Oracle® Fusion Middleware

High Availability Guide

11*g* Release 1 (11.1.1)

E10106-01

June 2009

Oracle Fusion Middleware High Availability Guide, 11*g* Release 1 (11.1.1)

E10106-01

Primary Author: Joseph Paul

Contributing Author: Fermin Castro, Shailesh Dwivedi, Pradeep Bhat, Michael Rhys, Samrat Ray, Jeni Ferns, Richard Delval, Bharath K. Reddy, Susan Kornberg, Bert Rich, Ajay Keni, Amit Sharma, Vasuki Ashok, Olaf Stullich, Jingjing Wei, Olfat Aly, Eileen He, Ramaprakash Sathyanarayan, Apurv Chandra, Atika Jain, Buddhika Kottahachchi, Arun Theebaprakasam, Rajiv Jaisankar, Vinay Kalra, Daniel Shih, Pratima Gogineni, Russ Hodgson, Christopher Johnson, Pramodini Gattu, Joseph Garcia, Deborah Steiner, Gururaj BS, Ben Gelernter, Peter Jacobsen.

Contributor: Eileen He, Ellen Desmond, Don Biasotti, Vinaye Misra, Dhaval Shah, Herbert Stiel, Albert Tam, Pushkar Kapasi, Shail Goel, Rahul Menezes, Satheesh Amilineni, Ratheesh Pai, Jan Carlin.

Contents

6 Configuring High Availability for Oracle ADF and WebCenter Applications

Preface

This preface contains these sections:

- Intended Audience
- Documentation Accessibility
- Related Documentation
- Conventions

Intended Audience

The *Oracle Fusion Middleware High Availability Guide* is intended for administrators, developers, and others whose role is to deploy and manage Oracle Fusion Middleware with high availability requirements.

Documentation Accessibility

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

Accessibility of Code Examples in Documentation

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

Accessibility of Links to External Web Sites in Documentation

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

Deaf/Hard of Hearing Access to Oracle Support Services

To reach Oracle Support Services, use a telecommunications relay service (TRS) to call Oracle Support at 1.800.223.1711. An Oracle Support Services engineer will handle technical issues and provide customer support according to the Oracle service request process. Information about TRS is available at `http://www.fcc.gov/cgb/consumerfacts/trs.html`, and a list of phone numbers is available at `http://www.fcc.gov/cgb/dro/trsphonebk.html`.

Related Documentation

For more information, see these Oracle resources:

- *Oracle Fusion Middleware Concepts*
- *Oracle Fusion Middleware Administrator's Guide*

Conventions

The following text conventions are used in this document:

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

1

Introduction to High Availability

A high availability architecture is one of the key requirements for any Enterprise Deployment. Oracle Fusion Middleware has an extensive set of high availability features, which protect its components and applications from unplanned down time and minimize planned downtime.

The solutions and procedures described in this book are designed to eliminate single points of failure for Oracle Fusion Middleware components with no or minimal down time. These solutions help ensure that applications that deployed with Oracle Fusion Middleware meet the required availability to achieve your business goals.

This guide discusses the architecture, interaction, and dependencies of Oracle Fusion Middleware components, and explains how they can be deployed in a high availability architecture.

This chapter explains high availability and its importance from the perspective of Oracle Fusion Middleware. This chapter includes the following sections:

- Section 1.1, "What is High Availability"
- Section 1.2, "High Availability Information in Other Documentation"

1.1 What is High Availability

High availability refers to the ability of users to access a system without loss of service. Deploying a high availability system minimizes the time when the system is down, or unavailable and maximizes the time when it is running, or available. This section provides an overview of high availability from a problem-solution perspective. This section includes the following topics:

- Section 1.1.1, "High Availability Problems"
- Section 1.1.2, "High Availability Solutions"

1.1.1 High Availability Problems

Mission critical computer systems need to be available 24 hours a day, 7 days a week, and 365 days a year. However, part or all of the system may be down during planned or unplanned downtime. A system's availability is measured by the percentage of time that it is providing service in the total time since it is deployed. Table 1–1 provides an example.

Table 1–1 Availability Percentages and Corresponding Downtime Values

Availability Percentage	Approximate Downtime Per Year
95%	18 days

Table 1–1 (Cont.) Availability Percentages and Corresponding Downtime Values

Availability Percentage	Approximate Downtime Per Year
99%	4 days
99.9%	9 hours
99.99%	1 hour
99.999%	5 minutes

System downtime may be categorized as planned or unplanned. Unplanned downtime is any sort of unexpected failure. Planned downtime refers to scheduled operations that are known in advance and that render the system unavailable. The effect of planned downtime on end users is typically minimized by scheduling operational windows when system traffic is slow. Unplanned downtime may have a larger effect because it can happen at peak hours, causing a greater impact on system users.

These two types of downtimes (planned and unplanned) are usually considered separately when designing a system's availability requirements. A system's needs may be very restrictive regarding its unplanned downtimes, but very flexible for planned downtimes. This is the typical case for applications with high peak loads during working hours, but that remain practically inactive at night and during weekends. You may choose different high availability features depending on the type of failure is being addressed.

1.1.2 High Availability Solutions

High availability solutions can be categorized into local high availability solutions that provide high availability in a single data center deployment, and disaster recovery solutions, which are usually geographically distributed deployments that protect your applications from disasters such as floods or regional network outages.

Amongst possible types of failures, process, node, and media failures as well as human errors can be protected by local high availability solutions. Local physical disasters that affect an entire data center can be protected by geographically distributed disaster recovery solutions.

To solve the high availability problem, a number of technologies and best practices are needed. The most important mechanism is redundancy. High availability comes from redundant systems and components. You can categorize local high availability solutions by their level of redundancy, into active-active solutions and active-passive solutions (see Figure 1–1):

- **Active-active solutions** deploy two or more active system instances and can be used to improve scalability as well as provide high availability. In active-active deployments, all instances handle requests concurrently.
- **Active-passive solutions** deploy an active instance that handles requests and a passive instance that is on standby. In addition, a heartbeat mechanism is set up between these two instances. This mechanism is provided and managed through operating system vendor-specific clusterware. Generally, vendor-specific cluster agents are also available to automatically monitor and failover between cluster nodes, so that when the active instance fails, an agent shuts down the active instance completely, brings up the passive instance, and application services can successfully resume processing. As a result, the active-passive roles are now switched. The same procedure can be done manually for planned or unplanned

downtime. Active-passive solutions are also generally referred to as cold failover clusters.

You can use Oracle Cluster Ready Services (CRS) to manage the Fusion Middleware Active-Passive (CFC) solutions.

Figure 1–1 Active-Active and Active-Passive High Availability Solutions

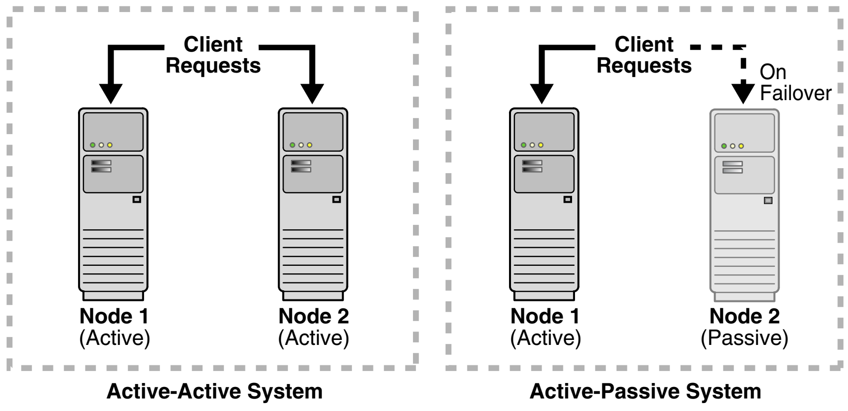


Figure 1–1 shows the general difference between an active-active system and an active-passive system.

****

In addition to architectural redundancies, the following local high availability technologies are also necessary in a comprehensive high availability system:

- **Process death detection and automatic restart**

  Processes may die unexpectedly due to configuration or software problems. A proper process monitoring and restart system should monitor all system processes constantly and restart them should problems appear.

  A system process should also maintain the number of restarts within a specified time interval. This is also important since continually restarting within short time periods may lead to additional faults or failures. Therefore a maximum number of restarts or retries within a specified time interval should also be designed as well.

- **Clustering**

  Clustering components of a system together allows the components to be viewed functionally as a single entity from the perspective of a client for runtime processing and manageability. A cluster is a set of processes running on single or multiple computers that share the same workload. There is a close correlation between clustering and redundancy. A cluster provides redundancy for a system.

  If failover occurs during a transaction in a clustered environment, the session data is retained as long as there is at least one surviving instance available in the cluster.

- **State replication and routing**

  For stateful applications, client state can be replicated to enable stateful failover of requests in the event that processes servicing these requests fail.

- **Failover**

  With a load-balancing mechanism in place, the instances are redundant. If any of the instances fail, requests to the failed instance can be sent to the surviving instances.

- **Server load balancing**

  When multiple instances of identical server components are available, client requests to these components can be load balanced to ensure that the instances have roughly the same workload.

- **Server Migration**

  Some services can only have one instance running at any given point of time. If the active instance becomes unavailable, the service is automatically started on a different cluster member. Alternatively, the whole server process can be automatically started on a different machine in the cluster.

- **Integrated High Availability**

  Components depend on other components to provide services. The component should be able to recover from dependent component failures without any service interruption.

- **Rolling Patching**

  Patching product binaries often requires down time. Patching a running cluster in a rolling fashion can avoid downtime. Patches can be uninstalled in a rolling fashion as well.

- **Configuration management**

  A clustered group of similar components often need to share common configuration. Proper configuration management ensures that components provide the same reply to the same incoming request, allows these components to synchronize their configurations, and provides high availability configuration management for less administration downtime.

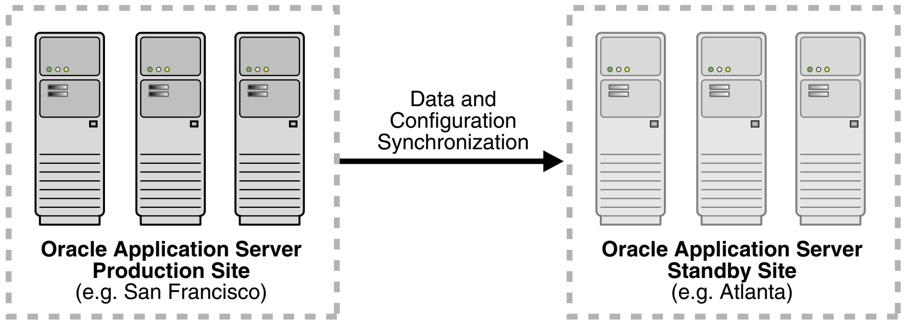
- **Backup and Recovery**

  User errors may cause a system to malfunction. In certain circumstances, a component or system failure may not be repairable. A backup and recovery facility should be available to back up the system at certain intervals and restore a backup when an unrepairable failure occurs.

Disaster Recovery

Disaster recovery solutions typically set up two homogeneous sites, one active and one passive. Each site is a self-contained system. The active site is generally called the production site, and the passive site is called the standby site. During normal operation, the production site services requests; in the event of a site failover or switchover, the standby site takes over the production role and all requests are routed to that site. To maintain the standby site for failover, not only must the standby site contain homogeneous installations and applications, data and configurations must also be synchronized constantly from the production site to the standby site.

Figure 1–2 Geographically Distributed Disaster Recovery



Oracle Fusion Middleware Components Protected by High Availability Solutions

The Oracle Fusion Middleware High Availability Guide discusses high availability solutions for the following components:

- Oracle WebLogic Server
- Oracle SOA Suite
- Oracle ADF
- Oracle WebCenter
- Oracle Identity Management Components
- Oracle HTTP Server
- Oracle Web Cache
- Oracle Portal, Forms, Reports, and Discoverer

1.2 High Availability Information in Other Documentation

Table 1–2 lists Oracle Fusion Middleware guides (other than this guide) that contain high availability information. This information pertains to high availability of various Oracle Fusion Middleware components.

