, including the ability to create and manage Web pages. It lets you display multiple Web pages on each Web page, with links to content through **Java** applications.
- **Java** engine lies underneath **Oracle Application Server Web Cache** and Oracle HTTP Server, supporting their ability to link efficiently.
- **OracleAS Single Sign-On** enables users to log in to Oracle Application Server and gain access to those applications for which they are authorized, without requiring them to re-enter a user name and password for each application.
- **Oracle Internet Directory** serves the rest of the stack by providing authentication and a centralized user model whereby users can be created and managed on an enterprise scale.

Figure 11-1 Oracle Application Server Security Architecture



Recommended Topologies

This chapter provides an overview of Oracle Application Server recommended topologies. The topics include:

- Introduction to Oracle Application Server Recommended Topologies
- Recommended General Development Topologies
- Recommended General Deployment Topologies
- Special Topologies

Introduction to Oracle Application Server Recommended Topologies

Oracle Application Server, a single integrated platform for any application deployment, offers a variety of topology options. This chapter identifies the most commonly used topologies and discusses some of the major considerations involved with deploying such topologies. The Oracle Application Server recommended topologies fall into three categories:

- General Development Topologies
- General Deployment Topologies
- Special Topologies

Note: Each real-world deployment will have unique characteristics that may vary the optimal topology for that site. The topologies described here are intended to provide high availability, manageability, security, and performance requirements for a broad range of users.

General Development Topologies

General Development Topologies consist of configuration topologies used by the following types of developers:

- Java developers
- Portal and Wireless developers
- Forms and Reports developers
- Integration architects and process modelers

General Deployment Topologies

General deployment topologies are determined based on the following types of usage:

- Enterprise Data Center topologies
- Departmental topology
- Development Life Cycle Support topology

Special Topologies

Oracle Application Server has some special topologies that are consumed as part of the general deployments identified in the previous sections. In order to meet your specific needs, a recommended deployment topology that includes some of the following topologies may be necessary:

- High Availability (HA) topologies

Key Considerations

The following perspectives are presented to best evaluate and recommend appropriate deployment options to satisfy your enterprise's individual needs. Detailed information on each of these perspectives is available in additional documents, as listed.

Installation

Installation considerations include which install model is best suited for the topology, what types of machines are typically used, and what type of user will be performing the installation. See the *Oracle Application Server 10g Installation Guide* for your platform for detailed steps to achieve each of these configurations.

Application Deployment and Performance

Deployment and performance considerations include performance goals, component distribution across hardware, application development strategies, and parameter tuning. See the *Oracle Application Server 10g Performance Guide* for your platform for more information.

Security

Security considerations include what level of security is required, whether components or applications should be isolated to individual hardware nodes, and what third-party security providers will be involved. See the *Oracle Application Server 10g Security Guide* for more information.

Management

Management considerations include what facilities are required for management, component and application distribution across hardware nodes, and what sorts of backup and recovery methods will be used. See the *Oracle Application Server 10g Administrator's Guide* for more information.

High Availability Deployment

High Availability (HA) considerations include which HA methodology will be used. See the *Oracle Application Server 10g High Availability Guide* for more information.

Third Party Products

Third Party Products considerations include which special third party products will be used, and how they will impact the high availability, security, and performance of the topology.

Recommended General Development Topologies

This section describes the following recommended general development topologies:

- Java Developer Topology: Low-end Machines with Java IDE Tools
- Portal and Wireless Developer Topology: Medium-sized Machines
- Forms, Reports, and Discoverer Developer Topology: Medium-sized Machines
- Integration Architect and Process Modeler Topology: Medium-sized Machines

Java Developer Topology: Low-end Machines with Java IDE Tools

The development requirements for Java developers are mainly related to ease of development and deployment. Table 12–1 outlines the key considerations for Java developers.

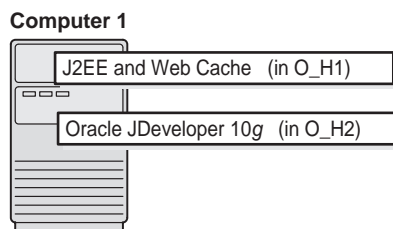
Table 12–1 Key Considerations for Java Developers

Consideration	Developer Assumptions
Install	<ul style="list-style-type: none"> ■ O/S patch releases applied quickly ■ Single host installation ■ Latest versions downloaded from OTN, making download size very important ■ Infrastructure Services not heavily used ■ Limited knowledge of Oracle database ■ IDEs such as JDeveloper used for development
Management	<ul style="list-style-type: none"> ■ Oracle Enterprise Manager Application Server Control installed automatically with each installation ■ Use Application Server Control for: <ul style="list-style-type: none"> Starting and stopping components as needed Enabling/disabling components to minimize system resource consumption Setting or changing configuration parameters for the J2EE container, Oracle HTTP Server, or OracleAS Web Cache Deploying and configuring applications Managing J2EE application security Monitoring application and component performance and resource consumption in real time Viewing and setting ports Browsing and searching log files Managing application server and J2EE clusters ■ Command line utilities are available for scripting and automation or if you are using standalone components ■ File backup and restore mechanisms are typically used ■ If using Infrastructure Services, use OracleDBA Studio for management
Security	<ul style="list-style-type: none"> ■ Some interest in file-based JAAS-XML

Table 12–1 Key Considerations for Java Developers

Consideration	Developer Assumptions
Application Deployment and Performance	<ul style="list-style-type: none"> ■ Development occurs on a single node ■ Important that components not being utilized do not impact development compiles, deployment, and start/stop times on limited hardware resources ■ J2EE applications deployed on standalone J2EE and Web Cache installs ■ J2EE applications deployed on Oracle Application Server clusters with or without Web Cache ■ Actual cluster testing likely to happen on a separate test system, not in the development environment ■ Web Cache testing desirable in the development cycle on the development machine ■ Load balancing is important when testing the deployment of applications
High Availability	<ul style="list-style-type: none"> ■ Not concerned with Infrastructure HA solutions
Third Party Products	<ul style="list-style-type: none"> ■ None

Figure 12–1 illustrates the recommended topology for the Java developer environment. Note that "O_H" in the figure denotes an Oracle home directory.

Figure 12–1 Java Developer Topology

Portal and Wireless Developer Topology: Medium-sized Machines

Both Oracle Application Server Portal and Oracle Application Server Wireless developers have similar application deployment requirements. They both need Infrastructure to deploy and test their applications. Table 12–2 outlines the key considerations of Portal and Wireless developers.

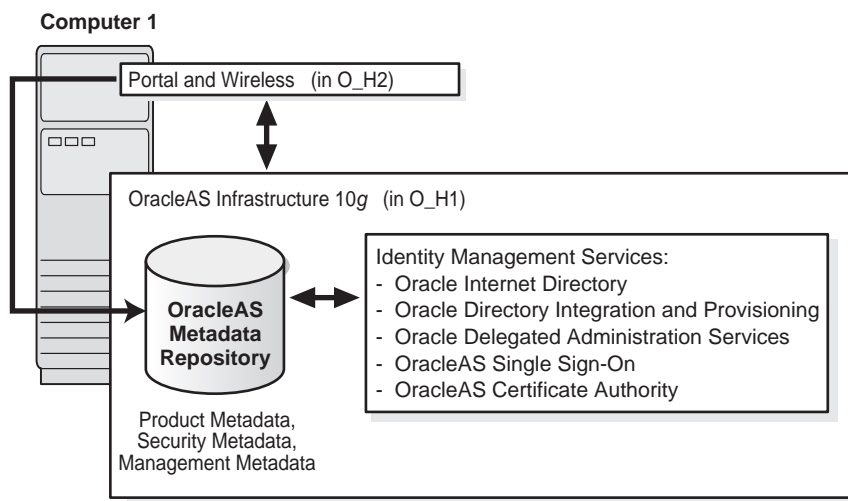
Table 12–2 Key Considerations for Portal and Wireless Developers

Considerations	Developer Assumptions
Install	<ul style="list-style-type: none"> ■ Single host installations ■ Portal and Wireless middle tier install type used ■ Infrastructure install type also required ■ Fair knowledge of Oracle database
Management	<ul style="list-style-type: none"> ■ Oracle Enterprise Manager Application Server Control installed automatically with each installation ■ Use Application Server Control for: <ul style="list-style-type: none"> Starting and stopping components as needed Enabling/disabling components to minimize system resource consumption Setting or changing configuration parameters for any of the application server components Deploying and configuring applications Managing application security Monitoring application and component performance and resource consumption in real time Viewing and setting ports Browsing and searching log files Managing infrastructure schemas ■ Use Oracle DBA Studio for managing the infrastructure database ■ Command line utilities are available for scripting and automation ■ File backup and restore mechanisms are typically used
Security	<ul style="list-style-type: none"> ■ Oracle Application Server Single Sign-On (SSO) ■ Oracle Internet Directory (OID)
Application Deployment and Performance	<ul style="list-style-type: none"> ■ Wireless applications can be built and tested with only Wireless installed and configured ■ Portlets can be built and tested if both Portal and Wireless are installed and configured ■ JPDK and the Wireless SDK that ship with Oracle Application Server used for development/deployment
High Availability	<ul style="list-style-type: none"> ■ Not concerned with Infrastructure HA solutions

Table 12–2 Key Considerations for Portal and Wireless Developers

Considerations	Developer Assumptions
Third Party Products	<ul style="list-style-type: none"> Wireless adapters, wireless network gateways (for example, WAP gateway, voice gateway, SMS gateway)

Figure 12–2 illustrates the recommended topology for the Portal and Wireless developer environment.

Figure 12–2 Portal and Wireless Developer Topology

Forms, Reports, and Discoverer Developer Topology: Medium-sized Machines

Oracle Application Server Forms, Oracle Application Server Reports, and Oracle Application Server Discoverer developers have similar application deployment requirements. They need Infrastructure and Oracle Developer Suite to deploy and test their applications. Table 12–3 outlines the key considerations of Forms, Reports, and Discoverer developers.

Table 12–3 Key Considerations for Forms and Reports Developers

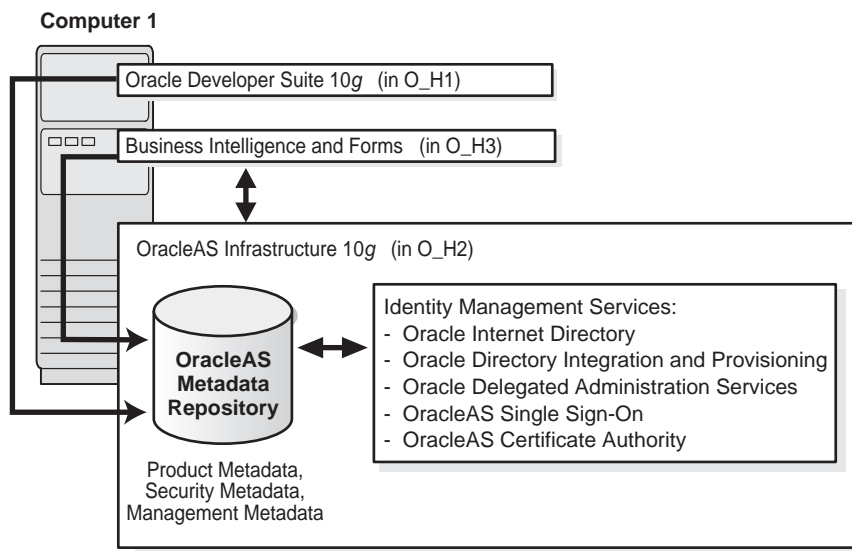
Considerations	Developer Assumptions
Install	<ul style="list-style-type: none"> ■ Single host installations ■ Business Intelligence and Forms middle tier install type ■ Fair knowledge of Oracle database
Management	<ul style="list-style-type: none"> ■ Oracle Enterprise Manager Application Server Control installed automatically with each installation ■ Use Application Server Control for: <ul style="list-style-type: none"> Starting and stopping components as needed Enabling/disabling components to minimize system resource consumption Setting or changing configuration parameters for any of the application server components Deploying and configuring applications Managing application security Monitoring application and component performance and resource consumption in real time Viewing and setting ports Browsing and searching log files Managing infrastructure schemas ■ Use Oracle DBA Studio for managing the infrastructure database ■ Command line utilities are available for scripting and automation ■ File backup and restore mechanisms are typically used
Security	<ul style="list-style-type: none"> ■ Existing Forms developers do not use Single Sign-On (SSO) ■ SSO for some Forms applications ■ SSO for some Reports applications
Application Deployment and Performance	<ul style="list-style-type: none"> ■ Forms Builder and Reports Builder are part of Oracle Developer Suite. During the development of these applications, the developers can test the applications on the Oracle Application Server instance that comes with Oracle Developer Suite ■ For real deployments, developers need Forms Services and Reports Services

Table 12–3 Key Considerations for Forms and Reports Developers

Considerations	Developer Assumptions
High Availability	<ul style="list-style-type: none"> Not concerned with Infrastructure HA solutions
Third Party Products	<ul style="list-style-type: none"> Reports uses Data Direct Adapters and Oracle gateways to access non-Oracle data sources

Figure 12–3 illustrates the recommended topology for the Forms and Reports developer environment.

Figure 12–3 Forms, Reports, and Discoverer Topology



Integration Architect and Process Modeler Topology: Medium-sized Machines

Integration architects and process modelers need the Oracle Application Server ProcessConnect middle tier install. They need appropriate adapters and also Infrastructure to deploy and test their integration applications. Table 12–4 outlines the key considerations of integration architects and process modelers.

Table 12–4 Key Considerations for Integration Architects and Process Modelers

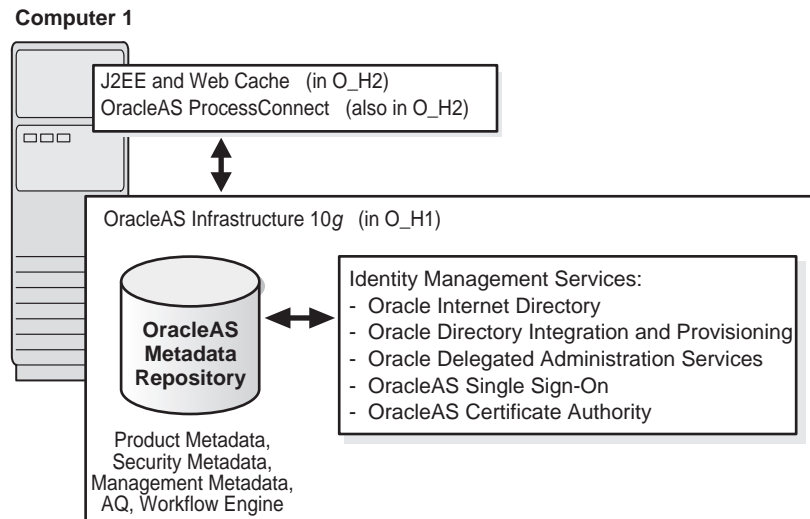
Considerations	Architect/Modeler Assumptions
Install	<ul style="list-style-type: none"> ■ Single host installations ■ J2EE and Web Cache middle tier install type and Oracle Application Server ProcessConnect Install ■ Infrastructure used for Product Metadata Services (Hub Database) ■ Fair knowledge of Oracle database ■ Good knowledge of applications for which they are building the integration
Management	<ul style="list-style-type: none"> ■ Oracle Enterprise Manager Application Server Control installed automatically with each installation ■ Use Application Server Control for: <ul style="list-style-type: none"> Starting and stopping components as needed Enabling/disabling components to minimize system resource consumption Setting or changing configuration parameters for any of the application server components Deploying and configuring applications Managing application security Monitoring application and component performance and resource consumption in real time Viewing and setting ports Browsing and searching log files Managing infrastructure schemas ■ Use Oracle DBA Studio for managing the infrastructure database ■ Command line utilities are available for scripting and automation ■ File backup and restore mechanisms are typically used
Security	<ul style="list-style-type: none"> ■ Oracle Application Server Single Sign-On (SSO) ■ Oracle Internet Directory (OID)

Table 12–4 Key Considerations for Integration Architects and Process Modelers

Considerations	Architect/Modeler Assumptions
Application Deployment and Performance	<ul style="list-style-type: none"> ■ Deployment tool to deploy the model in the Oracle Application Server integration repository that resides in the infrastructure ■ Oracle Enterprise Manager to modify some performance-related properties. Use Enterprise Manager to monitor throughput and latency for Oracle Application Server integration components
High Availability	<ul style="list-style-type: none"> ■ Not concerned with Infrastructure HA solutions
Third Party Products	<ul style="list-style-type: none"> ■ Application adapters

Figure 12–4 illustrates the recommended topology for the integration architect and process modeler environment.

Figure 12–4 Integration Architect and Process Modeler Topology



Recommended General Deployment Topologies

This section describes the following recommended general deployment topologies:

- Enterprise Data Center Topology: Multiple Departments Sharing the Same Data Center
- Departmental Topology: Departments Hosting Their Applications
- Development Life Cycle Support Topology: Test to Stage to Production Support

Enterprise Data Center Topology: Multiple Departments Sharing the Same Data Center

There are two Enterprise Data Center topologies, one best suited for Java applications, and one for other applications such as Portal, Wireless, Forms, Reports, and Discoverer. Table 12–5 outlines the key considerations for using the Enterprise Data Center topologies.

Table 12–5 Key Considerations for the Enterprise Data Center Topology

Considerations	Deployment Assumptions
Install	<ul style="list-style-type: none"> ■ Multiple host installations on hardware clusters, NFS machines ■ Multiple middle tier instances ■ Dedicated Product Metadata Services for Portal applications ■ Shared Product Metadata Services for some applications ■ Shared Security Services throughout the enterprise ■ Support for test to stage to production cycles ■ Oracle Application Server should withstand hard disk replacement, CPU change, or RAM upgrades

Table 12–5 Key Considerations for the Enterprise Data Center Topology

Considerations	Deployment Assumptions
Management	<ul style="list-style-type: none"> ■ Oracle Enterprise Manager Application Server Control installed automatically with each installation ■ Use Application Server Control for: <ul style="list-style-type: none"> Starting and stopping components as needed Enabling/disabling components to minimize system resource consumption Setting or changing configuration parameters for any of the application server components Deploying and configuring applications Managing application security Monitoring application and component performance and resource consumption in real time Viewing and setting ports Browsing and searching log files Managing infrastructure schemas ■ Command line utilities are available for scripting and automation ■ Backup and recovery: complete cold backup of the entire distributed environment
Security	<ul style="list-style-type: none"> ■ Global OID/SSO or logical SSO (one or more SSO) sharing the same logical OID (one or more OID) ■ Both SSO and OID behind the external firewall for internal users ■ When hosting applications for both internal and external users (such as MOC), security considerations will need to make sure that some security services can be shared by both types of users ■ Integrating with departmental third party directories such as iPlanet, ActiveDirectory, or eDirectory ■ Provisioning and de-provisioning users

Table 12–5 Key Considerations for the Enterprise Data Center Topology

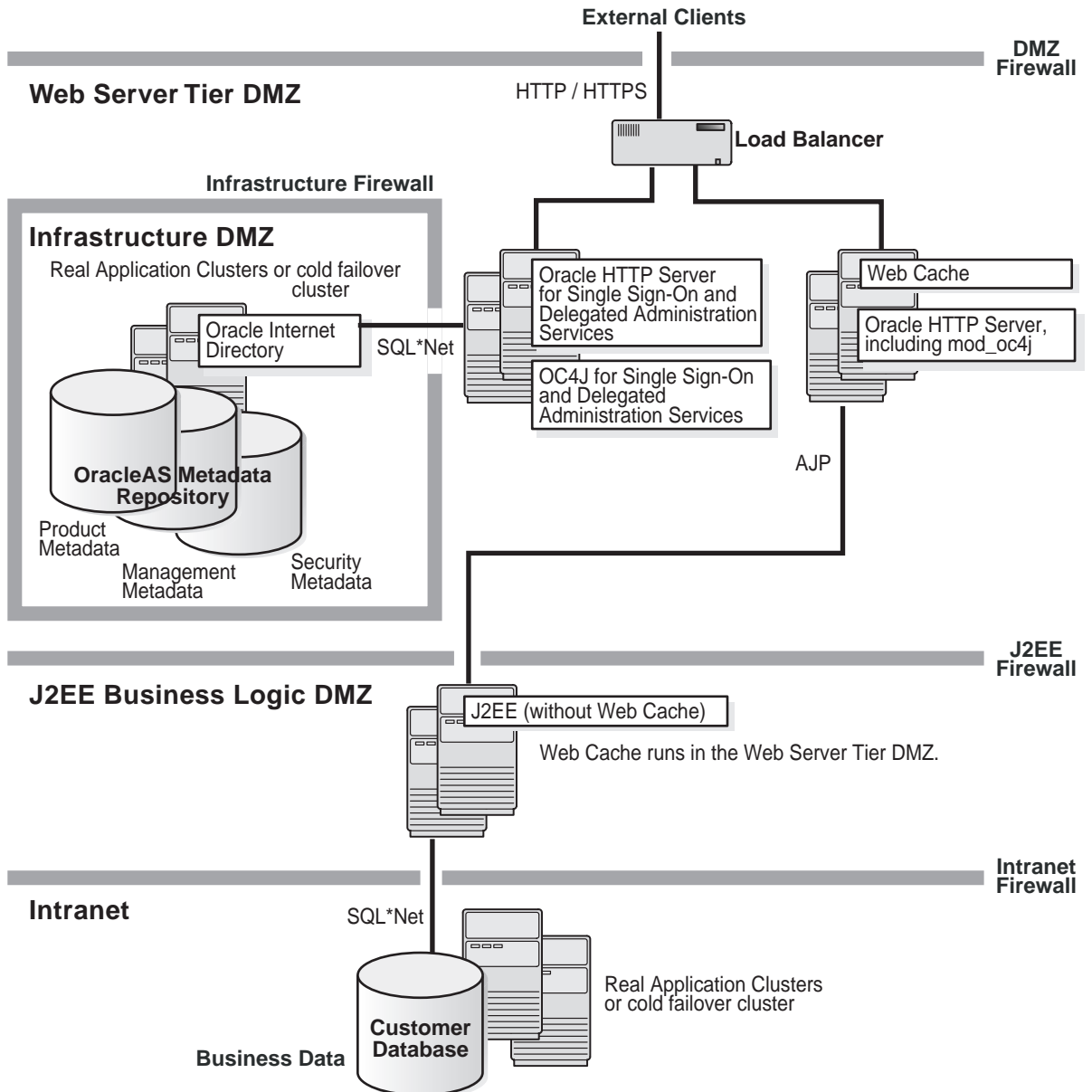
Considerations	Deployment Assumptions
Application Deployment and Performance	<ul style="list-style-type: none"> ■ J2EE applications deployed on Oracle Application Server clusters with or without Web Cache ■ Portal application using Web Cache, even on a single node environment ■ Forms applications working against an OLTP system with no SSO ■ BI applications working against a data warehouse with tighter security ■ All applications accessible by Portal and Wireless devices ■ Self Service Applications using IP and Workflow
High Availability	<ul style="list-style-type: none"> ■ Infrastructure HA: Multiple types of HA solutions for different Infrastructure Services ■ Optional: OPMN-based cluster management for middle tier applications
Third Party Products	<ul style="list-style-type: none"> ■ Firewall, load balancers, hardware clusters, hardware accelerators

Enterprise Data Center Topology for Java Applications

The following recommendation assumes that you want to create new databases for Product, Management, and Security services. If you choose to use an existing database, the product metadata will inherit the high availability solution already deployed for that database.

Figure 12–5 illustrates the recommended topology for the enterprise data center environment when deploying Java applications.

Figure 12-5 Enterprise Data Center Topology for Java Applications

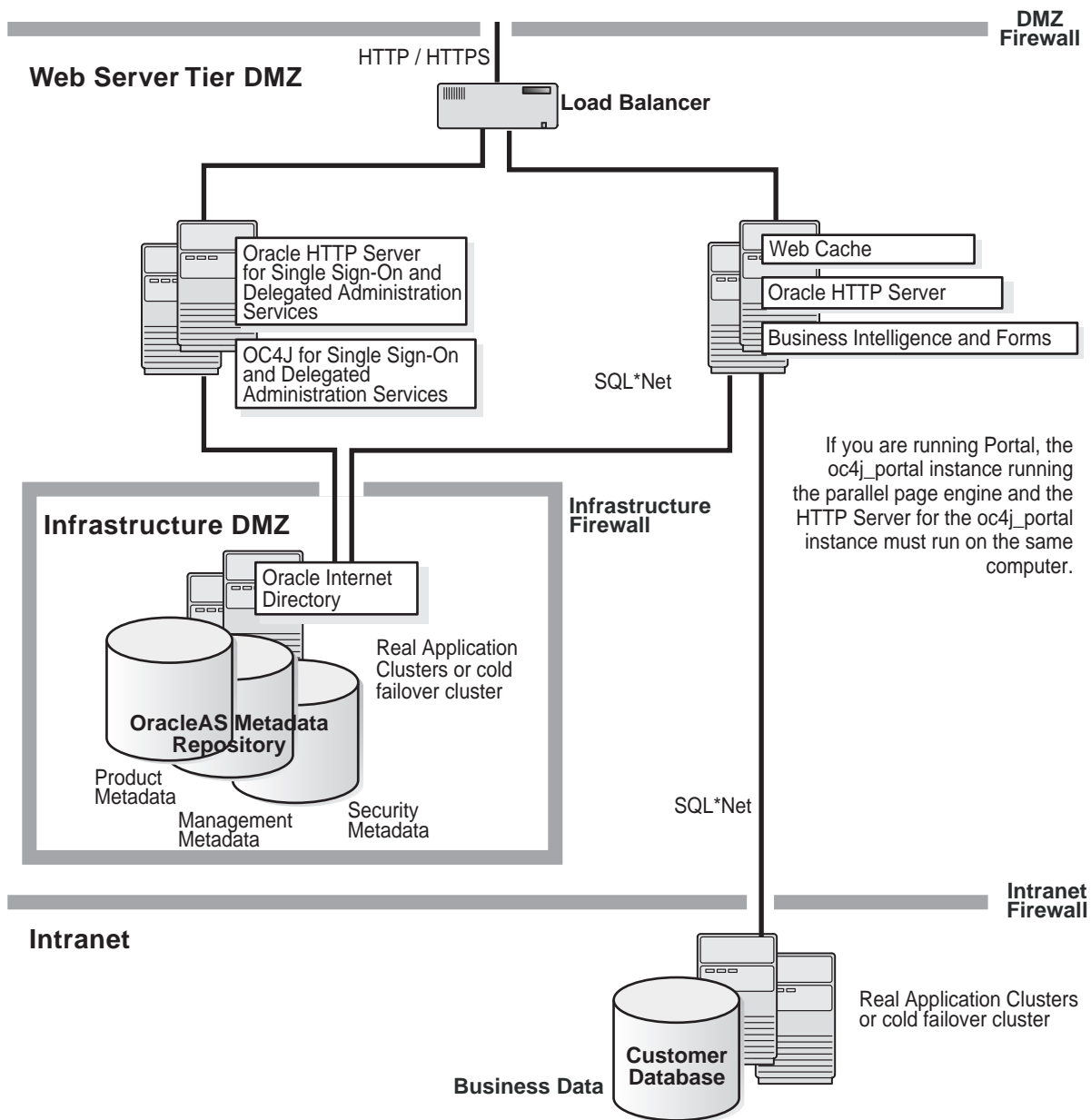


Enterprise Data Center Topology for Portal, Wireless, Business Intelligence, and Forms Applications

The following recommendation assumes that you want to create new databases for Product, Management, and Security services. If you choose to use an existing database, the product metadata will inherit the high availability solution already deployed for that database.

Figure 12–6 illustrates the recommended topology for the enterprise data center environment when deploying Portal, Wireless, Business Intelligence, and Forms applications.

Figure 12-6 Enterprise Data Center Topology for Portal, Wireless, Business Intelligence, and Forms Applications



Departmental Topology: Departments Hosting Their Applications

The Departmental topology consists of a subset of considerations and requirements from the Enterprise Data Center topology. Table 12–6 outlines the key considerations for using the Departmental topology.

Table 12–6 Key Considerations for the Departmental Topology

Considerations	Deployment Assumptions
Install	<ul style="list-style-type: none"> <li data-bbox="615 473 1210 526">■ Multiple host installations on cluster machines, NFS machines <li data-bbox="615 539 1033 565">■ Multiple middle tier instances used <li data-bbox="615 579 1225 631">■ No Infrastructure used if deploying only Java or J2EE applications <li data-bbox="615 645 1162 697">■ Dedicated Product Metadata Services for Portal applications <li data-bbox="615 711 1253 737">■ Shared Product Metadata Services for some applications <li data-bbox="615 751 1119 803">■ Shared Security Services to secure subset of enterprise-level users <li data-bbox="615 817 891 843">■ Management Services <li data-bbox="615 857 1243 937">■ Oracle Application Server should withstand hard disk replacement, CPU change, RAM upgrades, or Network interfaces

Table 12–6 Key Considerations for the Departmental Topology

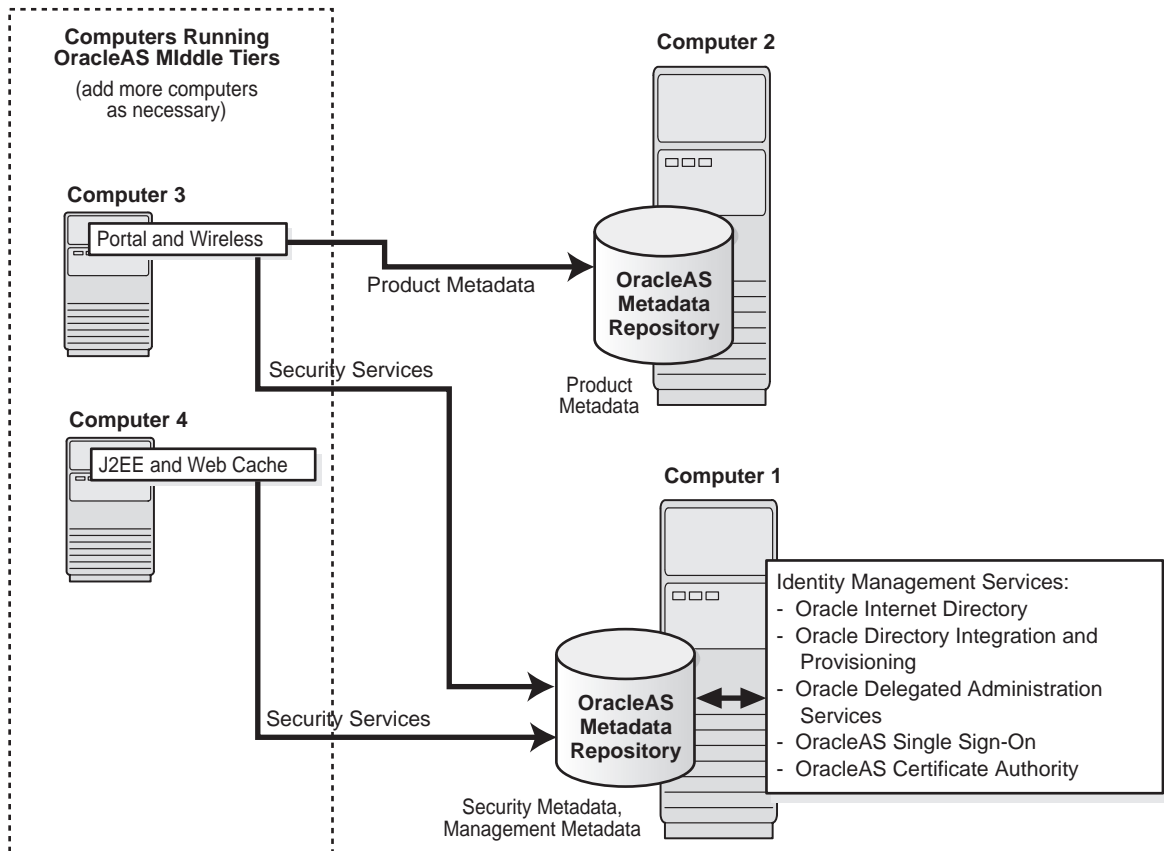
Considerations	Deployment Assumptions
Management	<ul style="list-style-type: none"> ■ Oracle Enterprise Manager Application Server Control installed automatically with each installation ■ Use Application Server Control for: <ul style="list-style-type: none"> Starting and stopping components as needed Enabling/disabling components to minimize system resource consumption Setting or changing configuration parameters for any of the application server components Deploying and configuring applications Managing application security Monitoring application and component performance and resource consumption in real time Viewing and setting ports Browsing and searching log files Managing infrastructure schemas ■ Command line utilities are available for scripting and automation ■ Backup and recovery: complete cold backup of the entire distributed environment
Security	<ul style="list-style-type: none"> ■ Single install that would contain both Infrastructure Software and OID/SSO data ■ Contains subset of users as compared to the enterprise OID
Application Deployment and Performance	<ul style="list-style-type: none"> ■ Important not to pay overhead for enterprise topology services ■ OHS used as load balancer for multiple OC4J instances ■ J2EE applications deployed on Oracle Application Server clusters with or without Web Cache ■ Portal applications using Web Cache

Table 12–6 Key Considerations for the Departmental Topology

Considerations	Deployment Assumptions
High Availability	<ul style="list-style-type: none"> ■ HA requirement for departmental deployment depends on the nature of the application ■ If there is a requirement, recommendation is to use Local Data Guard or Cold Failover Cluster ■ If there is no requirement, complete cold backup and recovery methodology is used
Third Party Products	<ul style="list-style-type: none"> ■ Depending on the load on the application, load balancers might be needed

Figure 12–7 illustrates the recommended topology for the Departmental topology.

Figure 12–7 Departmental Topology



Development Life Cycle Support Topology: Test to Stage to Production Support

There might be a situation using either the Enterprise Data Center and Departmental topologies that would require complete support for the development life cycle. This consists of seamless support to move an application from the test environment to the staging environment, and from there to the production environment. Table 12–7 outlines the key considerations for supporting the entire development life cycle.