Table 1–2 High Availability Information in Oracle Fusion Middleware Documentation

Component	Location of Information
Oracle SOA Suite	The *Oracle Fusion Middleware Administrator's Guide for Oracle SOA Suite* The *Oracle Fusion Middleware Installation Guide for Oracle SOA Suite* The *Oracle Fusion Middleware Enterprise Deployment Guide for Oracle SOA Suite*
Oracle WebCenter	The *Oracle Fusion Middleware Administrator's Guide for Oracle WebCenter* The *Oracle Fusion Middleware Installation Guide for Oracle WebCenter* The *Oracle Fusion Middleware Enterprise Deployment Guide for Oracle WebCenter*
Oracle ADF	The *Oracle Fusion Middleware Fusion Developer's Guide for Oracle Application Development Framework* The *Oracle Fusion Middleware Web User Interface Developer's Guide for Oracle Application Development Framework*
Oracle WebLogic Server Clusters	The *Oracle Fusion Middleware Using Clusters for Oracle WebLogic Server*
Oracle Fusion Middleware Backup and Recovery	The *Oracle Fusion Middleware Administrator's Guide*

Table 1–2 (Cont.) High Availability Information in Oracle Fusion Middleware Documentation

Component	Location of Information
Oracle Web Cache	The *Oracle Fusion Middleware Administrator's Guide for Oracle Web Cache*
Oracle Identity Management	The *Oracle Fusion Middleware Installation Guide for Oracle Identity Management* The *Oracle Fusion Middleware Enterprise Deployment Guide for Oracle Identity Management*
Oracle Virtual Directory	The *Oracle Fusion Middleware Administrator's Guide for Oracle Virtual Directory*
Oracle HTTP Server	The *Oracle Fusion Middleware Administrator's Guide for Oracle HTTP Server*
Oracle Internet Directory	The *Oracle Fusion Middleware Administrator's Guide for Oracle Internet Directory*
Oracle Real Application Clusters (RAC)	The *Oracle Real Application Clusters Installation Guide*
Oracle Repository Creation Utility (RCU)	The *Oracle Fusion Middleware Repository Creation Utility User's Guide*
Oracle Portal	The *Oracle Fusion Middleware Administrator's Guide for Oracle Portal*

2 Oracle Fusion Middleware High Availability Framework

This chapter describes the Oracle Fusion Middleware features that are important in high availability topologies. It contains the following topics:

- Section 2.1, "Understanding Key Oracle Fusion Middleware Concepts"
- Section 2.2, "Oracle Fusion Middleware High Availability Terminology"
- Section 2.3, "Oracle Fusion Middleware High Availability Solutions"
- Section 2.4, "Protection from Planned and Unplanned Down Time"

2.1 Understanding Key Oracle Fusion Middleware Concepts

Oracle Fusion Middleware provides two types of components:

- Java components, which are manageable processes deployed to Oracle WebLogic Server and are managed by Oracle WebLogic Server. A Java component has a one-to-one relationship with a domain extension template. It generally refers to a collection of applications and resources. Java components include Oracle SOA Suite and Oracle WebCenter Spaces.
- A system component, which is a manageable process that is not deployed as a Java application. Instead, a system component is managed by the Oracle Process Manager and Notification (OPMN). System components include Oracle Internet Directory, Oracle HTTP Server, Oracle Web Cache, and Java Standard Edition (JSE) components, such as Oracle BI Enterprise Edition.

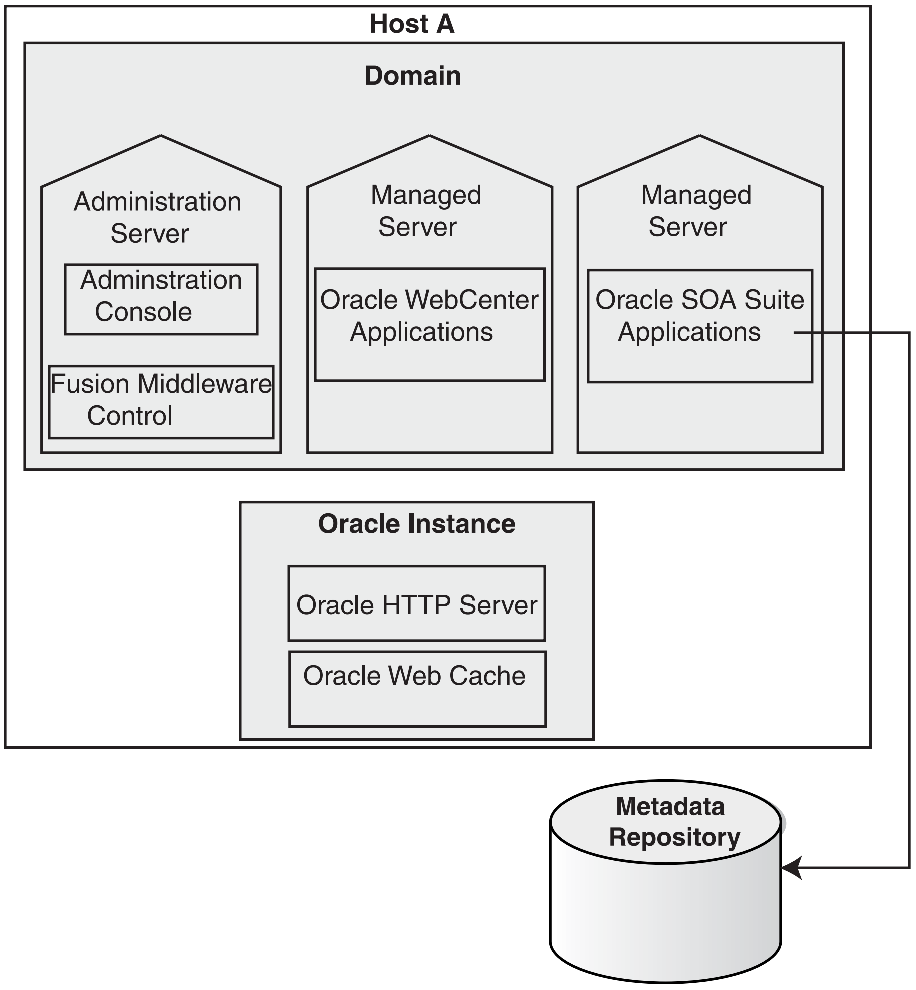
A Java component and a system component are peers.

After you install and configure Oracle Fusion Middleware, your Oracle Fusion Middleware environment contains the following:

- An Oracle WebLogic Server domain, which contains one Administration Server and one or more managed servers.
- If your environment includes system components, one or more system component domains.
- An Oracle Metadata Repository, if the components you installed require one. For example, Oracle SOA Suite requires an Oracle Metadata Repository.

Figure 2–1 shows an Oracle Fusion Middleware environment with an Oracle WebLogic Server domain with an administration server and two managed servers, a system component domain, and an Oracle Metadata Repository.

Figure 2–1 Oracle Fusion Middleware Environment



Your environment also includes a Middleware home, which consists of the Oracle WebLogic Server home, and, optionally, one or more Oracle homes.

2.1.1 What is a WebLogic Server Domain?

A WebLogic Server administration **domain** is a logically related group of Java components. A domain includes a special WebLogic Server instance called the **Administration Server,** which is the central point from which you configure and manage all resources in the domain. Usually, you configure a domain to include additional WebLogic Server instances called *managed servers*. You deploy Java components, such as Web applications, EJBs, and Web services, and other resources to the managed servers and use the Administration Server for configuration and management purposes only.

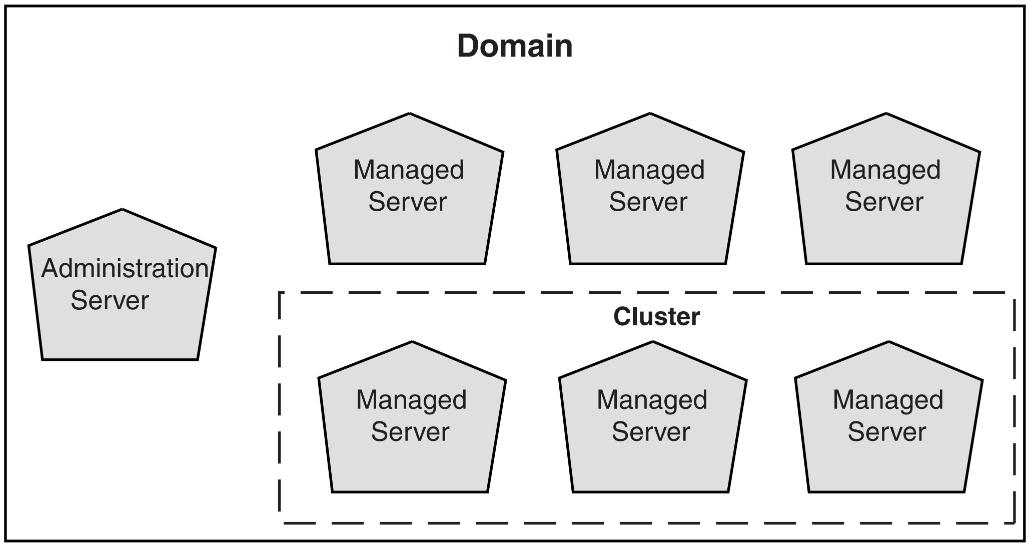
Managed servers in a domain can be grouped together into a cluster.

An Oracle WebLogic Server Domain is a peer of a system component domain. Both contain specific configurations outside of their Oracle homes.

The directory structure of an WebLogic Server domain is separate from the directory structure of the WebLogic Server Home. It can reside anywhere; it need not be within the Middleware home directory.

Figure 2–2 shows a Oracle WebLogic Server domain with an Administration Server, three standalone managed servers, and three managed servers in a cluster.

Figure 2–2 Oracle WebLogic Server Domain



See Also: *Oracle Fusion Middleware Understanding Domain Configuration for Oracle WebLogic Server* for more information about domain configuration

The following topics describe entities in the domain:

- What Is the Administration Server?
- Understanding Managed Servers and Managed Server Clusters
- What Is Node Manager?

2.1.1.1 What Is the Administration Server?

The **Administration Server** operates as the central control entity for the configuration of the entire domain. It maintains the domain's configuration documents and distributes changes in the configuration documents to managed servers. You can use the Administration Server as a central location from which to monitor all resources in a domain.

Each WebLogic Server domain must have one server instance that acts as the Administration Server.

To interact with the Administration Server, you can use the Oracle WebLogic Server Administration Console, Oracle WebLogic Scripting Tool (WLST), or create your own JMX client. In addition, you can use Oracle Enterprise Manager Fusion Middleware Control for some tasks.

Fusion Middleware Control and the WebLogic Administration Console run in the Administration Server. Fusion Middleware Control is a Web-based administration console used to manage Oracle Fusion Middleware, including components such as Oracle HTTP Server, Oracle SOA Suite and Oracle WebCenter, Oracle Portal, Forms, Reports, and Discoverer, and the Oracle Identity Management components. Oracle WebLogic Server Administration Console is the Web-based administration console used to manage the resources in an Oracle WebLogic Server domain, including the Administration Server and managed servers in the domain.

2.1.1.2 Understanding Managed Servers and Managed Server Clusters

Managed servers host business applications, application components, Web services, and their associated resources. To optimize performance, managed servers maintain a read-only copy of the domain's configuration document. When a managed server starts up, it connects to the domain's Administration Server to synchronize its configuration document with the document that the Administration Server maintains.