Table 12–7 Key Considerations for the Development Life Cycle Support Topology

Considerations	Deployment Assumptions
Install	<ul style="list-style-type: none"> ■ Test Environment: Single host for middle tier and Infrastructure (all services from one database) ■ Staging Environment: Multiple middle tiers on one single large machine or on multiple machines, with either dedicated or shared Product Metadata Services, but always shared Security Services ■ Production Environment: Very similar to the staging environment, except now using enterprise-wide Security Services ■ Oracle Application Server should withstand hard disk replacement, CPU change, or RAM upgrades
Management	<ul style="list-style-type: none"> ■ Test Environment: For application server installations use Application Server Control. For standalone components use command line tools. ■ Staging Environment: Application Server Control ■ Production Environment: Application Server Control
Security	<ul style="list-style-type: none"> ■ Moving security requirements ■ Re-association of Security Services is required
Application Deployment and Performance	<ul style="list-style-type: none"> ■ Shutdown, startup, and deploy times are priorities ■ Frequent reconfiguration of tunable parameters needs to be fast ■ May have multiple versions installed and running ■ Load balancing and application combinations will be tested on one machine
High Availability	<ul style="list-style-type: none"> ■ Test Environment: Not a concern for the test environment itself, applications and specific configuration files will be backed up; however, HA solutions may need to be tested ■ Staging Environment: Cold Failover Cluster or Local Data Guard, complete cold backup ■ Production Environment: RAC or Cold Failover Cluster and Remote Data Guard for disaster recovery, complete cold backup
Third Party Products	<ul style="list-style-type: none"> ■ Depending on the load on the application, DMZ, firewalls, load balancers, and routers might be needed

The Development Life Cycle Support topology is a combination of other topologies which supports moving applications from test to stage to production environments.

- Test environment: Application developers test their applications in their own environments: Examples of testing environments:
 - Java Developer Topology: Low-end Machines with Java IDE Tools
 - Portal and Wireless Developer Topology: Medium-sized Machines
 - Forms, Reports, and Discoverer Developer Topology: Medium-sized Machines
- Stage environment: QA personnel test all applications before deploying them to the production environment. In this environment, you can use the Departmental Topology: Departments Hosting Their Applications. This topology in a stage environment runs applications from all departments, not just from a single department.
- Production environment: Applications are ready for user by both internal and external users. See Enterprise Data Center Topology: Multiple Departments Sharing the Same Data Center.

Special Topologies

This section describes the following special topologies:

- High Availability Deployments

High Availability Deployments

In Oracle Application Server 10g (9.0.4), all Oracle High Availability (HA) solutions, including Cold Failover Cluster, Data Guard, and RAC, are supported for the Infrastructure.

The topologies described in this section can be combined with any of the general deployment topologies discussed in the previous sections. For example, the following recommended topologies assume that all Infrastructure services are in the same `ORACLE_HOME`. However, HA deployments can fit into any Infrastructure distributed topology that has been presented in this chapter.

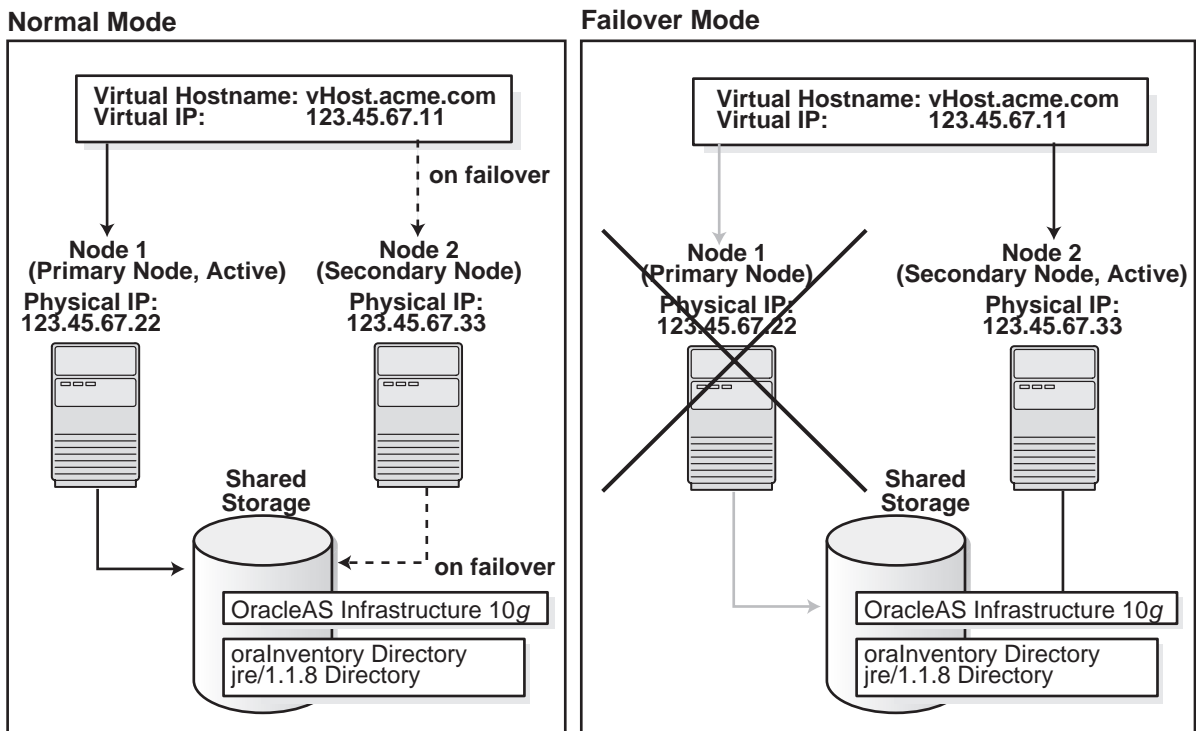
Cold Failover Cluster

Highlights of Cold Failover Cluster include the following:

- Support for all cluster vendors that the database supports, including Sun Clusters, HP Service Guard, and Veritas
- There can be any number of file systems making up a shared file location
- ORACLE_HOME and Oracle Application Server configuration files are on the shared system
- The virtual IP address and virtual hostname are enabled on the active node of the cluster; the ORACLE_HOME should also be in the shared disks

Figure 12–8 illustrates the recommended topology for the HA deployment using Cold Failover Cluster.

Figure 12–8 Cold Failover Cluster



Active Failover Clusters

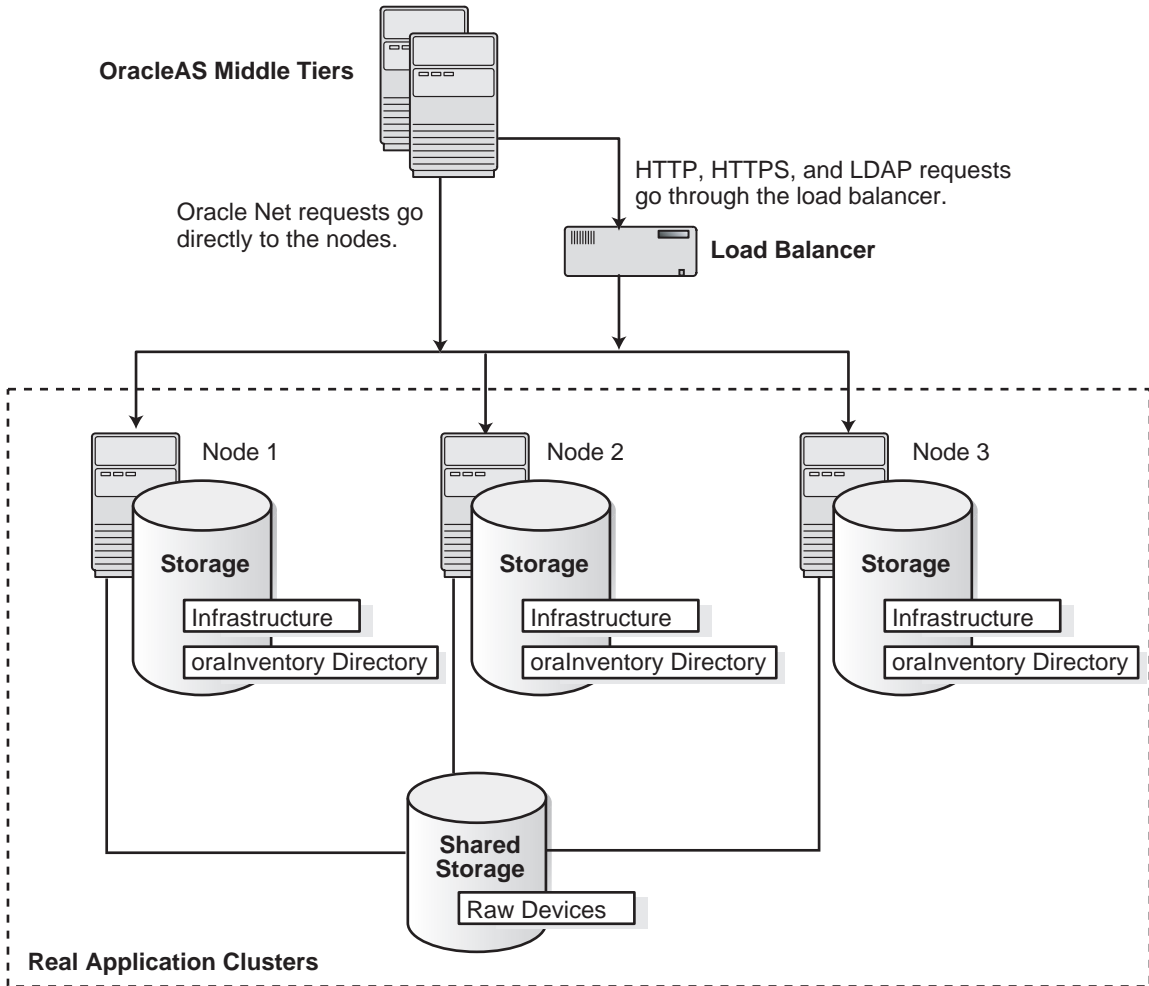
In the initial release of Oracle Application Server 10g (9.0.4), **Active Failover Cluster** is a **Limited Release** feature. Please check Oracle *MetaLink* (<http://metalink.oracle.com>) for the most current certification status of this feature or consult your Sales Representative, before deploying this feature in a production environment.

Highlights of Active Failover Clusters include the following:

- Support for Active/Active Mode
- Infrastructure software on both nodes
- Configuration files are present at both nodes

Figure 12–9 illustrates the recommended topology for the high availability deployment using Active Failover Clusters.

Figure 12–9 Active Failover Clusters



Glossary

A2A

Application to application (A2A) integration (also known as enterprise **application integration**) is the integration of applications and business processes within the same company (also known as an enterprise).

access control list

See **ACL**.

access control list model

The access control list model is a method for organizing **authorization** information. The access control list model is resource-centered or **object**-centered. This means that authorization information is associated with objects.

ACID characteristics

A **transaction** has ACID characteristics (Atomic, Consistent results, Isolated, and Durable), if it demonstrates the following:

- If interrupted by failure, all effects are undone or rolled back (Atomic).
- The effects of a transaction preserve invariant properties (Consistent results).
- Its intermediate states are not visible to other transactions. Transactions appear to execute serially, even if they are performed concurrently (Isolated).
- The effects of a completed transaction are persistent; they are never lost (except in a catastrophic failure) (Durable).

ACL

An access control list (ACL) is a list of groups and users authorized for specific access to an **object**. It can also be a list of entities, together with their access rights, which are authorized to have access to a specified resource.

Apache

Apache is a public domain **HTTP server** derived from the National Center for Supercomputing Applications (NCSA).

Apache module

An Apache module is an add-on to either **Oracle HTTP Server** or **Apache**. Modules extend the basic functionality of the **Web server** and support integration between the **Oracle HTTP Server** and other Oracle Application Server components.

API

An application program interface (API) is a set of exposed data structures and functions that an application can use to invoke services on a component.

applet

An applet is a **Java** program that runs from an applet viewer or a Web page.

application integration

Application integration is the ability to link different types of enterprise applications and business processes together so that they can smoothly and effectively communicate to conduct **e-business**. These applications can reside within a company's enterprise boundaries or across multiple company boundaries connected over the Internet.

application-to-application

See **A2A**.

authentication

Authentication ensures that access to static pages, **CGI** scripts, and applications is limited to authorized users. When access to an application or document is protected by an authentication scheme, the **client** sends identification information to the **server**, which checks if the client is authorized to access the **object**.

See Also: security scheme, restriction

authentication broker

An authentication broker is the portion of the **authentication server** that responds to and evaluates **authorization** requests.

authentication provider

An authentication provider is an **object** that specifies all of the **realms** used to implement a particular **security scheme**. It is a code **module** that runs within the **authentication server** and implements a particular security scheme.

authentication server

An authentication server is an **object** that encapsulates the **authentication** performed against applications. It consists of one **authentication broker** object and several **authentication provider** objects.

authorization

Authorization is the evaluation of security constraints to send a **message** or make a request. Authorization uses specific criteria to determine whether the request should be permitted. The criteria are **authentication** and **restriction**.

See Also: **authentication, restriction**

automatic deployment

Automatic deployment is a method for re-deploying applications, **servlets**, or **JSPs** after changes have been made to the application code that does not require updating the **server** configuration files or restarting the application server. With automatic deployment, the application server detects the changes in code and automatically re-deploys the applications.

See Also: **hot deployment** for comparison

availability

Availability is the percentage or amount of scheduled time that a computing system provides application service.

B2B

Business-to-business (B2B) integration is the integration of a company's applications and business processes with external business systems operated by customers, suppliers, trading partners, exchanges, and marketplaces. B2B integration extends **A2A** integration to the Internet.

basic authentication

Basic authentication is a username-and-password-based **authentication** scheme that does not encrypt passwords when sending them over the Internet.

See Also: **digest authentication, authentication**

bean-managed persistence (BMP) bean

A bean-managed persistence (BMP) bean is a **Java** bean that stores all state information within itself.

browser client

A browser client is a client that can access static pages, **CGI** scripts, and applications via a **URL** over **HTTP**, **HTTPS (secure HTTP)**, or **IIOP**.

business intelligence

Business intelligence is information describing your business, data, and Web site traffic.

business-to-business

See B2B.

CA

A certificate authority (**CA**) is a trusted third party that vouches for the identity of an individual, company, or **server** and signs a **certificate**.

See Also: **trustpoint**

caching

Caching is the act of storing information that is frequently accessed in a location where it can be accessed quickly. For example, **Oracle Application Server Web Cache** stores dynamically generated Web pages locally and serves them in response to incoming requests. This reduces the total time spent handling the request by avoiding costly connections to the back-end database and other Web site bottlenecks.

canonical

Authoritative or officially approved. The term is generally used to describe whether or not a programming interface follows an accepted standard.

capability model

A capability model is a method for organizing **authorization** information. The capability model is **principal**-centered or **subject**-centered. This means that authorization information is associated with subjects. The **Java 2 Security Model** is an example of a capability-based system.

cascading style sheets

See **CSS**.

certificate

A certificate is a specially formatted data item signed by a trusted third party to attest to the validity of the item's information. Public-key certificates use a **CA**'s signature to attest that the enclosed **public key** belongs to the **principal** identified by the enclosed name.

certificate authentication

Certificate authentication is an **authentication** method in which **clients** identify themselves using X.509 v3 **certificates**.

certificate authority

See **CA**.

certificate revocation list

See **CRL**.

CGI

Common Gateway Interface (CGI) is the industry-standard technique for transferring information between a Web **server** and any program designed to accept and return data that conforms to the CGI specifications.

character set

A character set is a set of symbols used to write one or more human languages, as defined by **RFC 1521**.

clickstream

A clickstream is a virtual trail that a user leaves behind while moving from place to place on the Internet. A clickstream is a record of a user's activity on the Internet, including every Web site and every page of every Web site that the user visits, how long the user was on a page or site, in what order the pages were visited, any

newsgroups that the user participated in, and even the e-mail addresses of mail that the user sends and receives.

client

A client is a user, software application (such as a browser), or computer that requests the services, data, or processing of another application or computer (the **server**).

cluster

A cluster is a collection of application server **instances** with identical configuration and application deployment. Clusters enforce homogeneity between member **instances** so that a cluster of application server instances can appear and function as a single instance. With appropriate front-end load balancing, any instance in an application server cluster can serve **client** requests. This simplifies configuration and deployment across multiple instances and enables **fault tolerance** among clustered instances.

clustering

Clustering is the process of collecting Oracle Application Server **instances** into **clusters** for load balancing and **fault tolerance**. With clustering, the collection of application server **instances** are treated as a pool to service incoming requests. If one instance does not respond, then the request is forwarded to another instance in the cluster. The load balancer also maintains session context, so that when a **client** reconnects, its request is sent to the application server instance that was previously serving it.

collaboration protocol agreements

See CPA.

collaboration protocol profile

See CPP.

common gateway interface

See CGI.

common object request broker architecture

See CORBA.

concurrency

Concurrency is the ability to handle multiple requests simultaneously. Threads and processes are examples of concurrency mechanisms.

connect string

A connect string is the set of parameters, including a protocol, that is used to connect to a specific database instance on the network. Other names for a connect string include: SQL*Net V2 service name and Net8 connect string.

container

A container is a component that contains other components, such as a **servlet**. A container executes and manages a servlet. A container is either part of or associated with and used by a **Web server**. When a **client HTTP** request calls a servlet, the Web server passes the HTTP request to the container. The container translates the HTTP request into a **Java** method invocation and then passes the request to the servlet.