When you create a domain, you create it using a particular domain template. That template supports a particular component or group of components, such as the Oracle SOA Suite. The Managed Servers in the domain are created specifically to host those particular Oracle Fusion Middleware system components.

Java-based Oracle Fusion Middleware system components (such as Oracle SOA Suite, Oracle WebCenter, and some Identity Management components), as well as customer-developed applications, are deployed to Managed Servers in the domain.

If you want to add other components, such as Oracle WebCenter, to a domain that was created using a template that supports another component, you can extend the domain by creating additional Managed Servers in the domain, using a domain template for the component which you want to add.

For production environments that require increased application performance, throughput, or high availability, you can configure two or more Managed Servers to operate as a cluster. A **cluster** is a collection of multiple WebLogic Server server instances running simultaneously and working together to provide increased scalability and reliability. In a cluster, most resources and services are deployed identically to each Managed Server (as opposed to a single Managed Server), enabling failover and load balancing. A single domain can contain multiple WebLogic Server clusters, as well as multiple Managed Servers that are not configured as clusters. The key difference between clustered and non-clustered Managed Servers is support for failover and load balancing. These features are available only in a cluster of Managed Servers.

See Also: Understanding WebLogic Server Clustering" in *Oracle Fusion Middleware Using Clusters for Oracle WebLogic Server*

2.1.1.3 What Is Node Manager?

Node Manager is a Java utility that runs as separate process from Oracle WebLogic Server and allows you to perform common operations for a Managed Server, regardless of its location with respect to its Administration Server. While use of Node Manager is optional, it provides valuable benefits if your WebLogic Server environment hosts applications with high-availability requirements.

If you run Node Manager on a machine that hosts Managed Servers, you can start and stop the Managed Servers remotely using the Administration Console or the command line. Node Manager can also automatically restart a Managed Server after an unexpected failure.

See Also: *Oracle Fusion Middleware Node Manager Administrator's Guide for Oracle WebLogic Server*

2.1.2 What Is a System Component Domain?

A **system component domain** contains one or more system components, such as Oracle Web Cache, Oracle HTTP Server, or Oracle Internet Directory. The system components in a system component domain must reside on the same machine. A system component domain directory contains files that can be updated, such as configuration files, log files, and temporary files.

A system component domain is a peer of a Oracle WebLogic Server domain. Both contain specific configurations outside of their Oracle homes.

The directory structure of a system component domain is separate from the directory structure of the Oracle home. It can reside anywhere; it need not be within the Middleware home directory.

2.1.3 What Is a Middleware Home?

A **Middleware home** consists of the Oracle WebLogic Server home, and, optionally, one or more Oracle homes.

A Middleware home can reside on a local file system or on a remote shared disk that is accessible through NFS.

See Section 2.1.4, "What Is an Oracle Home?" for information about Oracle homes. See Section 2.1.1, "What is a WebLogic Server Domain?" for information about Oracle WebLogic Server homes.

2.1.4 What Is an Oracle Home?

An **Oracle home** contains installed files necessary to host a specific product. For example, the SOA Oracle home contains a directory that contains binary and library files for Oracle SOA Suite.

An Oracle home resides within the directory structure of the Middleware home. Each Oracle home can be associated with multiple system component domains or Oracle WebLogic Server domains.

2.1.5 What Is a WebLogic Server Home?

A WebLogic Server home contains installed files necessary to host a WebLogic Server. The WebLogic Server home directory is a peer of Oracle home directories and resides within the directory structure of the Middleware home.

2.2 Oracle Fusion Middleware High Availability Terminology

The definitions of terms listed in this section are useful in helping to understand the concepts presented in this book:

- **failover**: When a member of a high availability system fails unexpectedly (unplanned downtime), in order to continue offering services to its consumers, the system undergoes a failover operation. If the system is an active-active system, the failover is performed by the load balancer entity serving requests to the active members. If an active member fails, the load balancer detects the failure and automatically redirects requests for the failed member to the surviving active members.

  If the system is an active-passive system, the passive member is activated during the failover operation and consumers are directed to it instead of the failed member. The failover process can be performed manually, or it can be automated by setting up hardware cluster services to detect failures and move cluster resources from the failed node to the standby node.

- **failback**: Failback is a planned operation after unplanned downtime. After a system undergoes a successful failover operation, the original failed member can be repaired over time and can be re-introduced into the system as a standby member. If desired, a failback process can be initiated to activate this member and deactivate the other. This process reverts the system back to its pre-failure configuration.
- **shared storage**: Although each node in a cluster is a standalone server that runs its own set of processes, there are some file system based data and configuration elements which need uniform access from all nodes in a cluster. Shared storage refers to the ability of the cluster to be able to access the same storage, usually disks, from any node in the cluster.

  For SAN based deployments, a clustered file system, such as OCFS, may also be needed. Some examples of this usage are, JMS file based persistence store, transaction logs persistence store, domain configuration in case of cold failover cluster setup.
- **primary node**: The node that is actively running Oracle Fusion Middleware at any given time in a cluster. If this node fails, Oracle Fusion Middleware is failed over to the secondary node. Because the primary node runs the active Oracle Fusion Middleware installation(s). See the definition for secondary node in this section.
- **secondary node**: If this node fails, Oracle Fusion Middleware is failed over to the secondary node.
- **network hostname**: Network hostname is a name assigned to an IP address either through the `/etc/hosts` file (on UNIX), `C:\WINDOWS\system32\drivers\etc\hosts` file (on Windows), or through DNS resolution. This name is visible in the network that the machine to which it refers to is connected. Often, the network hostname and physical hostname are identical. However, each machine has only one physical hostname but may have multiple network hostnames. Thus, a machine’s network hostname may not always be its physical hostname.
- **physical hostname**: This guide differentiates between the terms physical hostname and network hostname. This guide uses physical hostname to refer to the "internal name" of the current machine. On UNIX, this is the name returned by the `hostname` command.
- **switchover and switchback**: Switchover and switchback are planned operations. During normal operation, active members of a system may require maintenance or upgrading. A switchover process can be initiated to allow a substitute member to take over the workload performed by the member that requires maintenance, upgrading, or any planned downtime. The switchover operation ensures continued service to consumers of the system.

  When a switchover operation is performed, a member of the system is deactivated for maintenance or upgrading. When the maintenance or upgrading is completed, the system can undergo a switchback operation to activate the upgraded member and bring the system back to the pre-switchover configuration.
- **virtual IP**: A virtual IP can be assigned to a cluster or load balancer. To present a single system view of a cluster to network clients, a virtual IP serves as an entry point IP address to the group of servers which are members of the cluster. A virtual IP can be assigned to a server load balancer or a hardware cluster.

A load balancer also uses a virtual IP as the entry point to a set of servers. These servers tend to be active at the same time. This virtual IP address is not assigned to any individual server but to the load balancer which acts as a proxy between servers and their clients.

- **virtual hostname**: A virtual hostname in a cluster is a network hostname assigned to virtual IP bound to one of the nodes in the cluster at any given time.

> **Note:** Whenever the term "virtual hostname" is used in this document, it is assumed to be associated with a virtual IP address. In cases where just the IP address is needed or used, it is explicitly stated.

- **hardware cluster**: A hardware cluster is a collection of computers that provides a single view of network services (for example: an IP address) or application services (for example: databases, Web servers) to clients of these services. Each node in a hardware cluster is a standalone server that runs its own processes. These processes can communicate with one another to form what looks like a single system that cooperatively provides applications, system resources, and data to users.

  A hardware cluster achieves high availability and scalability through the use of specialized hardware (cluster interconnect, shared storage) and software (health monitors, resource monitors). (The cluster interconnect is a private link used by the hardware cluster for heartbeat information to detect node death.) Due to the need for specialized hardware and software, hardware clusters are commonly provided by hardware vendors such as Sun, HP, IBM, and Dell. While the number of nodes that can be configured in a hardware cluster is vendor dependent, for the purpose of Oracle Fusion Middleware high availability, only two nodes are required. Hence, this document assumes a two-node hardware cluster for high availability solutions employing a hardware cluster.

- **cluster agent**: The software that runs on a node member of a hardware cluster that coordinates availability and performance operations with other nodes. Clusterware provides resource grouping, monitoring, and the ability to move services. A cluster agent can automate the service failover.

- **clusterware**: A software that manages the operations of the members of a cluster as a system. It allows one to define a set of resources and services to monitor using a heartbeat mechanism between cluster members and to move these resources and services to a different member in the cluster as efficiently and transparently as possible.

2.3 Oracle Fusion Middleware High Availability Solutions

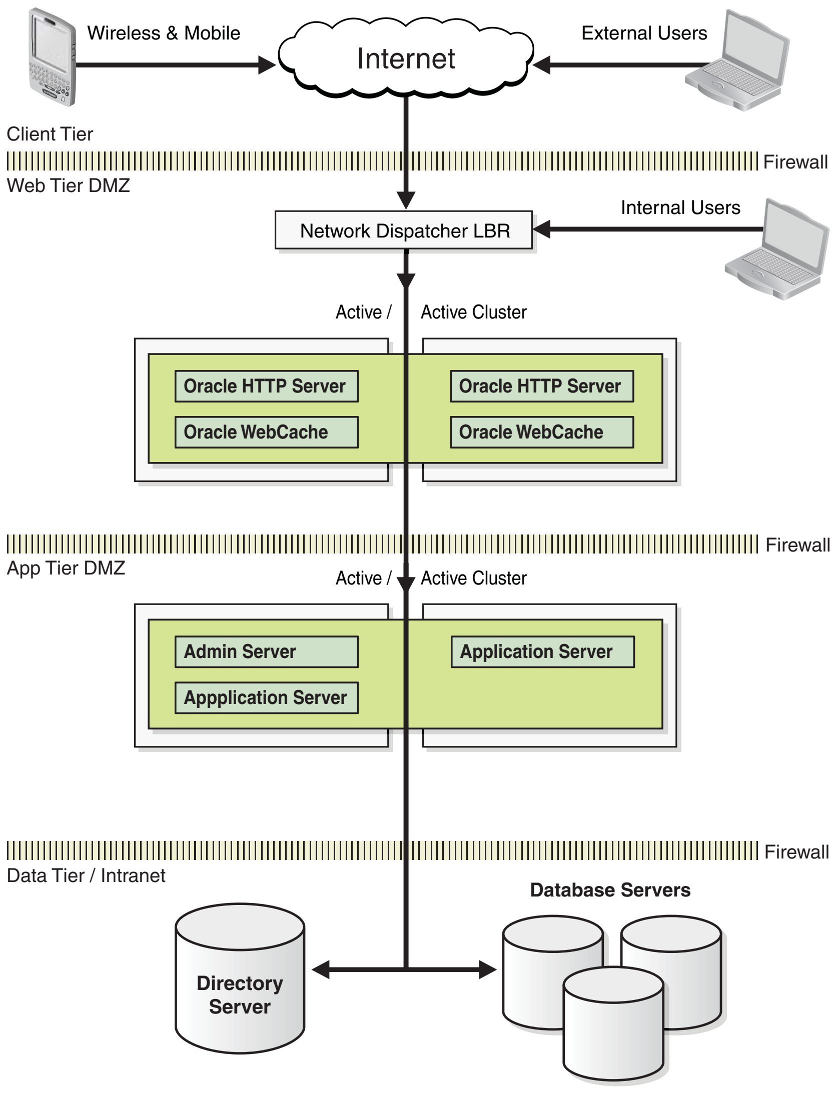
This section describes local high availability concepts, and Oracle Fusion Middleware high availability technologies

2.3.1 Local High Availability

Local high availability solutions can be categorized as either active-active or active-passive solutions. Oracle Fusion Middleware supports both active-active deployments as well as active-passive deployments.

Figure 2–3 shows an Oracle Fusion Middleware high availability active-active deployment topology.

Figure 2–3 Oracle Fusion Middleware Enterprise Deployment Architecture



As shown in Figure 2–3, this topology represents a multi-tiered architecture. Users access the system from the client tier. Requests go through a hardware load balancer, which then routes them to a Web server cluster, running Oracle HTTP Server. Web servers use Proxy Plug-in (mod_wl_ohs) to route the requests to the WebLogic cluster. Applications running on the WebLogic cluster then interact with the database cluster to service the request.

There is no single point of failure in the entire architecture. WebLogic Administration Server is configured in Cold Failover Cluster mode, as described in Section 10.2.2.3, "Transforming the Administration Server for Cold Failover Clusters," and is protected using external clusterware.

2.3.2 Oracle Fusion Middleware High Availability Technologies

The Oracle Fusion Middleware infrastructure has following high availability features:

- **Process death detection and automatic restart**

For Java EE components running on WebLogic Server, Node Manager monitors the Managed Servers. If a Managed Server goes down, it attempts to restart the Managed Server for a configured number of times.

For system components, OPMN monitors the processes. If a system component process goes down, OPMN attempts to restart the process for a configurable number of times.

- **Clustering**

  Oracle Fusion Middleware Java EE components leverage underlying powerful WebLogic Server clustering capabilities to provide clustering. Oracle Fusion Middleware uses WebLogic clustering capabilities, such as redundancy, failover, session state replication, cluster-wide JNDI services, Whole Server Migration, and cluster wide configuration.

  These capabilities provide for seamless failover of all Java EE Oracle Fusion Middleware system components transparent to the client preserving session and transaction data, as well as ensuring data consistency. For further description of these features, see Chapter 3, "High Availability for WebLogic Server."

  System components can also be deployed in a run time cluster. They are typically front-ended by a load balancer to route traffic.

- **State replication and routing**

  Oracle WebLogic Server can be configured for replicating the state of stateful applications. It does so by maintaining a replica of the state information on a different Managed Server, which is a cluster member. Oracle Fusion Middleware components, such as ADF and WebCenter, which are stateful, leverage this feature to ensure seamless failover to other members of the cluster.

  System components, such as Oracle Internet Directory, Oracle HTTP Server, Oracle Web Cache are stateless.

  Some Oracle Fusion Middleware components, which have part of the functionality implemented in C, such as Oracle Forms and Oracle Reports, are stateful and do not have state replication capabilities. Please refer to following paragraph for information about failover of these components.

- **Failover**

  Typically, a Managed Server running Oracle Fusion Middleware Java EE components has a Web server, such as Oracle HTTP Server, clustered in front of it. The Web server proxy plugin (mod_wl_ohs) is aware of the run time availability of the different Managed Servers, as well as the location of the Managed Server on which the state replica is maintained. If the primary Managed Server becomes unavailable, the plugin routes the request to the server where the application is available. If stateful applications, such as Oracle ADF and Oracle WebCenter, the location of the replica is also taken in to account while routing to the new Managed Server.

  For stateless system components, their multiple instances are deployed as a runtime cluster, behind a load balancer. The load balancer is configured to do a periodic health check of the component instances. If an instance becomes unavailable, the load balancer routes the subsequent requests to anther available instance, and the failover is seamless.

  For stateful components, which have parts based on C, and do not have state replication, sticky routing ensures that the subsequent requests go to the cluster member where the state was initially established. This is ensured by a Web server proxy plugin, as well as the Java EE parts of the components. If failure of the component instance occurs, subsequent requests are routed to another available member in the cluster. Though the state information is lost, and the user is required to recreate the session.

Some of the internal implementation of components use EJBs. EJB failover is seamlessly handled by replica aware WebLogic Server stubs.

Where needed, components are JTA compliant and data consistency is preserved in case of failover.

Singleton services leverage built-in failover capabilities, such as singleton SOA adapters, or use the underlying WebLogic Server infrastructure, such as Whole Server Migration.

- **Server Migration**

  Oracle Fusion Middleware components, such as SOA, which uses pinned services, such as JMS and JTA, leverage WebLogic Server capabilities to provide failover an automatic restart on a different cluster member.

- **Integrated High Availability**

  Oracle Fusion Middleware has a comprehensive feature set around load balancing and failover to leverage availability and scalability of Oracle RAC databases. All Oracle Fusion Middleware components have built-in protection against loss of service, data or transactions as a result of Oracle RAC instance unavailability due to planned or unplanned downtime. This is achieved by using Oracle WebLogic Server multi data sources. Additionally, components have proper exception handling and configurable retry logic for seamless failover of in-flight transactions at the time of failure.

  For XA compliant applications, such as Oracle SOA components, WebLogic server acts as a Transaction coordinator and ensures that all branches of a transaction are pinned to one of the Oracle RAC instances.

  In case of a Managed Server failure the transaction service is automatically migrated over to another node in the cluster and performs the transaction recovery.

  For communication between Web servers and application servers, proxy plugin has a built-in load balancing and failover capability to seamlessly reroute client requests to an available cluster member.

- **Rolling Patching**

  Oracle WebLogic Server allows for rolling patching where a minor maintenance patch can be applied to the product binaries in a rolling fashion without having to shut down the entire cluster.

  During the rolling patching of a cluster, each server in the cluster is individually patched and restarted while the other servers in the cluster continue to host your application. You can also uninstall a patch, maintenance pack, or minor release in a rolling fashion.

- **Configuration Management**

  Most of the Oracle Fusion Middleware component configuration can done at the cluster level. Oracle Fusion Middleware uses WebLogic Server's cluster wide configuration capabilities for server configuration, such as data sources, EJBs, and JMS, as well as component application artifacts, and ADF and WebCenter custom applications.

Additional application level components are stored in a central MDS repository which is available to all members of the cluster. This includes component level configuration for components, such as Oracle SOA, Oracle WebCenter, as well as application artifacts, such as SOA composites.

- **Backup and Recovery**

  Oracle Fusion Middleware backup and recovery is a simple solution based on file system copy for Middle-tier components. RMAN is used for Oracle databases. There is also support for online backups. With Oracle Fusion Middleware, you can integrate with existing backup and recovery tools, or use scheduled backup tasks through oracle Fusion Middleware Enterprise Manager or cron jobs.

2.3.2.1 Server Load Balancing

Typically, Oracle Fusion Middleware high availability deployments are front ended by a load balancer which can be configured to distributed incoming requests using various algorigthms.

Oracle Fusion Middleware also has built-in load balancing capabilities for intra component interaction. For example, Web server to application server, or application server to database server.

Oracle Fusion Middleware 11*g* does not provide external load balancers. To ensure that your external load balancer is compatible with Oracle Fusion Middleware, check that your external load balancer meets the requirements listed in Table 2–1.

You may not need to meet all of the requirements listed in the table. The requirements for external load balancers depend on the topology you are considering, and on the Oracle Fusion Middleware components you are load balancing.

Table 2–1 External Load Balancer Requirements

Requirement	Description
Virtual servers and port configuration	Configure virtual server names and ports on your external load balancer. The virtual server names and ports must meet the following requirements: ■ The load balancer should allow configuration of multiple virtual servers. For each virtual server, the load balancer should allow configuration of traffic management on more than one port. For example, for Oracle Fusion Middleware Identity Management, the load balancer needs to be configured with a virtual server and port for HTTP / HTTPS traffic, and separate virtual servers and ports for LDAP and LDAPS traffic. ■ The virtual server names must be associated with IP addresses and be part of your DNS. Clients must be able to access the external load balancer through the virtual server names.
Persistence/stickiness	Some Oracle Fusion Middleware components use persistence or stickiness in an external load balancer. If your external load balancer does not allow you to set cookie persistence at the URI level, set the cookie persistence for all HTTP traffic. In either case, set the cookie to expire when the browser session expires. Refer to your external load balancer documentation for details.

Table 2–1 (Cont.) External Load Balancer Requirements

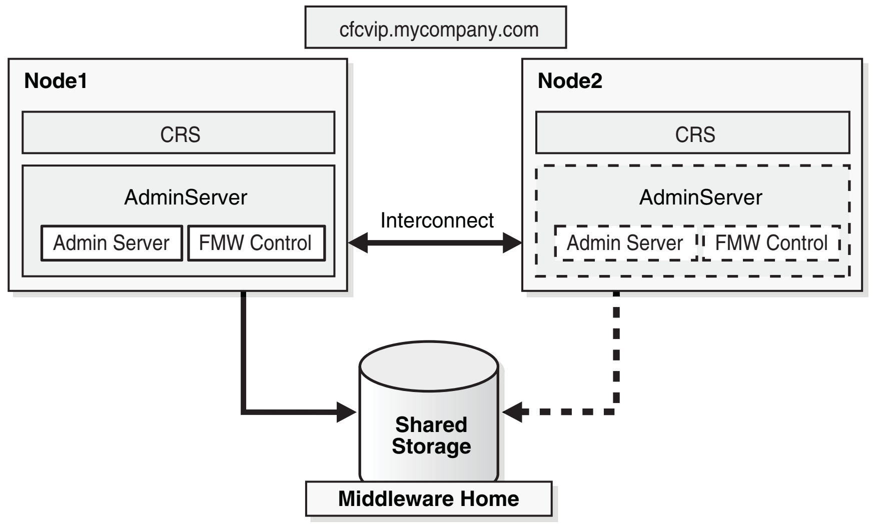
Requirement	Description
Resource monitoring/port monitoring/process failure detection	Configure the external load balancer to detect service and node failures (through notification or some other means) and to stop directing traffic to the failed node. Your external load balancer may have the ability to automatically detect failures. For example, for Oracle Fusion Middleware Identity Management, the external load balancer should monitor Oracle Internet Directory, Oracle Fusion Middleware Single Sign-On, and Oracle Delegated Administration Services. To monitor these components, set up monitors for the following protocols: ■ LDAP and LDAPS listen ports ■ HTTP and HTTPS listen ports (depending on the deployment type) These monitors use the respective protocols to monitor the services, meaning they use LDAP for the LDAP port, LDAP over SSL for the LDAP SSL port, and HTTP/HTTPS for the Oracle HTTP Server port. If your external load balancer does not offer these monitors, consult your external load balancer documentation for the best method of configuring it to automatically stop routing incoming requests to a service that is unavailable.
Network Address Translation (NAT)	The load balancer should have the capability to perform network address translation (NAT) for traffic being routed from clients to the Oracle Fusion Middleware nodes.
Fault tolerant mode	Oracle highly recommends configuring the load balancer to be in fault-tolerant mode, otherwise the load balancer becomes a single point of failure for the system. This rules out most software load balancers that are based on a single process/interceptor as reliable solutions.
Other	Oracle highly recommends configuring the load balancer virtual server to return immediately to the calling client when the back-end services to which it forwards traffic are unavailable. This configuration is preferred over the client disconnecting on its own after a timeout, based on the TCP/IP settings on the client machine.

2.3.3 Active-Passive Deployment

Oracle Fusion Middleware provides an active-passive model for all its components using Oracle Fusion Middleware Cold Failover Clusters. In an Oracle Fusion Middleware Cold Failover Cluster configuration, two or more application server instances are configured to serve the same application workload but only one is active at any particular time.

Figure 2–4 illustrates an example active-passive deployment.

Figure 2–4 Example Active-Passive Cold Failover Cluster Deployment



Example Active-Passive Cold Failover Cluster Deployment

****

In Figure 2–4, the Administration Server runs on a two-node hardware cluster: Node 1 and Node 2. The Administration Server is listening on the Virtual IP or hostname. The Middleware Home and the domain directory is on a shared disk that is mounted on Node 1 or Node 2 at any given point. Both the Middleware home and the domain directory should be on the same shared disk or shared disks that can fail over together. If an enterprise has multiple Fusion Middleware domains for multiple applications or environments, this topology is well suited for Administration Server high availability. A single hardware cluster can be deployed to host these multiple Administration Servers. Each Administration Server can use its own virtual IP and set of shared disks to provide high availability of domain services.