See Also: JSP translator, servlet container

container-managed persistence (CMP) bean

A container-managed persistence (CMP) bean is a **Java** bean that stores all state information in the **container**.

contention

Contention is a measure of competition for resources.

cookie

A cookie is a text string that is stored on the **client** browser by the **server** to maintain state between **HTTP** calls. Cookies enable applications to store and retrieve information about a client, such as the domain, path, lifetime, and other variables. Cookies can either expire when the user exits the browser or at a date specified by the creator of the cookie.

CORBA

Common Object Request Broker Architecture (CORBA) is an industry standard for allowing code modules called “**objects**” to communicate with one another regardless of the programming language in which they are written or the operating system on which they are running.

CORBA object

CORBA object is a generic term for a **server-side** program that conforms to the **OMG's CORBA** specification. Objects can be written in any language, deployed on any machine, and can exist locally or over a wide-area network.

CPA

A collaboration protocol agreement (CPA) documents the technical agreement between two or more trading partners to engage in electronic business collaboration.

CPP

A collaboration protocol profile (CPP) defines one business partner's technical capabilities to engage in electronic business collaborations with other partners to exchange electronic messages.

CRL

A Certificate Revocation List (CRL) is a list of **certificates** that have been revoked before their scheduled expiration date. CRLs only list revoked certificates. When a revoked certificate is past its original expiration date, it is removed from the CRL.

CRM

Customer relationship management (CRM) consists of the methodologies, software, and usually Internet capabilities that help an enterprise manage customer relationships in an organized way.

CSS

Cascading style sheets (CSS) provide a simple mechanism for adding style, such as fonts, colors, and spacing, to **XML** documents.

customer database

A customer database is the original and primary database for storing your data. The customer database is commonly located on the database **server** tier in a **three-tier architecture**.

customer relationship management

See **CRM**.

DAD

A database access descriptor (DAD) is a set of values that specify how an application connects to an Oracle database to fulfill an **HTTP** request. The

information in the DAD includes the username (which also specifies the **schema** and the **privileges**), password, connect-string, error log file, standard error message, and national language support (NLS) parameters such as NLS language, NLS date format, NLS date language, and NLS currency.

database access descriptor

See DAD.

default DAD

The default DAD is the database access descriptor that a **PL/SQL** application uses when the configuration information in the application does not specify a DAD.

default MIME type

The default MIME type is the **MIME type** the **HTTP listener** uses to interpret requested files of an unsupported MIME type.

See Also: **MIME type**

demilitarized zone

See DMZ.

deployment descriptor

A deployment descriptor (DD) is an **XML** text-based file with an `.xml` extension that describes the deployment settings for a component. A **J2EE** application and each of its modules has its own deployment descriptor (DD). For example, an enterprise bean module DD declares **transaction** attributes and security **authorizations** for an enterprise bean. DD information is declarative so it can be changed without modifying the bean source code. At runtime, the **J2EE Server** reads the DD and acts on the component accordingly.

digest authentication

Digest authentication is an **authentication** scheme that does not send passwords over the Internet. Digest authentication is safer than **basic authentication**, but is not supported by most browsers.

See Also: **authentication, basic authentication**

digital signature

A digital signature is a code attached to an electronic document that reliably identifies the author or sender, and verifies that the document has not been tampered with.

directory server

A directory server defines a hierarchical view of an organization's employees, units, and other resources. You can protect applications using directory servers by limiting access to the virtual paths of the applications to particular branches in the directory server.

distinguished name

A distinguished name is the unique name of an LDAP-based directory entry. A distinguished name comprises all of the individual names of the parent entries back to the root.

distributed transaction processing

See DTP.

DLL

A dynamic link library (DLL) is an archive of executable functions or data that can be used by a Microsoft Windows application. Typically, a DLL provides one or more particular functions and a program accesses the functions by creating either a static or dynamic link to the DLL. A DLL can be used by several applications at the same time.

DMZ

The demilitarized zone (DMZ) is the area between outer and inner **firewalls**. It is normally used to protect the internal application servers from being attacked by those attempting to gain unauthorized access to a network or intranet.

DNS

The domain name system (DNS) is the mechanism that divides the Internet into separate, hierarchical groups called domains, identified by unique alphanumeric names, such as `us.oracle.com`. DNS identifies each computer within a domain by a unique **hostname**. For example, a computer named `hal` in the `us.oracle.com` domain would be uniquely identified on the Internet as `hal.us.oracle.com`.

document type definition

See DTD.

domain-based restriction

Domain-based restriction is a **restriction** scheme that allows or denies access to files based on the **client** machine's domain name.

See Also: restriction, IP-based restriction

domain name system

See DNS.

DTD

A document type definition (DTD) is a set of rules that define the allowable structure of an **XML** document. DTDs are text files that derive their format from SGML and are either embedded within an XML document or referenced by an XML document.

DTP

Distributed transaction processing (DTP) is the protocol that guarantees a two-phase commit when multiple databases are involved in a **transaction**.

dynamic link library

See DLL.

e-business

E-business is the conduct of business over the Internet, such as buying and selling products, servicing customers, and collaborating with business partners.

EAI

Enterprise application integration (EAI) is the sharing and transferring of information and applications between systems so that they appear as a unified application.

EAR file

An Enterprise Archive (EAR) file is a standard JAR file with an `.ear` extension. A **J2EE** application with all of its modules is delivered in an EAR file. An EAR file contains the JAR and **WAR files** that comprise a J2EE application.

ebXML

Electronic Business Extensible Markup Language (ebXML) is a modular suite of specifications that enables enterprises of any size and in any geographical location

to conduct business on the Internet. By using ebXML, companies have a standard method for exchanging business messages, conducting trading relationships, and communicating data in common terms.

EDI

Electronic data interchange (EDI) is the computer-to-computer exchange of business data in standard formats. In EDI, information is organized according to a specified format that is set by both parties, allowing a computer transaction that requires no human intervention. All information contained in an EDI transaction set is, for the most part, the same as on a conventionally printed document. It defines the data formats and encoding rules that are required for a number of business transactions, including order placement and processing, shipping and receiving, invoicing, and payment systems.

EIS

Enterprise information systems (EIS) are systems that provide the information infrastructure for an enterprise. Enterprises run their businesses using the information stored in these systems. Examples of enterprise information systems include **enterprise resource planning** systems, mainframe **transaction** processing systems, relational database management systems, and other legacy information systems. Enterprise applications require access to applications running on enterprise information systems.

EJB

Enterprise JavaBeans (EJB) are the component-based application model for **Java** defined by JavaSoft. This model provides most of the system-level services, such as multi-threading, to ease application programming. EJB relies on various standardized enterprise services, such as **JNDI**, **JTS**, and **JDBC**, to facilitate application programming and enable **EJB objects** to be interoperable across various EJB servers. It fulfills the write once, run anywhere paradigm.

EJB application

An EJB application is a framework of deploying **CORBA objects** written in **Java**, which adhere to the **EJB** specification.

EJB container

An EJB container is the component coordinator in an **EJB application** and one of the key **EJB** runtime components.

See Also: **JNDI, EJB application**

EJB deployment descriptor

An EJB deployment descriptor is a serialized **object** that provides information, such as **transaction** and security policies, about how an **EJB application** or object should be deployed.

Electronic Business Extensible Markup Language

See ebXML.

electronic data interchange

See EDI.

encapsulation

Encapsulation is the mechanism that binds code together with the data that it manipulates. Encapsulation provides a wrapper that keeps both the code and the data safe from outside intervention.

encoding

Encoding is an algorithm used to alter a file's format, such as compression.

encryption

Encryption is the practice of **encoding** (encrypting) data in such a way that only an intended recipient can decode (decrypt) and read the data.

See Also: public-key encryption, shared secret-key encryption

enterprise application integration

See EAI.

enterprise beans

Enterprise beans are **server-side** components that encapsulate the business logic of an application, which is the code that fulfills the purpose of the application.

enterprise information systems

See EIS.

Enterprise JavaBeans

See EJB.

enterprise resource planning

See **ERP**.

entity bean

An entity bean is a complex business entity. An entity bean models a business entity or models multiple actions within a business process. Entity beans are often used to facilitate business services that involve data and computations on that data. For example, an application developer might implement an entity bean to retrieve and perform computation on items within a purchase order. Your entity bean can manage multiple, dependent, persistent **objects** in performing its necessary tasks.

ERP

The set of activities supported by multi-module application software that helps a manufacturer or other business manage important parts of their business, including product planning, parts purchasing, inventory management, supplier interaction, customer service, and order tracking.

extract, transform, and load (ETL) capabilities

The three database functions of extract, transform, and load are combined into one function and used to retrieve data from one database and write it to another.

- **Extract:** the retrieving of data from a data source
- **Transform:** the converting of the extracted data from its existing format into the format of the target database, using pre-defined rules
- **Load:** the writing of the transformed data into the target database

failover

Failover is the ability to reconfigure a computing system to utilize an alternate active component when a similar component fails.

failure recovery

Failure recovery is a system of failure detection and recovery. Components monitor each other continuously. When a component fails, Oracle Application Server detects the failure and restarts the failed component, restoring any preserved state information when possible.

farm

A farm is a collection of **clusters** and **instances** that share the same **Oracle Application Server Infrastructure**.

fault tolerance

Fault tolerance is the ability of a computing system to withstand errors while continuing to provide the required services.

file protection

File protection is the practice of assigning an **authentication** or **restriction** scheme by controlling access to a specific file or group of files.

filename extension

A filename extension is a short alphanumeric suffix attached to a filename following a dot "." that represents the file's format. Oracle Application Server uses filename extensions to identify several kinds of file formats, including **MIME types** and **encodings**.

firewall

A firewall is a machine that acts as an intermediary to protect a set of computers or networks from outside attack. It regulates access to computers on a local area network from outside, and regulates access to outside computers from within the local area network. A firewall can work either by acting as a **proxy server** that forwards requests so that the requests behave as though they were issued by the firewall machine, or by examining requests and attempting to eliminate suspect calls.

graphical user interface

See GUI.

GUI

A graphical user interface (GUI), sometimes referred to as the UI (user interface), is the graphical, as opposed to purely textual, user interface to a computer. Elements of the GUI include pull-down menus, buttons, icons, windows, and graphics.

Health Level Seven

See HL7.

HL7

Health Level Seven (HL7) is a standard for electronic data exchange in health care environments. It focuses on health care in the clinical and administrative data domain. "Level Seven" refers to the highest level of the International Standard Organization's (ISO) communications model for Open Systems Interconnection (OSI), the application level.

host

A host is a computer with a unique domain name.

hosted applications

Hosted applications are typically applications developed by independent software vendors and administered by application service providers. These applications are accessed on an external Web site that enables multiple companies to utilize the applications.

hosted environment

A hosted environment is an application deployment environment in which multiple customers and companies subscribe to shared services.

hosted services

Hosted services are services are offered by application service providers for multiple **subscribers**.

hostname

A hostname is a character string that uniquely identifies a computer within a **DNS** domain.

hot deployment

Hot deployment is a method for deploying new applications that does not require restarting the application server. With hot deployment, changes made to the **server** configuration files are automatically detected, and you can deploy new applications without restarting the server.

See Also: [automatic deployment](#) for comparison

HTML

Hypertext markup language (HTML) is a format for **encoding** hypertext documents that may contain text, graphics, and references to programs and other hypertext documents.

HTTP

Hypertext Transfer Protocol (HTTP) is the underlying format used by the Web to format and transmit **messages** and determine what actions Web **servers** and browsers should take in response to various commands. HTTP is the protocol used between Oracle Application Server and **clients**.

HTTP header

An HTTP header is a body of information that a browser sends along with a **URL** when requesting a Web page. It includes such information as the browser type and **MIME types**.

HTTP listener

See **listener**.

HTTP request information

HTTP request information is the information requested by a **client** in the form of an **HTTP header**.

HTTP response information

HTTP response information is the information supplied by an **HTTP listener** or an application.

HTTP server

An HTTP server is a **server** that receives HTTP requests from remote browsers, converts the requested **URL** to a filename, and returns the file to the requester.

HTTPS (secure HTTP)

HTTPS is a version of **HTTP** with provisions for secure data transmission.

See Also: **HTTP**

hypertext markup language

See **HTML**.

hypertext transfer protocol

See **HTTP**.

IBAC

Identity-based access control (IBAC) is the use of digital IDs to control access to a resource.

IDE

An integrated development environment is a visual tool containing editors, debuggers, screen painters, **object** browsers etc.

identity-based access control

See IBAC.

IDL

Interface definition language (IDL) is a standard language for interface specification primarily used for **CORBA object** interface definition. IDL is declarative and does not reveal the implementation of a CORBA object. **CORBA** defines standard mappings from IDL to various programming languages.

IIOp

Internet inter-ORB protocol (IIOp) is an Internet transport protocol used by **CORBA objects** to communicate with each other. In the context of Oracle Application Server, IIOp is used by **ECO/Java** and **EJB objects**. IIOp is also used between Oracle Application Server components.

inheritance

Inheritance is the process by which one class acquires the methods and properties of another in **object-oriented programming** methodology.

instance

An application server instance is the set of processes required to run the configured components within an application server installation. There can be only one application server instance per application server installation. The terms *installation* and *instance* are sometimes used interchangeably; however, it is important to remember that an installation is the set of files installed into an Oracle home and an instance is a set of processes associated with those files.

integrated development environment

See IDE.

interface definition language

See IDL.

Internet Information Server (IIS)

Internet Information Server (IIS) is Microsoft's Web server that runs on Windows NT/2000 platforms.

Internet inter-ORB protocol

See IIOp.

interoperable object reference

See **IOR**.

IOR

Interoperable object reference (IOR) is a unique string for each **CORBA object** and is created when an **object reference** is passed among different **ORBs**. You can pass this string to a method to determine the actual object reference.

See Also: **object reference**

IP address

An IP address is a four-part number separated by periods that uniquely identifies a computer on the Internet; the number format is defined by the Internet Protocol (IP).

IP-based restriction

IP-based restriction is a **restriction** scheme that allows or denies access to files based on the **client machine's IP address**.

See Also: **restriction**

island

An island is a logical grouping of **OC4J** processes that allows you to determine which OC4J processes will replication state.

J2EE

Java 2 Platform, Enterprise Edition (J2EE) is a platform that enables application developers to develop, deploy, and manage multi-tier, **server-centric**, enterprise level applications.

J2EE Server

The J2EE Server is the runtime portion of a **J2EE** product, which provides **EJB containers**, Web containers, or both. J2EE servers are usually located in the middle tier of a **three-tier architecture**.

J2SE

Java 2 Platform, Standard Edition (J2SE) is a platform that enables application developers to develop, deploy, and manage **Java applets** and applications on a desktop **client** platform such as a personal computer or workstation.

JAAS

Java Authentication and Authorization Service (JAAS) is a **Java package** that enables services to authenticate and enforce access control upon users. JAAS implements a Java version of the standard Pluggable Authentication Module (PAM) framework, and extends the access control architecture of the **Java 2 Security Model** to support user-based **authorization**.

JAR files

There are different types of Java Archive (JAR) files. An **EJB JAR** file contains its **deployment descriptor** (DD), related files, and the `.class` files for the enterprise bean. An application **client JAR** file contains its DD, related files, and the `.class` files for the application client. JAR files are packaged together with **WAR files** into **EAR files**.

Java

Java is a programming language developed by Sun Microsystems. This language is fully **object-oriented**, extremely portable, and optimized for creating distributed applications on the Internet or other computer networks.

Java 2 Platform, Enterprise Edition

See J2EE.

Java 2 Platform, Standard Edition

See J2SE.

Java 2 Security Model

The Java 2 Security Model provides developers and administrators with increased control over many aspects of enterprise **applet**, component, **servlet**, and application security. The Java 2 Security Model is capability-based and enables you to establish **protection domains**, and set security policies for these domains.

Permissions are the basis of the Java 2 Security Model. All **Java** classes (whether run locally or downloaded remotely) are subject to a configured security **policy** that defines the set of permissions available for those classes. Each permission represents a specific access to a particular resource.

Java Authentication and Authorization Service

See JAAS.

JavaBeans

JavaBeans is a portable, platform-independent component model that enables developers to write reusable components once and run them anywhere.

See Also: Enterprise JavaBeans

Java Connector Architecture

See JCA.

Javadoc

Javadoc is a tool for generating API documentation in HTML format from documentation comments in Java source code. These HTML pages describe the classes, inner classes, interfaces, constructors, methods, and fields.

Java interpreter

A Java interpreter is a program that interprets and executes Java bytecode independently of a Web browser.

Java Message Service

See JMS.