For details about active-passive concepts, as well as configuration procedures for Oracle Cold Failover Clusters, see Chapter 10, "Active-Passive Topologies for Oracle Fusion Middleware High Availability."

2.3.4 About Active-Active and Active-Passive Solutions

Oracle recommends using active-active solutions when possible. This is the primary recommendation for maximum availability. Active-active solutions provide faster failover, scalability, and protection against node, instance and component failures. In addition, active-active solutions also offer transparent failover and the easy addition of resources for scaling up vertically and horizontally.

Scalability requirements are an important consideration when designing an Oracle Fusion Middleware high availability solution. Active-active solutions scale up (vertically) by adding more instances or components inside the same node.

Adding multiple redundant services in the same node improves the availability of a system, but only against instance failures and not against node failures. In addition, as described in Oracle Fusion Middleware provides death detection and automatic restart of components. Active-active high availability solutions include multiple active instances installed on different nodes. As a result, when a node is completely lost other instances are available to keep the system going, uninterrupted. Using multiple instances in different nodes provides what is known as horizontal scalability, or scaling out.

Active-active solutions require logic to load balance and failover requests among the active Oracle Fusion Middleware instances. Load balancing is provided by distributing the incoming requests to different service providers. Failover is achieved by detecting any failures in this service providers and re-routing the request to other available service providers. This logic is implemented in different ways:

- Direct implementation: The logic is implemented directly by the client making a request to the system. For example, a JDBC client implements load balancing and failover logic to connect to multiple instances of an Oracle database (Real Application Cluster). It can be implemented by an external hardware load balancer.
- Third party implementation: The logic is provided by third party components that intercept the client requests and distribute the load to the multiple Oracle Instances. When several Oracle Instances are grouped to work together, they present themselves as a single virtual entry point to the system, which hides the multiple instance configuration. External load balancers can send requests to any application server instance in a cluster, as any instance can service any request.

Unlike the scalability properties of an active-active configuration, in active-passive configurations the passive component is used only when the active component fails. In active-active solutions all instances handle requests concurrently. As a result, active-active systems provide higher transparency and have greater scalability than an active-passive system.

Active-passive solutions are limited to vertical scalability, with just one node remaining active. Active-passive solutions also have an implicit failover time when failure occurs. This failover time is usually determined by the time it takes to restart the components in the node that becomes active post-failure. However, the operational and licensing costs of an active-passive model are lower than that of an active-active deployment.

There are situations where active-passive solutions are appropriate. Oracle recommends using hardware-cluster based active-passive solutions in the following scenarios:

- The licensing, management and the total cost of ownership of a load balancer, excludes an active-active solution, particularly if there is a hardware cluster available. Hardware clusters require two nodes, a switch for connecting the nodes, shared storage that can be reached from both hardware nodes.
- You may have concurrency issues with Singleton services. With Singleton services, only one active instance can exit at runtime. Singleton services may be important in relation to other components. They typically provide basic services to multiple components, so if the are not available, then many other services or processes may not be available. here are some issues to consider when protecting Singleton services
  - Recovery Time: Singleton services or components can not run in active-active configurations. Client requests are not transparently load balanced to multiple instances of the service. This implies that, in case of a failure, there is an implicit recovery time. This recovery time varies depending on the type of Singleton protection model you adopt.
  - Reliability in failure detection: The system must prevent "false positives," or, at minimum, they the system should analyze their affect on different singleton services. Most singleton services access data repositories that may or may not be designed for concurrent access. If a singleton service is reported as "dead" and the system decides to start a new instance, a "split brain" scenarios could arise. The dead service must be analyzed for implications of this concurrency

and how likely a false positive is to happen based on the failure detection mechanism.

– Consistency in service across restarts: Singleton components must provide consistent service after a failover. These components must maintain the same behavior after recovering from a crash. The configuration and persistent repositories used by the service must be available during failures. Also, start dependencies must be accounted for upon failover. For example, if the singleton service needs to be restarted it may have start dependencies on other services and these must be preserved.

– Cost (hardware/software resources required) Different protection mechanisms may require a pure software based solution, or a hardware based solution with the implicit costs.

– Installation/Configuration/Management: The different protection mechanisms for singleton services should not add complexity to the system

– Maintenance (patches, upgrades): Protection models for singleton services should enable easily and allow minimum downtime for applying patches and upgrades.

Based on these criteria, different solutions may be used for Oracle Fusion Middleware Singleton Components depending on the pertaining requirements to the specific singleton service:

- Cold Failover Cluster Solution: This solution requires a shared storage and a connection to detect hardware failures. The re-routing of requests by migration of the VHN through the failover procedure does not need intelligence on clients or load balancers.
- Whole Server Migration- This is the process of moving a clustered WebLogic Server instance or a component running on a clustered instance elsewhere in the event of failure. In the case of whole server migration, the server instance is migrated to a different physical machine upon failure. In the case of service-level migration, the services are moved to a different server instance within the cluster.
- Custom active-passive models based on software blocking mechanism: This logic is included in a component to prevent other instances of the same component from becoming active at the same time. Typical solutions use locks in a database, or custom in-memory active notifications that prevent concurrency.

In many cases, reliability of failure detection is an important factor for adopting one solution over another. This is especially true when concurrency can cause corruption of resources that are used by the singleton service. Typically, files may be written concurrently by different active instances.

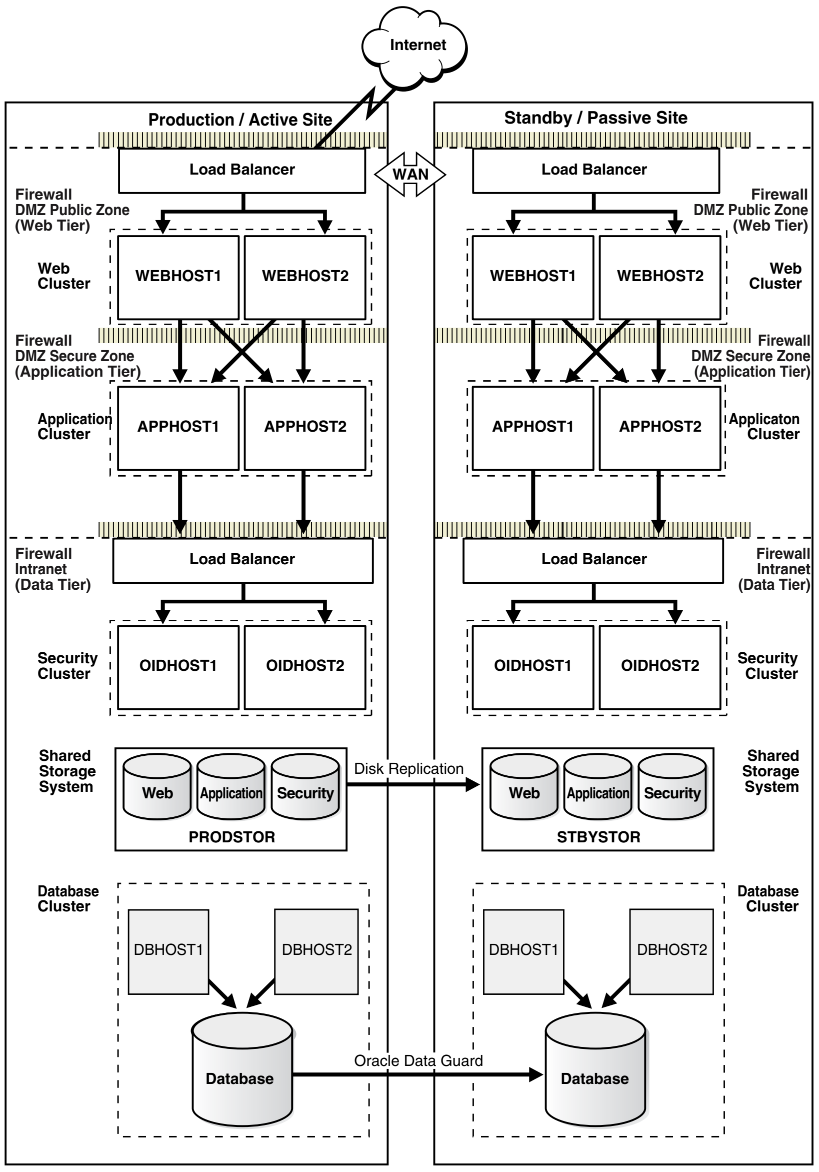
You may adopt other solutions for different components for the issues explained in this section.

2.3.5 Disaster Recovery

Figure 2–5 illustrates an Oracle fusion Middleware architecture configured for Disaster Recovery. For Oracle Fusion Middleware Product Binaries, Configuration and Metadata Files, the Disk Replication-based solution involves deploying Oracle Fusion Middleware on NAS/SAN devices. Product binaries and configuration data, stored in Oracle Homes, are stored on NAS/SAN devices using mounted locations from host machines. In addition, Disk Replication technologies are used to replicate product binaries and configuration from a primary site disk to a secondary site disk on a periodic basis. Secondary site servers are also mounted to the disks on the secondary site. If a failure or planned outage of the primary site occurs, replication to the secondary site is stopped. The services and applications are subsequently started on the secondary site. The network traffic is then be routed to the secondary site.

For Oracle Database content, because of its superior level of protection and high availability, Oracle Data Guard is the recommended solution for disaster protection of Oracle Databases. This includes the databases used for Oracle Fusion Middleware Repositories, as well as customer data.

Figure 2–5 Production and Standby Site for Oracle Fusion Middleware Disaster Recovery Topology



2.4 Protection from Planned and Unplanned Down Time

The following tables list possible planned and unplanned downtime and suggested solutions for these downtime possibilities. Table 2–2 describes planned downtime:

Table 2–2 Planned Down Time Solutions

Operations	Solutions
Deploying and redeploying applications	Hot Deployment
Patching	Rolling Patching
Configuration Changes	Online configuration Changes
	Change Notification
	Batching of changes
	Deferred Activation
Scalability and Topology Extensions	Cluster Scale-Out

Table 2–3 describes unplanned downtime:

Table 2–3 Unplanned Down Time Solutions

Failures	Solutions
Software Failure	Death Detection and restart using Node Manager for Java EE and OPMN for system components.
	Server Clusters & Load Balancing
	Cold Failover Clusters
	Server Migration
	Service Migration
	State Replication and Replica aware Stubs
Hardware Failure	Server Clusters & Load Balancing
	Server Migration
	Clusterware Integration
Data Failure	Backup and Recovery
Human Error	
Site Disaster	Oracle Fusion Middleware Disaster Recovery Solution

3 High Availability for WebLogic Server

This chapter describes the Oracle WebLogic Server high availability capabilities used to provide Oracle Fusion Middleware high availability.

For complete documentation of Oracle WebLogic Server clustering, see *Oracle Fusion Middleware Using Clusters for Oracle WebLogic Server*.

3.1 What Is a WebLogic Server Cluster?

A WebLogic Server cluster consists of multiple WebLogic Server server instances running simultaneously and working together to provide increased scalability and reliability. A cluster appears to clients to be a single WebLogic Server instance. The server instances that constitute a cluster can run on the same machine, or be located on different machines. You can increase a cluster's capacity by adding additional server instances to the cluster on an existing machine, or you can add machines to the cluster to host the incremental server instances. Each server instance in a cluster must run the same version of WebLogic Server.

3.2 WebLogic Server Clusters and WebLogic Server Domains

A cluster is part of a particular WebLogic Server domain. A domain is an interrelated set of WebLogic Server resources that are managed as a unit. A domain includes one or more WebLogic Server instances, which can be clustered, non-clustered, or a combination of clustered and non-clustered instances. A domain can include multiple clusters. A domain also contains the application components deployed in the domain, and the resources and services required by those application components and the server instances in the domain. Examples of the resources and services used by applications and server instances include machine definitions, optional network channels, connectors, and startup classes.

In each domain, one WebLogic Server instance acts as the Administration Server—the server instance which configures, manages, and monitors all other server instances and resources in the domain. Each Administration Server manages one domain only. If a domain contains multiple clusters, each cluster in the domain has the same Administration Server.

All server instances in a cluster must reside in the same domain; you cannot "split" a cluster over multiple domains. Similarly, you cannot share a configured resource or subsystem between domains. For example, if you create a JDBC connection pool in one domain, you cannot use it with a server instance or cluster in another domain. (Instead, you must create a similar connection pool in the second domain.)

Clustered WebLogic Server instances behave similarly to non-clustered instances, except that they provide failover and load balancing. The process and tools used to configure clustered WebLogic Server instances are the same as those used to configure non-clustered instances. However, to achieve the load balancing and failover benefits that clustering enables, you must adhere to certain guidelines for cluster configuration.

3.3 Benefits of Clustering

A WebLogic Server cluster provides these benefits:

- Scalability

  The capacity of an application deployed on a WebLogic Server cluster can be increased dynamically to meet demand. You can add server instances to a cluster without interruption of service—the application continues to run without impact to clients and end users.

- High Availability

  In a WebLogic Server cluster, application processing can continue when a server instance fails. You “cluster” application components by deploying them on multiple server instances in the cluster—so, if a server instance on which a component is running fails, another server instance on which that component is deployed can continue application processing.

The choice to cluster WebLogic Server instances is transparent to application developers and clients. However, understanding the technical infrastructure that enables clustering will help programmers and administrators maximize the scalability and availability of their applications.

3.4 Key Capabilities of a Cluster

The following sections define, in non-technical terms, the key clustering capabilities that enable scalability and high availability.

3.4.1 Application Failover

Simply put, failover means that when an application component (typically referred to as an "object" in the following sections) doing a particular "job"—some set of processing tasks—becomes unavailable for any reason, a copy of the failed object finishes the job.

For the new object to be able to take over for the failed object:

- There must be a copy of the failed object available to take over the job.
- There must be information, available to other objects and the program that manages failover, defining the location and operational status of all objects—so that it can be determined that the first object failed before finishing its job.
- There must be information, available to other objects and the program that manages failover, about the progress of jobs in process—so that an object taking over an interrupted job knows how much of the job was completed before the first object failed, for example, what data has been changed, and what steps in the process were completed.

WebLogic Server uses standards-based communication techniques and facilities—including IP sockets and the Java Naming and Directory Interface (JNDI)—to share and maintain information about the availability of objects in a cluster. These techniques allow WebLogic Server to determine that an object stopped before finishing its job, and where there is a copy of the object to complete the job that was interrupted.

Information about what has been done on a job is called state. WebLogic Server maintains information about state using techniques called *session replication* and *replica-aware stubs*. When a particular object unexpectedly stops doing its job, replication techniques enable a copy of the object pick up where the failed object stopped, and finish the job.

3.4.2 Migration

WebLogic Server supports automatic and manual migration of a clustered server instance from one machine to another. A Managed Server that can be migrated is referred to as a migratable server. This feature is designed for environments with requirements for high availability. The server migration capability is useful for:

- Ensuring uninterrupted availability of singleton services—services that must run on only a single server instance at any given time, such as JMS and the JTA transaction recovery system, when the hosting server instance fails. A Managed Server configured for automatic migration will be automatically migrated to another machine in the even of failure.
- Easing the process of relocating a Managed Server, and all the services it hosts, as part of a planned system administration process. An administrator can initiate the migration of a Managed Server from the Administration Console or command line.

The server migration process relocates a Managed Server in its entirety—including IP addresses and hosted applications—to on of a predefined set of available host machines.

3.4.3 Load Balancing

Load balancing is the even distribution of jobs and associated communications across the computing and networking resources in your environment. For load balancing to occur:

- There must be multiple copies of an object that can do a particular job.
- Information about the location and operational status of all objects must be available.

  WebLogic Server allows objects to be clustered—deployed on multiple server instances—so that there are alternative objects to do the same job. WebLogic Server shares and maintains the availability and location of deployed objects using unicast, IP sockets, and JNDI.

3.5 Types of Objects That Can Be Clustered

A clustered application or application component is one that is available on multiple WebLogic Server instances in a cluster. If an object is clustered, failover and load balancing for that object is available. Deploy objects homogeneously—to every server instance in your cluster—to simplify cluster administration, maintenance, and troubleshooting.

Web applications can consist of different types of objects, including Enterprise Java Beans (EJBs), servlets, and Java Server Pages (JSPs). Each object type has a unique set of behaviors related to control, invocation, and how it functions within an application. For this reason, the methods that WebLogic Server uses to support clustering—and hence to provide load balancing and failover—can vary for different types of objects. The following types of objects can be clustered in a WebLogic Server deployment:

- Servlets
- JSPs
- EJBs
- Remote Method Invocation (RMI) objects
- Java Messaging Service (JMS) destinations
- Java Database Connectivity (JDBC) connections

Different object types can have certain behaviors in common. When this is the case, the clustering support and implementation considerations for those similar object types may be same. In the sections that follow, explanations and instructions for the following types of objects are generally combined:

- Servlets and JSPs
- EJBs and RMI objects

3.6 Communications in a Cluster

WebLogic Server instances in a cluster communicate with one another using two basic network technologies:

- IP sockets, which are the conduits for peer-to-peer communication between clustered server instances.
- IP unicast or multicast, which server instances use to broadcast availability of services and heartbeats that indicate continued availability. When creating a new cluster, it is recommended that you use unicast for messaging within a cluster. For backward compatibility with previous versions, WebLogic Server you must use multicast for communications between clusters.

3.7 Cluster-Wide JNDI Naming Service

Clients of a non-clustered WebLogic Server server instance access objects and services by using a JNDI-compliant naming service. The JNDI naming service contains a list of the public services that the server instance offers, organized in a tree structure. A WebLogic Server instance offers a new service by binding into the JNDI tree a name that represents the service. Clients obtain the service by connecting to the server instance and looking up the bound name of the service.

Server instances in a cluster utilize a cluster-wide JNDI tree. A cluster-wide JNDI tree is similar to a single server instance JNDI tree, insofar as the tree contains a list of available services. In addition to storing the names of local services, however, the cluster-wide JNDI tree stores the services offered by clustered objects (EJBs and RMI classes) from other server instances in the cluster.

Each WebLogic Server instance in a cluster creates and maintains a local copy of the logical cluster-wide JNDI tree. Creation of a cluster-wide JNDI tree begins with the local JNDI tree bindings of each server instance. As a server instance boots (or as new services are dynamically deployed to a running server instance), the server instance first binds the implementations of those services to the local JNDI tree. The implementation is bound into the JNDI tree only if no other service of the same name exists.

Once the server instance successfully binds a service into the local JNDI tree, additional steps are performed for clustered objects that use replica-aware stubs. After binding the clustered object's implementation into the local JNDI tree, the server instance sends the object's stub to other members of the cluster. Other members of the cluster monitor the multicast or unicast address to detect when remote server instances offer new services.

3.8 Failover and Replication in a Cluster

In order for a cluster to provide high availability it must be able to recover from service failures.

WebLogic Server instances in a cluster detect failures of their peer server instances by monitoring:

- Socket connections to a peer server

  WebLogic Server instances monitor the use of IP sockets between peer server instances as an immediate method of detecting failures. If a server connects to one of its peers in a cluster and begins transmitting data over a socket, an unexpected closure of that socket causes the peer server to be marked as "failed," and its associated services are removed from the JNDI naming tree.

- Regular server heartbeat messages

  If clustered server instances do not have opened sockets for peer-to-peer communication, failed servers may also be detected via the WebLogic Server heartbeat. All server instances in a cluster use multicast or unicast to broadcast regular server heartbeat messages to other members of the cluster.

  Each heartbeat message contains data that uniquely identifies the server that sends the message. Servers broadcast their heartbeat messages at regular intervals of 10 seconds. In turn, each server in a cluster monitors the multicast or unicast address to ensure that all peer servers' heartbeat messages are being sent.

  If a server monitoring the multicast or unicast address misses three heartbeats from a peer server (i.e., if it does not receive a heartbeat from the server for 30 seconds or longer), the monitoring server marks the peer server as "failed." It then updates its local JNDI tree, if necessary, to retract the services that were hosted on the failed server.

3.8.1 Session Replication

User session data can be stored in two standard ways in a Java EE application: stateful session EJBs or HTTP sessions. By themselves, they are rarely a impact cluster scalability. However, when coupled with a session replication mechanism required to provide high-availability, bottlenecks are introduced. If a Java EE application has Web and EJB components, you should store user session data in HTTP sessions:

- HTTP session management provides more options for handling fail-over, such as replication, a shared database or file.
- Superior scalability.
- Replication of the HTTP session state occurs outside of any transactions. Stateful session bean replication occurs in a transaction which is more resource intensive.
- The HTTP session replication mechanism is more sophisticated and provides optimizations a wider variety of situations than stateful session bean replication.

3.9 Whole Server Migration

In a WebLogic Server cluster, most services are deployed homogeneously on all server instances in the cluster, enabling transparent failover from one server to another. In contrast, “pinned services” such as JMS and the JTA transaction recovery system are targeted at individual server instances within a cluster—for these services, WebLogic Server supports failure recovery with migration, as opposed to failover.

Migration in WebLogic Server is the process of moving a clustered WebLogic Server instance or a component running on a clustered instance elsewhere in the event of failure. In the case of whole server migration, the server instance is migrated to a different physical machine upon failure. In the case of service-level migration, the services are moved to a different server instance within the cluster.

WebLogic Server provides a feature for making JMS and the JTA transaction system highly available: migratable servers. Migratable servers provide for both automatic and manual migration at the server-level, rather than the service level.

Server migration is not supported on Windows. When a migratable server becomes unavailable for any reason, for example, if it hangs, loses network connectivity, or its host machine fails—migration is automatic. Upon failure, a migratable server is automatically restarted on the same machine if possible. If the migratable server cannot be restarted on the machine where it failed, it is migrated to another machine. In addition, an administrator can manually initiate migration of a server instance.

3.9.1 Node Manager's Role in Whole Server Migration

The use of Node Manager is required for server migration—it must run on each machine that hosts, or is intended to host.

Node Manager supports server migration in these ways:

- Node Manager must be used for initial startup of migratable servers.

  When you initiate the startup of a Managed Server from the Administration Console, the Administration Server uses Node Manager to start up the server instance. You can also invoke Node Manager to start the server instance using the stand-alone Node Manager client; however, the Administration Server must be available so that the Managed Server can obtain its configuration.

> **Note:** Migration of a server instance that not initially started with Node Manager will fail.

- Node Manager must be used for suspend, shutdown, or force shutdown of migratable servers.
- Node Manager tries to restart a migratable server whose lease has expired on the machine where it was running at the time of failure.

  Node Manager performs the steps in the server migrate process by running customizable shell scripts, provided with WebLogic Server, that start, restart and stop servers; migrate IP addresses; and mount and unmount disks. The scripts are available for Solaris and Linux.

  – In an automatic migration, the cluster master invokes Node Manager to perform the migration.

  – In a manual migration, the Administration Server invokes Node Manager to perform the migration.