Java Naming and Directory Interface

See JNDI.

JavaServer Pages

See JSP.

Java Transaction API

See JTA.

Java Transaction Service

See JTS.

Java Virtual Machine

See JVM.

JDBC

Java DataBase Connectivity (JDBC) is a Java package that provides connectivity to databases from within Java.

JCA

The Java Connector Architecture (JCA) provides a standard architecture for integrating heterogeneous Enterprise Information Systems (**EIS**).

JIT compilation

Just-in-time (JIT) compilation is the process by which the Java Virtual Machine (JVM) keeps a copy of native code that it generates from bytecode the first time a method is encountered. Subsequently, when the method is run, the JIT uses the native code without having to interpret the method, resulting in a boost in performance.

JMS

The Java Message Service (JMS) API provides a reliable, flexible service for the asynchronous exchange of critical business data and events throughout an enterprise.

JNDI

Java Naming and Directory Interface (JNDI) consists of a standard set of **APIs** that provide directory and naming services. Oracle Application Server has a JNDI naming server that **clients** can use to obtain **object references** to **ECO/Java objects** or Enterprise JavaBean objects.

JServ process

JServ process is an open source **servlet** engine that can communicate with Oracle HTTP Server by configuring it to accept a **servlet** request from `mod_jserv`, which is an Oracle HTTP Server plug-in.

JSP

JavaServer Pages (JSPs) are an extension to the **servlet** functionality that enables a simple programmatic interface to Web pages. JSPs are **HTML** pages with special tags and embedded **Java** code that is executed on the Web or application server, providing dynamic functionality to HTML pages. JSPs are actually compiled into **servlets** when first requested and run in the **servlet container**.

JSP engine

See **JSP translator**.

JSP tag

JSP tags are tags that are used in JavaServer Pages (JSPs). These tags use the `<jsp:` syntax and enclose action elements in the JSP with "begin" and "end" tags similar to XML statements.

JSP translator

A JSP translator is an entity that translates, executes, and processes JSP pages and delivers requests to them. The exact architecture of a JSP translator varies from implementation to implementation, but it consists of a **servlet** or a collection of servlets. The JSP translator, therefore, is executed by a **servlet container**.

JTA

Java Transaction API (JTA) enables your applications to participate in distributed transactions and to access transaction services from other components.

JTS

Java Transaction Service (JTS) provides the services necessary for applications and databases to become part of a **transaction**. JTS is the **Java** version of **OTS**.

just-in-time compilation

See **JIT compilation**.

JVM

A Java Virtual Machine (JVM) is an abstract specification for a computing device that can be implemented in different ways, in software or hardware. You compile to the instruction set of a **virtual machine** much like you would compile to the instruction set of a microprocessor.

The Java Virtual Machine is part of the **Java** runtime environment responsible for interpreting Java bytecode. It consists of a bytecode instruction set, a set of registers, a stack, a garbage-collected heap, and an area for storing methods. Java bytecode is executable by any JVM running on any machine.

key pair

A key pair is a pair of mathematically related keys (a **public key** and a **private key**) associated with a user and used in **public-key encryption**.

language identifier

A language identifier is a two-character alphanumeric string that identifies a human language, as defined by **RFC 1766**.

LDAP

Lightweight Directory Access Protocol (LDAP) is a protocol that allows **clients** to access information from a **directory server**. This protocol enables corporate directory entries to be arranged in a hierarchical structure that reflects geographic and organizational boundaries.

light weight directory access protocol

See LDAP.

listener

A listener is an **HTTP server** that handles incoming requests and routes them to the dispatcher.

Login Module

A Login Module is the **authentication** module configured for a particular application. The `LoginContext` class decouples the application code from the authentication services, and different login modules can be plugged in under an application without affecting the application code.

Login Server

The Login Server authenticates the username and password of a **client** user attempting to access an application. Once authenticated, the Login Server passes the client's identity to various applications. An encrypted login **cookie** identifies the client as being authenticated. The Login Server provides single sign-on (**SSO authentication**). This enables a user to access multiple accounts and applications with a single username and password.

manifest file

A manifest file is a text file that describes the contents of a JAR file. For **EJB JAR** files, the manifest file identifies it as an "ejb-jar" file. The manifest file also indicates which elements in the JAR file are the **EJB deployment descriptors** for the components to be deployed.

message

A message is the smallest unit of information inserted into and retrieved from a queue. A message consists of the following:

- Control information (metadata)
- Payload (data)

The control information represents message properties used by the queue to manage messages. The payload data is the information stored in the queue. A message can reside in only one queue.

message-driven beans

Message-driven beans (MDB) provide an easier method for implementing asynchronous communication than using straight JMS. MDBs were created to receive asynchronous JMS **messages**. The **container** handles much of the setup required for JMS queues and topics. It sends all messages to the interested MDB.

metric

A metric is a performance statistic, such as uptime or queue.

MIME type

The Multipurpose Internet Mail Extension (MIME) type is a **message** format used on the Internet to describe the contents of a message and defined by the MIME standard. MIME is used by **HTTP servers** to describe the type of content being delivered. Several **RFCs** define MIME.

See Also:

<http://www.oac.uci.edu/indiv/ehood/MIME/MIME.html>

mods

See **Apache module**.

mod_access

The `mod_access` **module**, or plug-in, of Oracle HTTP Server provides access control based on **client** host name or **IP address**.

mod_osso

The `mod_osso` **module** provides communication between the **SSO-enabled Login Server** and the Oracle HTTP Server **listener**.

module

See **Apache module**.

multiport

A multiport is a single **listener** that can respond to requests directed at more than one address/**port** combination. A multiport can be configured to respond to

requests differently if they are made to a different address/port combination on the same listener.

Multipurpose Internet Mail Extensions

See MIME type.

national language support

See NLS.

NLS

National language support (NLS) is the set of mechanisms used to translate data between various languages and **character sets**.

node

See host.

noun

In the scope of performance monitoring tools, a noun is a component of an Oracle Application Server site. Examples of a noun are component, **object**, class, bean, process, piece of code, computer, and process.

OAGI

The Open Applications Group, Inc. (OAGI) is a nonprofit consortium focusing on the best practices and processes that are based on XML content for e-business and application integration. OAGI was set up to create common standards for the integration of enterprise business applications. Member companies are building specifications to standardize integration between enterprise business applications. The OAGI Business Object Document (BOD) is the architecture that is used to communication message or business documents between software applications or components. Each BOD includes supporting details to enable the destination business application to accomplish the action.

object

An object is any item that can be individually selected or manipulated. This includes buttons and pull-down menus. In **object-oriented programming**, an object is a self-contained entity that consists of both data and procedures to manipulate the data.

Object Management Group

See OMG.

object-oriented programming

Object-oriented programming is a method of programming that organizes a program around its data (**objects**) and a set of well-defined interfaces to that data. It is a revolutionary new way of looking at computer programming. A generalization of the data object along with its possible data variables and methods (what to do with variables) in a class of data objects.

object reference

An object reference is a unique identifier that is used to represent an **object** instance in a distributed system.

See Also: IOR

Object Transaction Service

See OTS.

OC4J

See **Oracle Application Server Containers for J2EE (OC4J)**.

OCI

Oracle Call Interface (OCI) is an **API** or low-level tool for accessing Oracle databases and executing **SQL** and **PL/SQL** statements.

OLTP

Online **transaction** processing (OLTP) refers to the immediate processing of a transaction. The opposite of transaction processing is batch processing, where transactions are stored (batched) and then all of the stored transactions are executed at one time.

OMG

Object Management Group (OMG) is a consortium with a membership of more than 700 companies. The organization's goal is to provide a common framework for developing applications using **object-oriented programming** techniques. OMG is responsible for the **CORBA** specification.

online transaction processing

See OLTP.

Open Applications Group, Inc.

See OAGI.

ORACLE_HOME

ORACLE_HOME is an environment variable (UNIX) or registry key (Windows) that indicates the directory where Oracle software is installed.

Oracle Business Components for Java (BC4J)

Oracle Business Components for Java (BC4J) is a 100% **Java**-compatible, **XML**-powered framework that enables productive development, portable deployment, and flexible customizations of multi-tier database applications from business components.

Oracle platform

The Oracle platform is a **server** deployment platform consisting of Oracle Application Server, Oracle Database Server, and Oracle Developer Suite.

Oracle Application Server Active Failover Cluster

Oracle Application Server Active Failover Clusters are a high availability solution where the Oracle Application Server Infrastructure can be deployed on a hardware cluster comprised of multiple nodes, such that each node of the cluster is able to simultaneously process incoming requests. The Infrastructure is installed on each of the nodes, but the Infrastructure database is installed on a shared disk, accessible by all nodes.

Oracle Application Server Cold Failover Cluster

Oracle Application Server Cold Failover Clusters are a high availability solution where the Oracle Application Server Infrastructure is typically deployed on a two-node hardware cluster with a shared storage device. Once node is active or "hot," meaning it is running the Infrastructure, while the other node is "cold" and is not running the Infrastructure. When the active node fails, the clusterware switches Infrastructure operations to the previously "cold" node and the Infrastructure is started on that node.

Oracle Application Server Containers for J2EE (OC4J)

Oracle Application Server Containers for J2EE (OC4J) is a complete set of **J2EE containers** written entirely in **Java** that execute on the Java Virtual Machine (**JVM**) of the standard Java Development Kit (JDK).

Oracle Application Server Discoverer

Oracle Application Server Discoverer is a **business intelligence** tool for analyzing data. Using Oracle Application Server Discoverer's award-winning user interface, users can access and analyze database data. There are two Oracle Application Server Discoverer products:

- Oracle Application Server Discoverer Plus is the Internet version of the award-winning Windows version of Discoverer. With Discoverer Plus, business professionals can get and analyze data in a company's database without having to understand complex database concepts. Using Wizard dialogs and menus, Discoverer Plus guides users through the steps to get and analyze data to support their business decisions.
- Oracle Application Server Discoverer Viewer is a tool for viewing workbooks created by Discoverer Plus. Oracle Application Server Discoverer Viewer can also be used to integrate database output into a Web site and **portal**. It is easy to customize Discoverer Viewer both to conform to a particular Web site look and feel and to build custom Discoverer applications for the Web. Discoverer Viewer is optimized for performance and designed to minimize network traffic.

Oracle Application Server Forms Services

Oracle Application Server Forms Services deploys Forms applications with database access to **Java clients** in a Web environment. Oracle Application Server Forms Services automatically optimizes class downloads, network traffic, and interactions with Oracle database. Applications are automatically load-balanced across multiple servers and, therefore, can easily scale to service any number of requests.

Oracle Application Server Infrastructure

Oracle Application Server Infrastructure is a single, consistent combination of the **Oracle Application Server Metadata Repository**, a **directory server**, management server, and **single sign-on** server. Most Oracle Application Server components use the infrastructure to provide a more integrated environment for application developers and system managers.

Oracle Application Server InterConnect

Oracle Application Server InterConnect is the integration hub that coordinates the communication and transformation of **messages** between two or more heterogeneous applications. Oracle Application Server InterConnect defines business events, their associated data, and any transformations required to map one application's view of a business **object** to another's view.

Oracle Application Server InterConnect Adapters

Oracle Application Server InterConnect Adapters are a set of connectivity adapters that enable third-party applications and technology environments to participate in integration.

Oracle Application Server Metadata Repository

Oracle Application Server Metadata Repository is a pre-seeded database containing metadata required by Oracle Application Server **instances**.

Oracle Application Server Portal

Oracle Application Server Portal is a complete solution for building, deploying and monitoring Web database applications and content-driven Web sites. Oracle Application Server Portal enables you to create and view database **objects** through an easy-to-use **HTML**-based interface, and provides tools for creating **HTML**-based interfaces. It also allows you to resolve performance problems using performance tracking facilities, and enables you to manage database security through its interface.

See Also: portal, portlet

Oracle Application Server Portal Login Server

See Login Server.

Oracle Application Server Reports Services

Oracle Application Server Reports Services provides an easy-to-use, scalable, and manageable solution for high-quality database publishing and reporting by creating dynamic reports for the Web and across the enterprise. It enables you to implement a multi-tiered architecture for running your reports.

Oracle Application Server Web Cache

Oracle Application Server Web Cache is a **server** accelerator **caching** service that improves the performance, **scalability**, and **availability** of frequently used **e-business** Web sites that run on the **Oracle platform**. By storing frequently accessed **URLs** in virtual memory, Oracle Application Server Web Cache eliminates the need to repeatedly process requests for those **URLs** on the Web server, and it caches both static and dynamically-generated **HTTP** content from one or more applications Web servers.

Oracle Application Server Wireless

Oracle Application Server Wireless is a **portal** service for delivering information and applications to mobile devices. Using Oracle Application Server Wireless, you can create custom portal sites that use different kinds of content, including Web pages, custom **Java** applications, and **XML**-based applications. Oracle Application Server Wireless sites make this diverse information accessible to mobile devices without you having to rewrite the content for each target device platform.

Oracle Advanced Queuing

Oracle Advanced Queuing provides database-resident messaging capabilities that enable reliable, asynchronous communication between applications. Oracle Application Server integration uses Oracle Advanced Queuing as the communication mechanism between adapters, participating applications, and **Oracle Workflow**.

Oracle Advanced Security

Oracle Advanced Security provides a comprehensive suite of security features to protect enterprise networks and securely extend corporate networks to the Internet. It provides a single source of integration with network **encryption** and **authentication** solutions, **single sign-on** services, and security protocols. By integrating industry standards, it delivers unparalleled security to the Oracle network and beyond.

Oracle Call Interface

See **OCI**.

Oracle Database Client Developer Kit

The Oracle Database Client Developer Kit contains the following **client** libraries:

- Oracle Java Database Connectivity (**JDBC**) Drivers
- Oracle Java Messaging Service (**JMS**) Toolkit
- Oracle SQLJ Translator

Oracle HTTP Server

Oracle HTTP Server is the Web **server** that Oracle Application Server uses, which is built on **Apache** Web server technology. Oracle HTTP Server offers **scalability**, stability, speed, and extensibility. It also supports Java **servlets**, JavaServer Pages (**JSPs**), Perl, **PL/SQL**, and **CGI** applications.

Oracle Internet Directory

Oracle Internet Directory is a general purpose directory service that enables retrieval of information about dispersed users and network resources. Oracle Internet Directory combines **LDAP** version 3 with the high performance, **scalability**, robustness, and **availability** of the Oracle database.

Oracle Internet Directory runs as an application in the Oracle database. It communicates with the database, which may be on the same or a different operating system.

Oracle Internet File System

Oracle Internet File System is a file system that stores its contents in an Oracle database. The contents can range from files to metadata that defines the file system (including foldering, security, and users). Oracle Internet File System indexes textual content in files as they are added or updated. This enables you to search the contents of more than 150 file formats.

Oracle LDAP Developer Kit

Oracle LDAP Developer Kit supports **client** interaction with any **LDAP**-compliant **directory server**; for example, **Oracle Internet Directory**. **LDAP** (Lightweight Directory Access Protocol) is the emerging Internet standard for directory services. The toolkit provides tools and development libraries to support client calls to directory services, encrypted connections, and enables you to manage your directory data.

Oracle Wallet Manager

Oracle Wallet Manager is a **Java**-based application that security administrators use to manage public-key security credentials on **clients** and **servers**. Security credentials consist of a public/private **key pair**, a **certificate**, and a **trustpoint**.

See Also: **public-key cryptography, wallet**

Oracle Workflow

Oracle Workflow lets you automate and continuously improve business processes, **routing** information of any type according to easily-changeable business rules to users both inside and outside a company's enterprise.

Oracle XML Developer Kit

Oracle XML Developer Kit (XDK) contains the necessary **XML** component libraries and utilities to give developers the ability to easily XML-enable applications and

Web sites. Oracle XDK supports development in **Java**, C, C++, and **PL/SQL** with a collection of libraries, command-line utilities, and tools.

ORB

An Object Request Broker (ORB), in **CORBA**, acts as a "broker" between a **client** request for a service from a distributed **object** or component and completion of that request. Having ORB support in a network means that a client program can request a service without having to understand where the **server** is in the distributed network or in which language the server object is written.

OTS

Object Transaction Service (OTS) provides the services necessary for applications and databases to become part of a **transaction**.

See Also: JTS

overloading

Overloading is the ability to use the same name for more than one variable or procedure, requiring the compiler to differentiate them based on context. Overloaded procedures and functions (in **PL/SQL**) or methods (in **Java**) have the same name but take different parameters and do similar but not identical things.

package

A package is a group of **PL/SQL** or **Java** functions and procedures.

partner interface processes

See PIP.

PIP

Partner interface processes (PIPs) are specialized XML-based dialogs that determine how two trading partners must interact to conduct a business collaboration. Some examples are: a buyer submitting a purchase order to a seller, a buyer notifying a seller of a change in a purchase order, or a customer submitting a forecast to a supplier. PIPs are grouped into clusters and these clusters are further grouped into segments. These segments are divided into business functions. The PEP name is coded to reflect the cluster, segment, and business functions of the PIP.

PKCS

The Public-Key Cryptography Standards (PKCS) are specifications produced by RSA Laboratories in cooperation with secure systems developers worldwide for the purpose of accelerating the development of **public-key cryptography**.

PL/SQL

PL/SQL is Oracle's proprietary extension to the **SQL** language. PL/SQL adds procedural and other constructs to SQL that make it suitable for writing applications.

policy

A policy is an **authorization** rules repository for **JAAS**. A policy contains the rules with which a user is authorized to use resources, such as a database. The primary function of a policy implementation is to provide the following information:

Given a grantee, return the granted permissions for the grantee.

policy file

A policy file grants permissions to the trusted codes or applications that you run. These codes or applications are not allowed to access resources unless they are explicitly granted permission to do so by the security **policy** in effect. In **Java** environments, the permission must be granted by an entry in a policy file.

policy store

A policy store provides secure, centralized storage, retrieval, and administration of **JAAS** policies (the **roles** and **privileges** of a user). Oracle Internet Directory (OID) is the policy store used for storing this information.

polymorphism

In **object-oriented programming**, polymorphism (from the greek meaning "having multiple forms") is the characteristic of being able to assign a different meaning to a particular symbol or "operator" in different contexts.

port

A port is a number that TCP uses to route transmitted data to and from a particular program.

portal

Portal sites give users a single, centralized, personalized view of relevant applications and data.

See Also: Oracle Application Server Portal, portlet

portlet

A portlet is a reusable information component that summarizes or provides access to an information source. Portlets are the fundament building blocks of an **Oracle Application Server Portal** page.

See Also: portal

principal

A principal is a specific authenticated identity, such as a user named `frank`. A principal is essentially an identity associated with a **subject**. A principal can be associated with a subject (such as a user) upon successful **authentication** to a computing service.

private key

A private key is a key used by a limited number of communicating parties to decrypt data encrypted with a **public key**.

See Also: public-key encryption

privilege

A privilege is the right to perform an action on the database. Privileges can either be general (system privileges) or specific to particular database **objects** (object privileges). They can also be grouped into **roles**.

See Also: role

processor sets

Processor sets, which are specific to Solaris systems and were introduced in Solaris 2.6, allow a group of processors to be allocated for the exclusive use of one or more applications. This gives a system administrator control over the creation, management, and binding of processes into processor sets.

protection domain

A protection domain is a fundamental building block of system security. The protection domain concept serves as a convenient mechanism for grouping and isolation between units of protection. For example, it is possible to separate protection domains from interacting with each other so that any permitted

interaction must be either through trusted system code or explicitly allowed by the domains concerned.

proxy server

A proxy server typically sits on a network **firewall** and allows **clients** behind the firewall to access Web resources. All requests from clients go to the proxy server rather than directly to the destination **server**. The proxy server forwards the request to the destination server and passes the received information back to the client. The proxy server channels all Web traffic at a site through a single, secure **port**; this allows an organization to create a secure firewall by preventing Internet access to internal machines, while allowing Web access.

public key

A public key is a key known to all users, used to encrypt data in such a way that only a specific user can decrypt it.

See Also: **private key, public-key encryption**

public-key cryptography

Public-key cryptography is a set of well-established techniques and standards that ensure that data passes securely over a network, from the sender to the receiver, without any interference from a third party. Public-key cryptography facilitates **encryption** and decryption, tamper detection, **authentication**, and nonrepudiation.

Public-Key Cryptography Standards

See PKCS.

public-key encryption

Public-key encryption is a form of **encryption** that uses a **key pair** (a **public key** and a **private key**) to encrypt and decrypt data.

query string

A query string is the optional portion of a **URL** that specifies parameters to be passed to an application.

RAC

Real Application Clusters (RAC) is a parallel database clustering technology from Oracle. RAC is an active-active cluster with shared storage, whereby multiple servers can work in parallel on the same set of data.

Real Application Clusters

See RAC.

realm

A realm provides access to a **policy store** of users and **roles** (groups) and optionally provides administrative functionality. Oracle provides a proprietary Realm API that implements the notion of a user community. Essentially a user community instance is a realm that is maintained internally by the **authorization** system.

RC4, RC5

RC4 and RC5 are **RSA encryption** algorithms.

redundant

Redundant components are duplicate or extra computing components that protect a computing system.

reliability

Reliability is the ability of a computing system to operate without failing. Reliability is measured by mean-time-between-failures (MTBF).

remote method invocation

See RMI.

remote method invocation

See RPC.

request latency

Request latency is the time required to process a request.

request throughput

Request throughput is the number of requests processed per unit of time.

response time

Response time is the amount of time between the submission of a request and the receipt of the response.

restriction

Restriction is a **security scheme** that restricts access to files provided by the **server** to **client** machines within certain groups of **IP addresses** or **DNS domains**.

See Also: domain-based restriction, IP-based restriction

reverse mapper

A reverse mapper is a utility that converts a **CORBA** server **object**'s interfaces to **IDL** (for example, **Java** to **IDL**). These **IDL** interfaces can then be converted to **CORBA** stub and skeleton code by an **IDL** compiler (for example, **IDL** to **Java**) for use by **clients**.

RFC

Request for Comments (**RFC**) is a collection of Internet-related articles.

See Also: <http://www.rfc-editor.org>

RMI

Remote method invocation (**RMI**) is an interaction scheme for distributed **objects** written in **Java**. It enables a **Java** program running on one computer to access the methods of another **Java** program running on another computer.

RNIF

The RosettaNet Implementation Framework (**RNIF**) defines implementation guidelines for creating software applications that provide for the secure and reliable transport of **PIPs** in **XML**-formatted business documents between trading partners.

role

A role is a group of database **privileges** that can be granted and revoked as a unit. Specific privileges can be granted and revoked from the role dynamically, and the role can be enabled or disabled dynamically for specific users.

role activation

Role activation is the process by which a user can selectively enable the required **roles** to accomplish a specific task in a user session. A user is granted multiple roles, but not all roles are enabled by default. Enabling only those roles the user needs, the user ensures the principle of least **privilege**. That is, the user is not enabling permissions or privileges unnecessary for the task. This limits the damage that can result from an accident or error.

role-based access control

Role-based access control is a process that simplifies the administrative process of **authorization** management based on direct assignment of permissions to users.

Instead of directly assigning permissions to users, permissions are assigned to **roles**, and users are granted their permissions by being made members of appropriate roles. Multiple roles can be granted to a user, and a role can also be granted other roles, thus forming a role hierarchy that provides security administrators with a tool to model enterprise security policies.

RosettaNet

RosettaNet is a business-to-business (**B2B**) protocol standard with the following components: **PIPs**, **RNIF**, and **RosettaNet Dictionaries**.

RosettaNet Dictionaries

RosettaNet dictionaries define and describe a common set of properties for use in **PIP** messages. These properties define a common vocabulary for conduction business so that the information that is exchanged between trading partners is consistent.

RosettaNet Implementation Framework

See **RNIF**.

routing

Routing is the process of directing data from one machine on the Internet to another by way of intermediate machines.

RPC

Remote procedure call (RPC) is a type of protocol that allows a program on one computer to execute a program on a **server**. Using RPC, a system developer need not develop specific procedures for the server. The client program sends a message to the server with appropriate arguments and the server returns a message containing the results of the program executed.

RSA

RSA is an Oracle partner supplying **encryption** algorithms for the **HTTP server** and Web Request Broker.

scalability

Scalability is the ability to handle increasing numbers of hardware requests without adversely affecting latency and throughput.

schema

A schema is a collection of database **objects**, including logical structures such as tables, views, sequences, **stored procedures**, synonyms, indexes, **clusters**, and database links.

A schema has the name of the user who controls it.

secure sockets layer

See **SSL**.

security scheme

A security scheme prevents unauthorized users from accessing protected files or applications. The **Oracle platform** supports both **authentication** schemes and **restriction** schemes.

The Oracle platform supports the following types of **authentication**:

- **basic authentication**
- **digest authentication**
- **certificate authentication**

The following **restriction** schemes are supported:

- **IP-based restriction**
- **domain-based restriction**

serializable

An interface that a **Java** class implements to enable itself to be persistent through the Java Serialization service. This service allows a bean's state to be stored and retrieved.

server

There are two types of servers relevant to this product. The first is Oracle9i Database Server, which is a relational database server dedicated to performing data management duties on behalf of **clients** using any number of possible interfaces. The other is Oracle Application Server, which is a collection of middleware services and tools that provide a scalable, robust, secure, and extensible platform for distributed, **object-oriented** applications. Oracle Application Server supports access to applications from both Web clients (browsers) using **HTTP** and Common Object Request Broker Architecture (**CORBA**) clients, which use the CORBA and the Internet Inter-ORB (**IIOP**) protocols.

service time

Service time is the time between the receipt of a request and the completion of the response to the request.

servlet

A servlet is a small **Java** program that runs on a **Web server**, as opposed to an **applet**, which is a Java program that runs in a **client** browser. Servlets take client **HTTP** requests from a browser, generate dynamic content (such as through querying a database), and provide an HTTP response back to the browser.

servlet container

A servlet container is the runtime environment for a Java **servlet**. The servlet container calls the methods of a servlet and provides services to the servlet while it is executing.

servlet zones

Servlet zones are analogous to virtual directories in a **Web server**. Each zone can have its own set of **servlets** and can use a dedicated Java Virtual Machine (**JVM**). All of the servlets in one servlet zone share the same servlet repository, which is a location where **Java** class files are loaded.

session bean

A session bean implements one or more business tasks. A session bean might contain methods that query and update data in a relational table. Session beans are often used to implement services. For example, an application developer might implement one or several session beans that retrieve and update inventory data in a database.

session key

A session key is a secret key used by **SSL** to encrypt data transmitted over a secure connection. The **client** generates the session key after Oracle Application Server authenticates itself and sends it to Oracle Application Server using **public-key encryption**.

shadow process

A shadow process is a simple process on every **host** that detects if an Oracle HTTP Server **listener** process on that host is alive. If the listener process dies, then the shadow process brings a new instance up to handle requests.

shared secret-key encryption

Shared secret-key encryption is a form of **encryption** that uses a single key both to encrypt and to decrypt a document. Shared secret-key encryption is much faster than **public-key encryption**, but is more vulnerable to attack.

short message service

See SMS.

Simple Network Management Protocol

See SNMP.

Simple Object Access Protocol

See SOAP.

single sign-on

See SSO.

skeletons

In **CORBA**, skeletons are classes that are generated as part of the **IDL** compilation process. They receive calls from **client** stubs, unmarshal parameters, and call the method implementation.

See Also: **IDL, skeletons**

SMS

Short message service (SMS) is the transmission of short text messages (under 160 alpha-numeric characters and no images) to a mobile device.

SNMP

Simple Network Management Protocol (SNMP) is a set of protocols for managing complex networks. The versions of SNMP were developed in the early 1980s. SNMP works by sending **messages**, called protocol data units (PDUs), to different parts of a network. SNMP-compliant devices, called agents, store data about themselves in Management Information Bases (MIBs) and return this data to the SNMP requesters.

SOAP

Simple Object Access Protocol (SOAP) is a lightweight, **XML**-based protocol for exchanging information in a decentralized, distributed environment. SOAP

supports different styles of information exchange, including Remote Procedure Call style (RPC) and Message-oriented exchange. RPC style information exchange allows for request-response processing, where an endpoint receives a procedure-oriented **message** and replies with a correlated response message. Message-oriented information exchange supports organizations and applications that need to exchange business or other types of documents where a message is sent but the sender may not expect or wait for an immediate response.

SQL

Structured query language (SQL) is the internationally accepted standard for relational systems, covering not only query but also data definition, manipulation, security, and some aspects of referential integrity.

See Also: PL/SQL

SSL

Secure Sockets Layer (SSL) is a standard for the secure transmission of documents over the Internet using **HTTPS (secure HTTP)**. SSL uses **digital signatures** to ensure that transmitted data is not tampered with.

SSO

Single Sign-on (SSO) is the ability for a user to authenticate once, combined with strong **authentication** occurring transparently in subsequent connections to other databases or applications. SSO lets a user access multiple accounts and applications with a single password.

stateful application

A stateful application is an application that maintains its own state information.

stateful session bean

A stateful session bean is a bean in which instance variables represent the state of a unique **client**-bean session. Because the client interacts with its bean, this state is often called the conversational state. This conversation state is retained for the duration of the client-bean session. If the client removes the bean or terminates, the session ends and the state disappears.

stateless application

A stateless application is an application that does not store state information. Requests to a stateless application must include all information that the application needs to run properly.

stateless session bean

A stateless session bean is a bean which does not maintain a conversational state for a particular **client**. When a client invokes the method of a stateless bean, the bean instance variables may contain a state, but only for the duration of the invocation. When the method is finished, the state is no longer retained. Except during method invocation, all instances of a stateless bean are equivalent, allowing the **EJB container** to assign an instance to any client. Because stateless session beans can support multiple clients, they can offer better **scalability** for applications that require large numbers of clients.

stored procedure

A stored procedure is a set of **PL/SQL** instructions that are stored in a database.

structured query language

See **SQL**.

stub

In **CORBA**, a stub is a class that is generated as part of the **IDL** compilation process (e.g. using an **IDL-to-Java** compiler). The compiler creates the stub from the **IDL** interfaces of a server **object**. The stub contains code that handles operations needed to access remote objects. A **client** application uses the generated stubs to access methods in the remote server object.

See Also: **IDL** and **skeletons**

subagent

A subagent is a process that receives queries for a particular managed element from the master agent and sends back the appropriate answers to the master agent. The **SNMP** subagent detects unexpected events and forwards them to the master agent.

subject

A subject refers to any user of a computing service. Since users, **roles**, and computing services use computing services, all are subjects. A subject is the source of a service request. To identify the subjects with which it interacts, a computing service typically relies on names. Subjects can have different names for each

individual service with which they interact. These individual names are known as **principals**. For example, a user can have a name principal (user `frank`), a role principal (role `hr`), and a social security number principal (`123-45-6789`).

subscriber

A subscriber is customer (such as a company) that uses (subscribes to) an application deployed in a **hosted environment**.

subscription

In the scope of the performance monitoring tools, a subscription is a request by a user to track a particular **metric**.

SWIFT

SWIFT is the industry-owned cooperative that supplies a standard for secure messaging services and interface software to financial institutions throughout the world. SWIFT focuses on messaging services to banks, brokers or dealers, investment managers, as well as to market infrastructures in payments, treasury, securities, and trade.

three-tier architecture

Three-tier architecture is a **client-server** computing architecture where systems are separated into three layers: the interface layer, the application logic layer, and the database layer. Clients in this model are thin clients, which do not contain as many resources as clients in the **two-tier architecture**. The application logic layer, which is also known as the middle-tier, provides the computing power and resources for the client. Oracle Application Server fulfills this role ideally as a development and deployment platform for database-enabled applications.

throughput

Throughput is the number of requests processed per unit of time.

TNS string

A Transparent Network Substrate (TNS) string is a foundation technology built into Net8, Oracle Connection Manager and Oracle Names, that works with any standard network transport protocol. The TNS string contains destination service and network route information. Users initiate a connect request by passing a user name and password along with a connect identifier in a TNS String for the service to which they want to connect.

topology management request

A topology management request is an **HTTP** request that is handled by the Oracle HTTP Server **module**, `mod_oprocmgr`. It is used for **server** process management and administration.

transaction

A transaction is a collection of operations that work toward a goal of satisfying a single user's need. A transaction guarantees that all operations within it either succeed or fail.

As an example, a transfer from a savings account to a checking account has a transaction that debits the savings account and credits the checking account. If both operations succeed, the transaction is committed, otherwise the transaction is rolled back.

transaction processing

See **OLTP**.

transaction service

A transaction service is an **OTS** service that enables you to perform **transactions** that span several **HTTP** requests. The transaction service is based on the XA open model transactions defined by the X/Open Company.

transactional DAD

A transactional DAD is a database that has been configured to be involved within a **transaction**. Once configured, more than one database can be managed by Oracle Application Server in committing resources on all databases involved in the transaction. A database cannot be involved in a transaction unless the **DAD** is configured to be transactional.

trustpoint

Also referred to as a trusted certificate authority (**CA**), a trustpoint is an issuer of **certificates** whom you can trust. You can use **Oracle Wallet Manager** to manage trustpoint information associated with your application.

two-tier architecture

Two-tier architecture is the traditional **client-server** computing architecture with application front-end and logic executing on a client that accesses a database.

UDDI

Universal Description, Discover, and Integration (UDDI) is an online electronic registry that serves as electronic Yellow Pages, providing an information structure where various business entities register themselves and the services they offer through their **WSDL** definitions. There are two types of UDDI registries, public UDDI registries that serve as aggregation points for a variety of businesses to publish their services, and private UDDI registries that serve a similar role within organizations.

UNICODE

UNICODE is a 16-bit character **encoding** scheme supporting a large subset of characters found in both Western and Asian languages.

uniform resource identifier

See **URI**.

uniform resource locator

See **URL**.

Universal Description, Discover, and Integration

See **UDDI**.

URI

Uniform Resource Identifier (URI) is the generic term for all types of names and addresses that refer to **objects** on the World Wide Web. A **URL** is one kind of URI.