3.9.2 Server Migration Processes and Communications

The sections that follow describe key processes in a cluster that contains migratable servers:

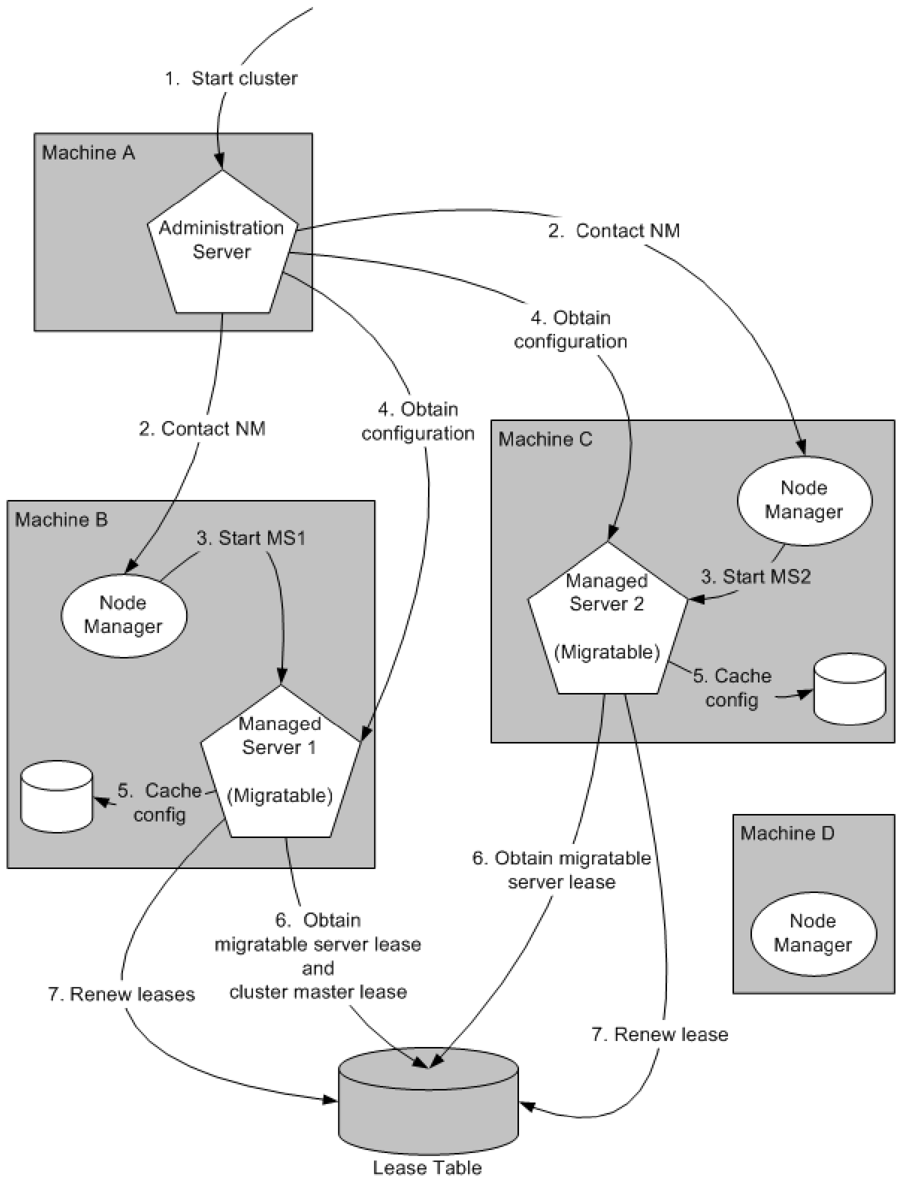
- Section 3.9.2.1, "Startup Process in a Cluster with Migratable Servers"
- Section 3.9.2.2, "Automatic Whole Server Migration Process"
- Section 3.9.2.3, "Manual Whole Server Migration Process"
- Section 3.9.2.4, "Administration Server's Role in Whole Server Migration"
- Section 3.9.2.5, "Migratable Server Behavior in a Cluster"
- Section 3.9.2.6, "Node Manager's Role in Whole Server Migration"
- Section 3.9.2.7, "Cluster Master's Role in Whole Server Migration"

3.9.2.1 Startup Process in a Cluster with Migratable Servers

Figure 3–1 illustrates the processing and communications that occur during startup of a cluster that contains migratable servers.

The example cluster contains two Managed Servers, both of which are migratable. The Administration Server and the two Managed Servers each run on different machines. A fourth machine is available as a backup—in the event that one of the migratable servers fails. Node Manager is running on the backup machine and on each machine with a running migratable server.

Figure 3–1 Startup of Cluster with Migratable Servers



****

These are the key steps that occur during startup of the cluster illustrated in Figure 3–1:

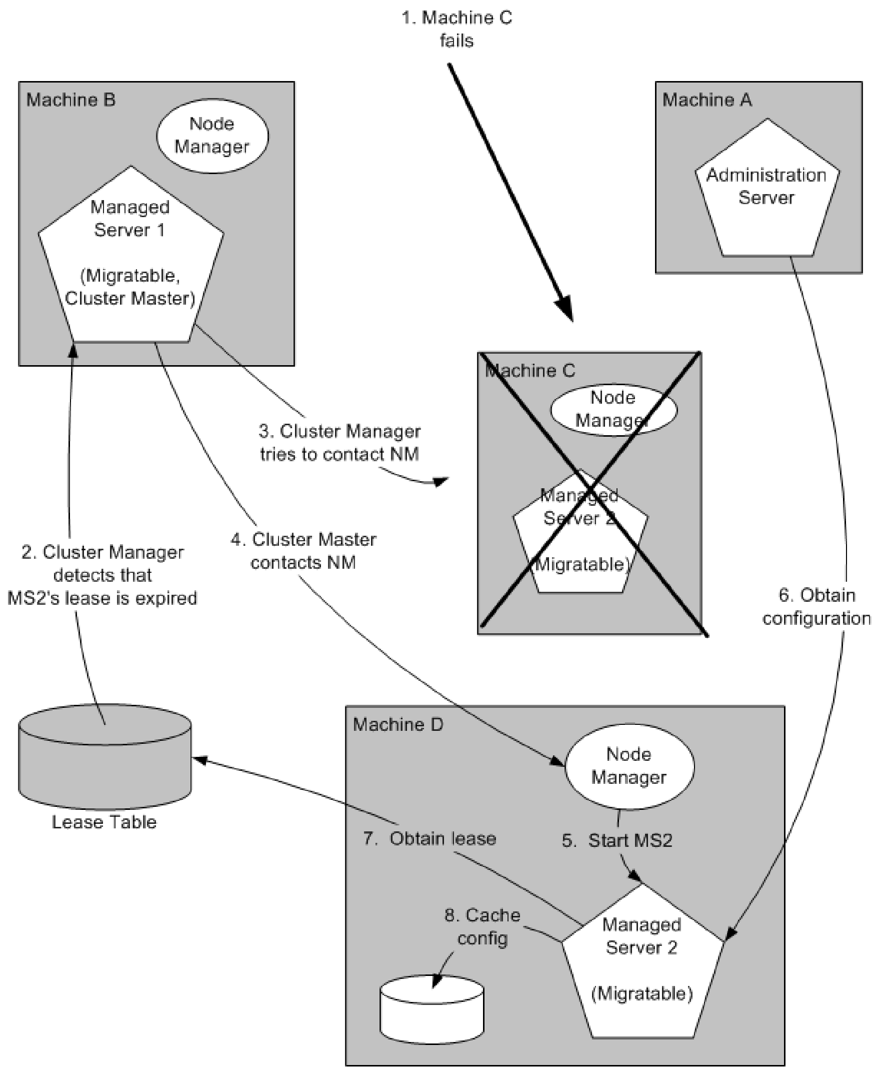
1. The administrator starts up the cluster.
2. The Administration Server invokes Node Manager on Machines B and C to start Managed Servers 1 and 2, respectively. See Section 3.9.2.4, "Administration Server's Role in Whole Server Migration."
3. The Node Manager on each machine starts up the Managed Server that runs there. See Section 3.9.2.6, "Node Manager's Role in Whole Server Migration."
4. Managed Servers 1 and 2 contact the Administration Server for their configuration. See Section 3.9.2.5, "Migratable Server Behavior in a Cluster."

5. Managed Servers 1 and 2 cache the configuration they started up.
6. Managed Servers 1 and 2 each obtain a migratable server lease in the lease table. Because Managed Server 1 starts up first, it also obtains a cluster master lease. See Section 3.9.2.7, "Cluster Master's Role in Whole Server Migration."
7. Managed Server 1 and 2 periodically renew their leases in the lease table, proving their health and liveness.

3.9.2.2 Automatic Whole Server Migration Process

Figure 3–2 illustrates the automatic migration process after the failure of the machine hosting Managed Server 2.

Figure 3–2 Automatic Migration of a Failed Server



1. Machine C, which hosts Managed Server 2, fails.
2. Upon its next periodic review of the lease table, the cluster master detects that Managed Server 2's lease has expired. See Section 3.9.2.7, "Cluster Master's Role in Whole Server Migration."

3. The cluster master tries to contact Node Manager on Machine C to restart Managed Server 2, but fails, because Machine C is unreachable.

> **Note:** If the Managed Server 2's lease had expired because it was hung, and Machine C was reachable, the cluster master would use Node Manager to restart Managed Server 2 on Machine C.

4. The cluster master contacts Node Manager on Machine D, which is configured as an available host for migratable servers in the cluster.
5. Node Manager on Machine D starts Managed Server 2. See Section 3.9.2.6, "Node Manager's Role in Whole Server Migration."
6. Managed Server 2 starts up and contacts the Administration Server to obtain its configuration.
7. Managed Server 2 caches the configuration it started up with.
8. Managed Server 2 obtains a migratable server lease.

During migration, the clients of the Managed Server that is migrating may experience a brief interruption in service; it may be necessary to reconnect. On Solaris and Linux operating systems, this can be done using `ifconfig` command. The clients of a migrated server do not need to know the particular machine to which it has migrated.

When a machine that previously hosted a server instance that was migrated becomes available again, the reversal of the migration process—migrating the server instance back to its original host machine—is known as *failback*. WebLogic Server does not automate the process of failback. An administrator can accomplish failback by manually restoring the server instance to its original host.

The general procedures for restoring a server to its original host are as follows:

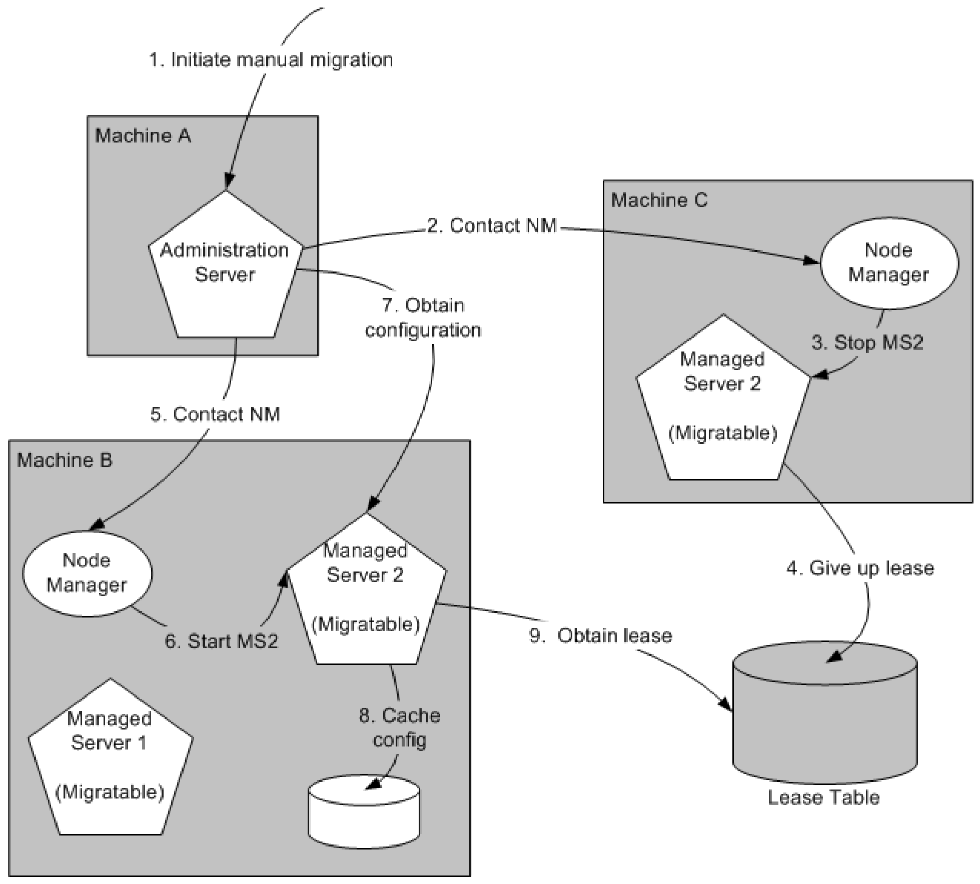
- Gracefully shutdown the new instance of the server
- After you have restarted the failed machine, restart Node Manager and the managed server.