URL

Uniform Resource Locator (URL), a form of **URI**, is a compact string representation of the location for a resource that is available via the Internet. It is also the text-string format Web **clients** use to encode requests to Oracle Application Server.

virtual host

See **multiport**.

virtual machine

See **JVM**.

virtual pathname

In a virtual file system, a virtual pathname is a synonym that the virtual file system maps to a file stored in the file system maintained by the **host** machine's operating system.

VM

See **JVM**.

wait time

Wait time is the time between the submission of a request and initiation of the request.

wallet

A wallet is an abstraction used to store and manage security credentials for an individual entity. It implements the storage and retrieval of credentials for use with various cryptographic services.

WAR files

Web Archive (WAR) files contain the **Web component modules** of **J2EE** applications. A WAR file contains its **deployment descriptor** (DD), related files, and the `.class` files for the **servlet** or the `.jsp` files for a **JSP** page.

watcher

A watcher is a thread in an Oracle HTTP Server parent process. It is spawned by the parent `init` process of `mod_oprocmgr`. Its purpose is to periodically check the status of each process running on the **server**. If a process is dead or if it fails to start up, then the watcher thread will issue an **HTTP** request to start a new process.

Web component

J2EE Web components can be either JavaServer Pages (**JSPs**) or **servlets**. Static **HTML** pages, **applets**, or **server-side** utilities can be bundled with Web components during application assembly, but are not considered Web components by the **J2EE** specification.

Web services

Web services are Web applications that are transmitted over the Internet using the **XML**, **SOAP**, **WSDL**, and **UDDI** open standards. **XML** is used to tag the data, **SOAP** is used to transfer the data, **WSDL** is used for describing the available services, and **UDDI** is used for listing the available services.

Web Services Description Language

See **WSDL**.

wireless portal

A wireless portal is accessible from wireless devices such as cellular telephones.

See Also: Oracle Application Server Wireless, portal

WSDL

Web Services Description Language (WSDL) is an **XML** format for describing network services containing **RPC-oriented** and **message-oriented** information. Programmers or automated development tools can create WSDL files to describe a service and can make the description available over the Internet. Client-side programmers and development tools can use published WSDL descriptions to obtain information about available Web Services and to build and create proxies or program templates that access available services.

X.500

X.500 is a standard for developing an electronic directory of users in an organization so that it can be part of a global directory available to anyone in the world with Internet access. Such a directory is sometimes called a global White Pages directory. The idea is to be able to look up people in a user-friendly way by name, department, or organization.

XDK

See **Oracle XML Developer Kit**.

XML

eXtensible Markup Language (XML) is a set of rules for defining data markup in a plain text format.

XSL

eXtensible Stylesheet Language (XSL) is used within style sheets to transform or render XML documents.

XSLT

The eXtensible Stylesheet Language Transformation (XSLT) processor transforms XML into other text-based formats, such as **HTML** and **Wireless Markup Language (WML)**.

XSQL servlet

The XSQL servlet combines XML, SQL, and XSLT in the server to deliver dynamic Web content.

Index

A

- access control
 - portals, 10-11
- Active Failover Clusters, 7-13
- Apache HTTP Server, 2-3
- application development
 - deployment, 2-33
 - mapping, 2-31
 - overview, 2-31
 - packaging, 2-33
 - performance tuning, 2-33
 - querying, 2-32
 - session management, 2-32
 - transactions, 2-33
- application level caching, 8-12
- Application Server Control, 9-1
 - architecture, 9-3
 - features, 9-2
 - monitoring, 9-3
 - supporting technologies, 9-4
- applications
 - debugging, 2-36
 - deploying, 2-1
 - developing, 2-1, 2-36
 - distributed, 2-10
 - FastCGI, 2-4
 - forms, 2-45
 - J2EE clients, 2-12
 - J2EE packaging, 2-13
 - packaging, J2EE, 2-13
 - portals, 3-1, 3-7, 3-9
 - types, 2-2

architecture

- OC4J, 2-19
- Oracle Content Management SDK, 2-54
- Oracle HTTP Server, 2-6
- Oracle HTTP Server process, 2-7
- OracleAS Forms Services, 2-48
- OracleAS MapViewer, 2-56
- OracleAS Web Cache, 8-10
- OracleAS Wireless, 4-8
 - portals, 3-12
 - Web services, 2-42
 - wireless, 4-8
- assertion services, 11-7
- authentication, 11-7, 10-11
- automatic content compression, 8-6
- availability, 1-5
 - definition, 7-4
 - introduction, 1-4

B

- back-up, 7-14
- BC4J, 2-37
 - customizing applications, 2-39
 - deploying applications, 2-38
 - integration with JDeveloper, 2-39
 - integration with Oracle InterMedia, 2-40
 - object-relational mapping, 2-37, 2-38
 - performance, 2-38
 - scalability, 2-38
 - user interface support, 2-39
 - validation model, 2-38
- business challenges, 3-2
- business intelligence, 1-8, 5-1

- Business Intelligence Beans, 2-37
- business logic
 - Java, 2-20
- business process automation, 6-2
- business-to-business integration, 6-3

C

- C++
 - Oracle HTTP Server, 2-4
 - XML, 2-51
- cache clustering, 8-8
- cache hierarchies, 8-11
- caching, 1-5, 1-8
 - application level, 8-12
 - dynamic, 8-7
 - full page, 8-7
 - invalidation, 8-7
 - Java, 2-19
 - Java object cache, 8-12
 - overview, 2-27, 8-5
 - partial-page, 8-7
 - portals, 3-11
 - rules for browser types, 8-9
 - server accelerators, 8-5
 - solutions, 8-4
 - static, 8-7
 - Web object cache, 8-12
- categories, 3-5
- CGI
 - Oracle HTTP Server, 2-8
 - PL/SQL, comparison, 2-52
- challenges
 - deployment, 1-3
 - development, 1-1
- cloning, 7-5
- clustering
 - benefits, 7-9
 - cache clusters, 8-8
 - Oracle Enterprise Manager, 7-9

- clusters
 - active failover, 7-13
 - cold failover, 7-12
 - definition, 7-8
 - manually configured, 7-10
 - Web Cache clusters, 7-1
- CodeCoach, 2-36
- Cold Failover Clusters, 7-12
- commit
 - two-phase, 2-17
- common configurations
 - See* recommended topologies, 11-1
- compression, 8-6
- connectivity, remote, 2-16
- container managed transactions, 2-16
- containers
 - JSP, 2-14
 - OC4J, 2-13
 - OC4J EJB, 2-15
 - OC4J JSP, 2-14
- content
 - delivery, 3-6
 - personalization, 3-6
 - sharing, 3-3
 - subscribers, 3-6
- content compression, 8-6
- content publishing, 3-4
- content syndication, 3-6
- corporate portals, 3-2
- critical resources, 8-2

D

- data access
 - overview, 2-27
- data consistency, 6-2
- Data Guard, 7-16
- database events, 5-6
- database providers, 3-12
- DCM
 - see Distributed Configuration Management, 11-4*
- Delegated Administration Services, 11-6, 10-6
 - and identity management, 10-6

- deployment
 - as part of the application development
 - process, 2-33
 - recommended topologies, 11-1
 - deployment challenges, 1-3
 - descriptors
 - OracleAS TopLink, 2-30
 - development
 - wireless applications, 4-5
 - development challenges, 1-1
 - development components, 2-24
 - directory administration, 11-6
 - directory integration, 11-6
 - Directory Integration and Provisioning, 11-6
 - and identity management, 10-6
 - disaster recovery, 7-16
 - Discoverer reports
 - portals, 3-8
 - distributed applications
 - Java, 2-10
 - distributed cache, 7-6
 - Java Object Cache, 7-6
 - Distributed Configuration Management, 11-4, 9-4
 - recovery, 7-17
 - DMS, 2-5
 - document lifecycle management, 2-54
 - Document Object Model
 - See DOM
 - DOM, 2-3
 - APIs, 2-50
 - DTDs, 2-50
 - dynamic
 - pages, 2-52
 - dynamic HTML, 2-12
 - PL/SQL, 2-53
 - dynamic monitoring service, 2-5

E

- e-business
 - integration, 1-8
 - portals, 3-2
 - solution, 1-4
- e-business dashboard, 5-20

- Edge Side Includes
 - see *ESI*, 8-7
- Edge Side Includes for Java
 - see *JESI*, 8-7
- EJB client routing, 7-4
- EJB container, 2-15
- EJBs, 2-3
 - container, 2-13, 2-16
 - managing, 2-15
 - running, 2-15
 - types, 2-15
 - version supported, 2-9
- e-mail
 - notifications, Java, 2-18
- enhanced report bursting and distribution, 5-7
- enterprise application integration, 6-3
- enterprise beans, 2-12
- enterprise integration, 1-2
- Enterprise JavaBeans
 - See EJBs
- enterprise reporting, 5-21
- entity beans, 2-15
- ESI, 8-7
- Extensible Stylesheet Language
 - See XSLT
- external API for performance monitoring, 2-6

F

- failover, 8-8
- FastCGI, 2-4, 2-8
- features
 - wireless, 4-2
- file management, 2-4
- forms
 - applications, 2-45
- forms applications, 2-3

G

- Gateway Provider, wireless, 4-8

H

- high availability
 - application tier, 7-5
 - distributed cache, 7-6
 - session persistence, 7-6
 - session replication, 7-5
 - clusters, 7-8
 - infrastructure, 7-11
 - Active Failover Clusters, 7-13
 - Cold Failover Clusters, 7-12
 - requirements, 7-11
 - load balancing, 7-10
 - middle tier, 7-7
 - clustering, 7-8
 - load balancing, 7-10
 - operating system clusters, 7-11
 - overview, 7-4
 - single instance, 7-7
 - features, 7-7
- HTML, 3-4
 - dynamic, 2-12
 - PL/SQL, 2-53
 - forms, 2-53
- HTTP Adapter
 - wireless, 4-9
- HTTP listener, 2-4
- HTTP server modules, 2-4

I

- IDAP, 2-3
- identity management, 11-5
 - architecture, 10-12
 - benefits, 10-2
 - components, 10-3
 - introduction, 10-1
- JAAS, 10-8
- OCA, 10-8
- Oracle Internet Directory, 10-6
 - Delegated Administration Services, 10-6
 - Directory Integration and Provisioning, 10-6
 - features, 10-7
 - searching, 10-7
 - uses, 10-7
- OracleAS Single Sign-On, 10-4
- information sharing, 3-3
- Infrastructure
 - see *Oracle Application Server Infrastructure*, 11-1
- installation types
 - Business Intelligence and Forms, 11-4
 - infrastructure requirements, 11-4
 - J2EE and Web Cache, 11-4
 - Portal and Wireless, 11-4

- integration
 - adapter framework, 6-10
 - adapters, 6-10
 - application, 6-10
 - protocol, 6-10
 - technology, 6-10
 - administration, 6-7
 - benefits, 6-2
 - business activity monitoring, 6-13
 - business process monitoring, 6-13
 - common data format, 6-3
 - components, 6-5
 - data management
 - transformation, 6-12
 - translation, 6-12
 - validation, 6-12
 - design concepts, 6-4
 - features, 6-5
 - integration manager, 6-9
 - introduction, 6-1
 - management, 6-14
 - model-based approach, 6-4
 - modeling, 6-7
 - monitoring, 6-7
 - agreement reports, 6-13
 - business process reports, 6-13
 - event reports, 6-13
 - Oracle Application Server Business Intelligence, 5-18
 - pre-packaged integrations, 6-5
 - reasons for, 6-1
 - systems monitoring, 6-14
- integration manager
 - functions, 6-4
- intelligent routing, 7-2
- internet applications, 1-6
- Internet Data Access Presentation
 - See IDAP
- intranet, 3-2
- introduction
 - wireless, 4-1
- invalidation, 8-7
 - inline, 8-10
 - search keys, 8-10
- items, 3-4, 3-5

J

- J2EE, 1-1, 1-6
 - application clients, 2-12
 - application components, 2-12
 - applications, 1-2, 2-2, 2-10
 - applications, distributed, 2-10
 - caching, 2-19
 - clients, 2-12
 - components, 2-12
 - Connector API
 - version supported, 2-9
 - connector API, 2-18
 - containers, 2-12
 - distributed applications, 2-10
 - Java Object Cache, 2-19
 - modelling, 2-19
 - packaging, 2-13
 - security, 2-18
 - services, 2-16, 2-18
- J2EE Connector Architecture 1.0, 2-3
- J2ME, 4-4
 - using Web services, 4-4
- J2SE, 2-20
- JAAS, 2-3, 2-18
 - Oracle Application Server Single Sign-On, 2-18
 - Oracle Internet Directory, 2-18, 10-7
 - version supported, 2-9
- Java
 - Object Cache, 2-19
 - Oracle HTTP Server, 2-4
 - See J2EE
 - XML, 2-51
- Java 2 Micro Edition See J2ME
- Java API for XML
 - See JAXP
- Java API for XML Parsing
 - See JAXP
- Java Authentication and Authorization Service
 - See JAAS, 10-8
- Java Authentication and Authorization Service (JAAS), 10-8
- Java Database Connectivity
 - See JDBC

- Java Mail, 2-3
- Java Message Service
 - See JMS
- Java Naming and Directory Interface
 - See JNDI
- Java Object Cache, 7-6
- Java object cache, 8-12
- Java servlets
 - See servlets
- Java Transaction API
 - See JTA
- Java2 Platform
 - Enterprise Edition
 - See J2EE
 - Standard Edition
 - See J2SE
- JavaBeans
 - Activation Framework
 - version supported, 2-9
- JavaMail, 2-18
 - version supported, 2-9
- JavaServer Pages
 - See JSPs
- JAXP, 2-3, 2-18
 - version supported, 2-9
- JClient, 2-39
- JDBC, 2-3, 2-16
 - DataDirect drivers, 2-17
 - J2EE Connectors, 2-17
 - non-Oracle connectivity, 2-17
 - Oracle drivers, 2-17
 - version supported, 2-9
- JDeveloper
 - JClient, 2-39
 - See Oracle JDeveloper, 2-35
- JESI, 8-7
- JMS, 2-3, 2-14, 2-17
 - version supported, 2-9
- JNDI, 2-3, 2-17
 - EJBs, relationship, 2-16
- JSP container, 2-14

- JSPs, 2-3, 2-12
 - HTML, 2-14
 - JDBC, 2-16
 - managing, 2-14
 - request flow, 2-15
 - running, 2-14
 - servlets, relationship, 2-14
 - translator, 2-15
 - version supported, 2-9
 - XML, 2-14
- JTA, 2-3, 2-17
 - version supported, 2-9

L

- listener, 2-3
- load balancers
 - hardware, 7-10
 - network, 7-10
 - Web Cache, 7-10
- load balancing, 7-10, 8-8
 - benefits, 7-11
- location-based services, 4-6

M

- management, 1-8, 9-1
 - architecture, 9-3
 - availability, 9-1
 - Business Intelligence, 5-21
 - features, 9-2
 - monitoring, 9-3
 - Oracle Enterprise Manager, 9-1
 - scalability, 9-1
 - supporting technologies, 9-4
 - systems, 1-5
 - user and security, 1-5
- management metadata, identity management
 - metadata, 11-4
- mapping
 - as part of the application development
 - process, 2-31
- mappings
 - OracleAS TopLink metadata, 2-30
- message-driven beans, 2-15

- messaging
 - multimedia, 4-4
- metadata
 - identity management, 11-4
 - management, 11-4
 - product, 11-4
 - project.xml file, 2-29
- metadata repository, 11-3
 - sub-repositories, 11-5
- Microsoft Internet Information Server, 2-6
- middle tier
 - with infrastructure, 11-4
- mobile office applications, 4-7
 - protocols, 4-7
- mod_fastcgi, 2-4
- mod_oc4j, 2-4, 7-2
- mod_oradav, 2-4
- mod_oss, 2-4
- mod_osso, 2-4
- mod_perl, 2-4
- mod_plsql, 2-4, 2-52
- modules, 2-3
 - HTTP server, 2-4
 - mod_plsql, 2-52
- monitoring, 9-3
 - dynamic monitoring service, 2-5
- Multi-Channel Server, 4-3
 - Multimedia Adaptaion Services, 4-3

N

- namespace
 - XML APIs, 2-50
- Netscape iPlanet, 2-6
- notifications, 4-4

O

- object-relational mapping
 - BC4J, 2-37, 2-38
- OC4J
 - architecture, 2-19
 - monitoring, 2-5
 - Web container, 2-13

- OC4J containers, 2-13
 - EJB, 2-15
 - JSP, 2-14
 - servlet, 2-13
- OC4J JSP container, 2-14
- OC4J plug-in, 2-6
- OC4J routing, 7-2
- OCA, 10-8
- online transaction processing systems, 5-2, 5-9
- operating system clusters, 7-11
- Oracle Application Server
 - Overview, 1-6
 - solutions, 1-6
 - Wireless, 1-8
- Oracle Application Server Active Failover Clusters, 7-13
- Oracle Application Server Certificate Authority, 11-2, 11-7
 - See OCA, 10-8
- Oracle Application Server Clusters, 7-8
 - benefits, 7-9
 - managed with database repository, 7-9
 - managed with file-based repository, 7-9
 - manually configured, 7-10
- Oracle Application Server Cold Failover Clusters, 7-12
- Oracle Application Server Infrastructure architecture, 11-7
 - components, 11-2
 - identity management, 11-1
 - Identity Management Services, 11-2
 - introduction, 11-1
 - Product Metadata Services, 11-2
 - using with middle tier, 11-4
- Oracle Application Server Integration
 - <italic>See integration, 6-1
- Oracle Application Server MapViewer, 2-55
- Oracle Application Server Personalization, 5-3
- Oracle Application Server ProcessConnect
 - See OracleAS ProcessConnect, 6-3
- Oracle Application Server Reports Services, 5-4
- Oracle Application Server security, 10-1
- Oracle Application Server Single Sign-On, 5-19, 11-2
- Oracle Business Components for Java, 2-37

- Oracle Content Management SDK, 2-54
 - architecture, 2-54
 - features, 2-54
 - portals, 3-8
- Oracle Data Guard, 7-16
- Oracle Discoverer
 - tools, 5-11
- Oracle Dynamic Monitoring Service, 9-4
- Oracle Enterprise Manager, 9-1
 - Application Server Control, 9-1
 - architecture, 9-3
 - features, 9-2
 - monitoring, 9-3
 - overview, 9-1
 - supporting technologies, 9-4
- Oracle Forms, 2-3
- Oracle HTTP Server, 2-3
 - architecture, 2-6
 - CGI, 2-8
 - components, 2-4
 - FastCGI, 2-8
 - features, 2-5
 - modules, 2-3
 - monitoring, 2-5
 - OC4J plug-in, 2-6
 - proxy plug-in, 2-6
 - request flow, 2-5, 2-7
 - Single Sign-On plug-in, 2-6
- Oracle HTTP Server process architecture, 2-7
- Oracle Identity Management, 11-2, 11-5
- Oracle Internet Developer Suite, 1-5
- Oracle Internet Directory, 11-6, 10-6
 - and identity management, 10-6
 - Delegated Administration Services, 10-6
 - Directory Integration and Provisioning, 10-6
 - features, 10-7
 - JAAS, 2-18, 10-7
 - OracleAS Single Sign-On, 10-7
 - searching, 10-7
 - uses, 10-7
- Oracle JDeveloper, 2-35, 2-52
 - Business Intelligence Beans, 2-37
 - CodeCoach, 2-36
 - debugging, 2-36
 - development environment, 2-36
 - features, 2-36
 - languages supported, 2-36
 - overview, 2-35
- Oracle Management Agent, 9-4
- Oracle Process Manager and Notification Server, 9-4
- Oracle Reports Builder, 5-4, 5-5, 5-22
 - enterprise reporting, 5-21
- Oracle Reports Developer, 5-8
- Oracle UDDI registry, 2-42
- Oracle Ultra Search, 3-6
- Oracle Wireless Development Kit, 4-5
- Oracle XML Developer Kit
 - See XML
- OracleAS Business Intelligence management, 5-21
- OracleAS Discoverer, 5-2, 5-9
 - introduction, 5-9
- OracleAS Discoverer Viewer, 5-11
- OracleAS Forms Services, 2-45
 - architecture, 2-48
 - components, 2-46
 - features, 2-46
- OracleAS MapViewer, 2-55
 - architecture, 2-56
 - components, 2-56
 - features, 2-56
- OracleAS Metadata Repository, 11-2, 11-3
 - sub-repositories, 11-5
- OracleAS Personalization, 5-3
 - features, 5-16
 - introduction, 5-15
- OracleAS Portal
 - deployment with Web Cache, 3-10
 - integration with business intelligence, 5-20
 - integration with SSO, 3-7
 - integration with Web Cache, 3-10
 - searching, 3-5
 - security, 10-11
- OracleAS Portal Developer Kit, 3-7, 3-9

- OracleAS Portlets, 3-8
- OracleAS ProcessConnect
 - adapter framework, 6-10
 - adapters, 6-10
 - application, 6-10
 - protocol, 6-10
 - technology, 6-10
 - administration, 6-7
 - architecture, 6-16
 - topologies, 6-16
 - business activity monitoring, 6-13
 - business process monitoring, 6-13
 - components, 6-5
 - data management
 - transformation, 6-12
 - translation, 6-12
 - validation, 6-12
 - events, 6-6
 - features, 6-5
 - integration manager, 6-9
 - management, 6-14
 - modeling, 6-7
 - monitoring, 6-7
 - agreement reports, 6-13
 - business process reports, 6-13
 - event reports, 6-13
 - overview, 6-3
 - parties, 6-6
 - pre-packaged integrations, 6-5, 6-15
 - collaboration, 6-15
 - document exchange, 6-15
 - packaging, 6-15
 - signing, 6-15
 - transport, 6-15
 - processes, 6-6
 - repositories, 6-8
 - design-time, 6-8
 - runtime, 6-9
 - standards, 6-3
 - systems monitoring, 6-14
 - visual design tools, 6-7
- OracleAS Reports Services, 5-2, 5-4, 5-6
 - architecture, 5-8
 - features, 5-5
- OracleAS Single Sign-On, 5-19, 11-6, 10-4
 - and identity management, 10-4
 - features
 - convenience, 10-5
 - ease of administration, 10-5
 - increased security, 10-5
 - JAAS, 2-18
 - Oracle Internet Directory, 10-7
 - partner and external applications, 10-5
- OracleAS Syndication Server, 3-6
- OracleAS TopLink, 2-24
 - advantages, 2-22
 - application development overview, 2-31
 - problem space, 2-21
- OracleAS TopLink file
 - metadata, 2-29
- OracleAS TopLink Foundation Library
 - overview, 2-26
- OracleAS TopLink Mapping Workbench
 - overview, 2-24
- OracleAS TopLink metadata
 - descriptors, 2-30
 - mappings, 2-30
- OracleAS TopLink Sessions Editor
 - overview, 2-25

- OracleAS Web Cache
 - and third-party systems, 8-12
 - architecture, 8-10
 - automatic content compression, 8-6
 - cache hierarchies, 8-11
 - clustering, 8-8
 - clusters, 7-1
 - deployment with Portal, 3-10
 - dynamic caching, 8-7
 - Edge Side Includes for Java (JESI), 8-7
 - end-user performance monitoring, 8-9
 - failover, 8-8
 - features, 8-6
 - integration with OPMN, 8-9
 - integration with Portal, 3-10
 - introduction, 8-5
 - invalidation, 8-7
 - inline, 8-10
 - search keys, 8-10
 - load balancing, 8-8
 - page assembly components, 8-7
 - partial-page caching, 8-7
 - performance assurance heuristics, 8-8
 - personalized content assembly, 8-7
 - security, 10-8
 - certificate, 10-9
 - certificate authority, 10-9
 - restricted administration, 10-9
 - secure sockets layer support, 10-9
 - wallet, 10-10
 - SSL certificates, 8-9
 - static caching, 8-7
 - workload management, 8-8
- OracleAS Web Cache Manager, 3-11
- OracleAS Web Services
 - deployment, 2-42
 - development tools, 2-41
 - features, 2-41
 - management, 2-42
 - UDDI registry, 2-41

- OracleAS Wireless
 - architecture, 4-8
 - development tools, 4-2
 - features, 4-2
 - Foundation Services, 4-2
 - foundation services, 4-2
 - information flow, 4-1
 - introduction, 4-1
 - J2ME support, 4-4
 - Location Services, 4-6
 - mobile applications, 4-2
 - mobile office applications, 4-7
 - Mobile Portal, 4-2
 - mobile portal, 4-2
 - Multi-Channel Server, 4-2
 - multimedia messaging, 4-4
 - notifications, 4-4
 - overview, 4-1
- overview
 - caching, 8-5

P

- packaging
 - as part of the application development
 - process, 2-33
- pages, portals, 3-1
- parallel page engine, 3-11
- parsing
 - XML, 2-50
 - XML APIs, 2-51
- partner and external applications, 10-5
- partner portlets, 3-9
- PDK, 3-7, 3-9

- performance, 1-5
 - adjustments, 8-4
 - assurance heuristics, 8-8
 - critical resources, 8-2
 - determining factors, 8-1
 - enhancing, 8-3
 - evaluation, 8-3
 - excessive demand, 8-4
 - introduction, 8-1
 - methodology, 8-2
 - response time, 8-2
 - system throughput, 8-2
 - targets, 8-2
 - user expectations, 8-3
 - wait time, 8-2
- performance monitoring
 - end user, 8-9
 - external API, 2-6
- performance tuning
 - as part of the application development process, 2-33
- Perl
 - Oracle HTTP Server, 2-4, 2-8
- personalization, 1-3
 - portal views, 3-3
 - portals, 3-6
- perspectives, 3-5
- PKI, 11-7
- PL/SQL, 2-52
 - applications, 2-2
 - CGI, comparison, 2-52
 - Oracle HTTP Server, 2-4
 - Server Pages
 - See PSP
 - Web toolkit, 2-3, 2-53
- portals, 1-3, 1-6, 3-2
 - access control, 10-11
 - applications, 3-1, 3-7, 3-9
 - architecture, 3-11, 3-12
 - building blocks, 3-4
 - caching, 3-11
 - categories, 3-5
 - content syndication, 3-6
 - developer kit, 3-7, 3-9
 - features, 3-3
 - introduction, 3-1
 - pages, 3-4
 - parallel page engine, 3-11
 - perspectives, 3-5
 - providers, 3-12
 - request flow, 3-12
 - security, 10-11
 - access control, 10-11
 - authentication, 10-11
 - single signon, 3-3, 10-11
 - user authentication, 10-11
 - wireless access, 3-7
- portlets, 3-3, 3-4, 3-8
 - applications, 3-7, 3-9
 - partner, 3-9
 - providers, 3-8
- ProcessConnect
 - See Oracle AS ProcessConnect, 6-3
- product metadata, 11-4
- programming languages, 2-2
- providers
 - portlets, 3-8
 - provisioing, 11-6
 - proxy plug-in, 2-6
 - PSP, 2-3, 2-52
 - public Key Interface (PKI), 10-5
 - publishing content, 3-4

Q

- queries
 - overview, 2-27
- querying
 - as part of the application development process, 2-32

R

- recommended topologies
 - and third-party products, 11-3
 - Departmental, 11-18
 - deployment, 11-2, 11-3
 - development, 11-2
 - Development Life Cycle Support, 11-21
 - Enterprise Data Center, 11-12
 - Java applications, 11-14
 - Portal, Wireless, Business Intelligence, and Forms applications, 11-16
 - Forms, Reports, and Discoverer, 11-7
 - high availability, 11-2, 11-3, 11-23
 - active failover clusters, 11-25
 - cold failover cluster, 11-23
 - installation, 11-2
 - Integration Architect and Process Modeler, 11-9
 - introduction, 11-1
 - Java Developer, 11-4
 - management, 11-3
 - performance, 11-3
 - Portal and Wireless, 11-5
 - security, 11-3
- recovery, 7-14
 - cold backup and restore, 7-15
 - disaster, 7-16
 - Distributed Configuration Management, 7-17
 - online backup and restore, 7-15
 - restarting processes, 7-14
- regions, 3-4
- registries
 - UDDI, 2-45
- remote connectivity, 2-16
- Remote Method Invocation
 - See RMI
- Remote Procedure Call
 - See RPC
- reports
 - portals, 3-8

- request flow
 - Java
 - JSPs, 2-15
 - servlets, 2-14
 - Oracle HTTP Server, 2-5, 2-7
 - OracleAS Forms Services, 2-48
 - portals, 3-12
- request ID, 2-5
- Request Processor, wireless, 4-8
- response time, 8-2
- RMI, 2-14
- routing
 - algorithms, 7-2
 - EJB client, 7-4
 - intelligent, 7-2
 - OC4J, 7-2
 - styles, 7-2
- RPC, 2-45

S

- SAX, 2-3
 - APIs, 2-50
- scalability, 1-4
 - definition, 7-1
- schemas
 - XML, 2-3
 - XML processors, 2-51
- searching
 - OracleAS Portal, 3-5
- Secure Sockets Layer
 - see SSL, 8-9
- security
 - architecture, 10-12
 - features, 10-2
 - Oracle Application Server components, 10-2
 - OracleAS Web Cache, 10-8

- security features, 1-8, 10-2
 - Java, 2-18
 - Java Authentication and Authorization Service (JAAS), 10-8
 - Oracle Internet Directory, 10-6, 10-7
 - Delegated Administration Services, 10-6
 - Directory Integration and Provisioning, 10-6
 - searching, 10-7
 - self-service console, 10-6
 - synchronization with Third Party LDAP servers, 10-6
 - OracleAS Single Sign-On, 10-4
 - portals, 10-11
- security management, 1-4, 1-5
- self-service, 3-3, 3-4
- server accelerators, 8-5
- services
 - J2EE, 2-16
- servlet container, 2-13
- servlets, 2-3, 2-12
 - container, 2-14
 - JDBC, 2-16
 - JSPs, relationship, 2-14
 - managing, 2-13
 - request flow, 2-14
 - running, 2-13
 - version supported, 2-9
 - Web services, relationship, 2-42
 - XSQL, 2-51
- session beans, 2-15
- session management
 - as part of the application development process, 2-32
- session persistence, 7-6
- session replication, 7-5
- sessions
 - overview, 2-27
- Simple API for XML
 - See SAX
- Simple Object Access Protocol
 - See SOAP
- single signon, 2-4, 3-3
 - portals, 3-3, 10-11
- Single Sign-On plug-in, 2-6
- smart routing, 7-2

- SOAP, 2-3, 2-45
- SQL
 - JDBC, 2-16
 - XML, 2-51
- SSL
 - certificates, 8-9
 - styles, 3-5
 - subscribers, content, 3-6
 - syndication server, 3-6
 - system failure
 - recovery, 7-14
 - system management, 1-4
 - features, 9-2
 - Oracle Internet Directory, 10-6
 - OracleAS JAAS, 10-8
 - OracleAS Single Sign-On, 10-4
 - Oracle Enterprise Manager, 9-1
 - Oracle Internet Directory, 10-1
 - overview, 9-1
 - security, 10-1
 - system throughput, 8-2
 - systems management, 1-5

T

- technologies
 - supported, 2-2
- third-party
 - portlets, 3-9
- transactions
 - as part of the application development process, 2-33
 - overview, 2-28
 - two-phase commit, 2-17
- transviewer beans
 - XML, 2-52
- two-phase commit, 2-17

U

- UDDI, 2-3, 2-42, 2-45
- UDDI registry, 2-41
- Ultra Search, 3-6
- Universal Description, Discovery, and Integration
 - See UDDI

- user authentication
 - portals, 10-11
- user interfaces
 - Java, 2-20
- user management, 1-5
- user profiles, 3-6

V

- version control, 2-4

W

W3C

- XML schemas, 2-51
- XML specifications, 2-49

- wait time, 8-2

- Web Cache, 1-8

- Web Cache clusters, 7-1

- web clipping, 4-6

- Web listener, 2-3

- Web object cache, 8-12

- Web pages, 3-4

- Web providers, 3-12

- Web servers, 2-3

- non-Oracle servers, 2-6

- Web services, 1-2, 2-40

- architecture, 2-42

- characteristics, 2-40

- Description Language

- See* WSDL

- framework, 2-44

- J2ME and wireless devices, 4-4

- servlets, relationship, 2-42

- SOAP, 2-45

- UDDI, 2-3, 2-45

- WSDL, 2-45

- wireless, 1-3, 1-8

- architecture, 4-8

- device/network adapter, 4-9

- features, 4-2

- HTTP Adapter, 4-9

- introduction, 4-1

- portal connectivity, 3-7

- transformers, 4-9, 4-10

- device-specific, 4-10

- generic, 4-10

- WAP device, 4-9

- XML, 3-7

- XML object, 4-9

- wireless development, 4-5

- Wireless Web Clipping Server, 4-6

- WML, 2-12

- WSDL, 2-3, 2-45

- WSDLWeb Services Description Language *See* WSDL

X

XDK

See XML

XML

applications, 2-2

C++, 2-51

class generators, 2-51

database, 2-51

datagrams, 2-51

descriptor files, 2-10

developer kit, 2-49

DOM, 2-3

DTDs, 2-50

IDAP, 2-3

Java, 2-51

JAXP, 2-9

JSPs, 2-12

namespace

 APIs, 2-50

navigating, 2-50

pages, 2-12

parser APIs, 2-51

parsers, 2-50

parsing, 2-50

Path Language

See XPath

portals, 3-4

SAX, 2-3

schema processors, 2-51

schemas, 2-3

SOAP, 2-3

SQL, 2-3

stylesheets, 2-50

supported version, 2-3

transviewer beans, 2-52

UDDI, 2-3

W3C specifications, 2-49

wireless, 3-7

WSDL, 2-3

XPath, 2-3

XSLT, 2-3

XSLT APIs, 2-51

XML Namespaces

supported version, 2-3

XPath, 2-3

XSLT, 2-3

 APIs, 2-51

 parsers, 2-50

XSQL, 2-3

 servlet, 2-51