The exact procedures you will follow depend on your server and network environment.

3.9.2.3 Manual Whole Server Migration Process

Figure 3–3 illustrates what happens when an administrator manually migrates a migratable server.

Figure 3–3 Manual Whole Server Migration



1. An administrator uses the Administration Console to initiate the migration of Managed Server 2 from Machine C to Machine B.
2. The Administration Server contacts Node Manager on Machine C. See Section 3.9.2.4, "Administration Server's Role in Whole Server Migration."
3. Node Manager on Machine C stops Managed Server 2.
4. Managed Server 2 removes its row from the lease table.
5. The Administration Server invokes Node Manager on Machine B.
6. Node Manager on Machine B starts Managed Server 2.
7. Managed Server 2 obtains its configuration from the Administration Server.
8. Managed Server 2 caches the configuration it started up with.
9. Managed Server 2 adds a row to the lease table.

3.9.2.4 Administration Server's Role in Whole Server Migration

In a cluster that contains migratable servers, the Administration Server:

- Invokes Node Manager, on each machine that hosts cluster members, to start up the migratable servers. This is a prerequisite for server migratability—if a server instance was not initially started by Node Manager, it cannot be migrated.
- Invokes Node Manager on each machine involved in a manual migration process to stop and start the migratable server.

- Invokes Node Manager on each machine that hosts cluster members to stop server instances during a normal shutdown. This is a prerequisite for server migratability—if a server instance is shut down directly, without using Node Manager, when the cluster master detects that the server instance is not running, it will call Node Manager to restart it.

In addition, the Administration Server provides its regular domain management functionality, persisting configuration updates issued by an administrator, and providing a run-time view of the domain, including the migratable servers it contains.

3.9.2.5 Migratable Server Behavior in a Cluster

A migratable server is a clustered Managed Server that has been configured as migratable. These are the key behaviors of a migratable server:

- If you are using a database to manage leasing information, during startup and restart by Node Manager, a migratable server adds a row to the lease table. The row for a migratable server contains a timestamp, and the machine where it is running.
- When using a database to manage leasing information, a migratable server adds a row to the database as a result of startup, it tries to take on the role of cluster master, and succeeds if it is the first server instance to join the cluster.
- Periodically, the server renews its "lease" by updating the timestamp in the lease table.

  By default a migratable server renews its lease every 30,000 milliseconds—the product of two configurable `ServerMBean` properties:

  – `HealthCheckIntervalMillis`, which by default is 10,000.

  – `HealthCheckPeriodsUntilFencing`, which by default is 3.
- If a migratable server fails to reach the lease table and renew its lease before the lease expires, it terminates as quickly as possible using a Java `System.exit`—in this case, the lease table still contains a row for that server instance. For information about how this relates to automatic migration, see Section 3.9.2.7, "Cluster Master's Role in Whole Server Migration."
- During operation, a migratable server listens for heartbeats from the cluster master. When it detects that the cluster master is not sending heartbeats, it attempts to take over the role of cluster master, and succeeds if no other server instance has claimed that role.

3.9.2.6 Node Manager's Role in Whole Server Migration

The use of Node Manager is required for server migration—it must run on each machine that hosts, or is intended to host.

Node Manager supports server migration in these ways:

- Node Manager must be used for initial startup of migratable servers.

  When you initiate the startup of a Managed Server from the Administration Console, the Administration Server uses Node Manager to start up the server instance. You can also invoke Node Manager to start the server instance using the stand-alone Node Manager client; however, the Administration Server must be available so that the Managed Server can obtain its configuration.

Note: Migration of a server instance that not initially started with Node Manager will fail.

- Node Manager must be used for suspend, shutdown, or force shutdown of migratable servers.
- Node Manager tries to restart a migratable server whose lease has expired on the machine where it was running at the time of failure.

  Node Manager performs the steps in the server migrate process by running customizable shell scripts, provided with WebLogic Server, that start, restart and stop servers; migrate IP addresses; and mount and unmount disks. The scripts are available for Solaris and Linux.

  – In an automatic migration, the cluster master invokes Node Manager to perform the migration.

  – In a manual migration, the Administration Server invokes Node Manager to perform the migration.

3.9.2.7 Cluster Master's Role in Whole Server Migration

In a cluster that contains migratable servers, one server instance acts as the cluster master. Its role is to orchestrate the server migration process. Any server instance in the cluster can serve as the cluster master. When you start a cluster that contains migratable servers, the first server to join the cluster becomes the cluster master and starts up the cluster manager service. If a cluster does not include at least one migratable server, it does not require a cluster master, and the cluster master service does not start up. In the absence of a cluster master, migratable servers can continue to operate, but server migration is not possible. These are the key functions of the cluster master:

- Issues periodic heartbeats to the other servers in the cluster.
- Periodically reads the lease table to verify that each migratable server has a current lease. An expired lease indicates to the cluster master that the migratable server should be restarted.
- Upon determining that a migratable server's lease is expired, waits for period specified by the `FencingGracePeriodMillis` on the `ClusterMBean`, and then tries to invoke the Node Manager process on the machine that hosts the migratable server whose lease is expired, to restart the migratable server.
- If unable to restart a migratable server whose lease has expired on its current machine, the cluster master selects a target machine in this fashion:
- If you have configured a list of preferred destination machines for the migratable server, the cluster master chooses a machine on that list, in the order the machines are listed.
- Otherwise, the cluster master chooses a machine on the list of those configured as available for hosting migratable servers in the cluster.

  A list of machines that can host migratable servers can be configured at two levels: for the cluster as a whole, and for an individual migratable server. You can define a machine list at both levels. You must define a machine list at least one level.
- To accomplish the migration of a server instance to a new machine, the cluster master invokes the Node Manager process on the target machine to create a process for the server instance.

The time required to perform the migration depends on the server configuration and startup time.

- The maximum time taken for cluster master to restart the migratable server is (`HealthCheckPeriodsUntilFencing` * `HealthCheckIntervalMillis`) + `FencingGracePeriodMillis`.
- The total time before the server becomes available for client requests depends on the server startup time and the application deployment time.

3.10 JMS and JTA High Availability

You can configure JMS and JTA services for high availability by using migratable targets. A migratable target is a special target that can migrate from one server in a cluster to another. As such, a migratable target provides a way to group migratable services that should move together. When the migratable target is migrated, all services hosted by that target are migrated.

In order to configure a migratable JMS service for migration, it must be deployed to a migratable target. A migratable target specifies a set of servers that can host a target, and can optionally specify a user-preferred host for the services and an ordered list of candidate backup servers should the preferred server fail. Only one of these servers can host the migratable target at any one time.

Once a service is configured to use a migratable target, then the service is independent from the server member that is currently hosting it. For example, if a JMS server with a deployed JMS queue is configured to use a migratable target, then the queue is independent of when a specific server member is available. In other words, the queue is always available when the migratable target is hosted by any server in the cluster.

An administrator can manually migrate pinned migratable services from one server instance to another in the cluster, either in response to a server failure or as part of regularly scheduled maintenance. If you do not configure a migratable target in the cluster, migratable services can be migrated to any WebLogic Server instance in the cluster.

3.10.1 User-Preferred Servers and Candidate Servers

When deploying a JMS service to the migratable target, you can select a the user-preferred server (UPS) target to host the service. When configuring a migratable target, you can also specify constrained candidate servers (CCS) that can potentially host the service should the user-preferred server fail. If the migratable target does not specify a constrained candidate server, the JMS server can be migrated to any available server in the cluster.

WebLogic Server enables you to create separate migratable targets for JMS services. This allows you to always keep each service running on a different server in the cluster, if necessary. Conversely, you can configure the same selection of servers as the constrained candidate servers for both JTA and JMS, to ensure that the services remain co-located on the same server in the cluster.

3.11 Administration Server and Node Manager High Availability

The Administration Server is the WebLogic Server instance that configures and manages the WebLogic Server instances in its domain.

A domain can include multiple WebLogic Server clusters and non-clustered WebLogic Server instances. Strictly speaking, a domain could consist of only one WebLogic

Server instance—however, in that case that sole server instance would be an Administration Server, because each domain must have exactly one Administration Server.

There are a variety of ways to invoke the services of the Administration Server to accomplish configuration tasks. Whichever method is used, the Administration Server for a cluster must be running when you modify the configuration.

3.11.1 Administration Server Failure

The failure of an Administration Server for a domain does not affect the operation of managed servers in the domain. If an Administration Server for a domain becomes unavailable while the server instances it manages—clustered or otherwise—are up and running, those managed servers continue to run. If the domain contains clustered server instances, the load balancing and failover capabilities supported by the domain configuration remain available, even if the Administration Server fails.

> **Note:** If an Administration Server fails because of a hardware or software failure on its host machine, other server instances on the same machine may be similarly affected. However, the failure of an Administration Server itself does not interrupt the operation of managed servers in the domain.

For instructions on re-starting an Administration Server, see the *Oracle Fusion Middleware Using Clusters for Oracle Server*.

3.11.2 Node Manager Failure

If Node Manager fails or is explicitly shut down, upon restart, it determines the server instances that were under its control when it exited. Node Manager can restart any failed server instances as needed.

Note: It is advisable to run Node Manager as an operating system service, so that it restarts automatically if its host machine is restarted..

3.12 Load Balancing

Load balancing configuration consists of three pieces of information, the load-balancing algorithm to be used, an indicator of whether local affinity should be applied, and weights that are assigned to each member of the topology to influence any routing algorithms that utilize weights.

The load-balancing algorithm specifies how requests are load balanced across components. Oracle Fusion Middleware uses the following three load-balancing methods:

- Round Robin - Requests are balanced across a list of available servers by selecting from the list sequentially.
- Random - Requests are balanced across a list of available servers by selecting a random server on each request.
- Weighted - Requests are balanced across a list of available servers using weights assigned to each server to determine the percentage of requests sent to each

Local affinity determines whether clients show a preference to servers that run on the same machine to avoid network latency. If the flag is set to true, then requests are routed across the list of servers on the local machine using the load-balancing algorithm if any local servers are available. If no local servers are available, requests are routed to all available remote servers according to the load-balancing algorithm. If local affinity is set to false, requests are routed across all available servers (local and remote) based on the load-balancing algorithm.

You configure weights as single integer values that are associated with component instances. You can assign weights to components that are not currently in a group, however, the weight is not used unless you later configure the component as a member of a group and select the weighted load-balancing algorithm. The weight is a unitless number. The percentage of requests to be sent to each member is calculated by summing the weights of all available members and dividing the weight for each member by the sum of the weights.

3.13 Multi Data Sources

A multi data source is an abstraction around a group of data sources that provides load balancing or failover processing at the time of connection requests, between the data sources associated with the multi data source. Multi data sources are bound to the JNDI tree or local application context just like data sources are bound to the JNDI tree. Applications lookup a multi data source on the JNDI tree or in the local application context (java:comp/env) just as they do for data sources, and then request a database connection. The multi data source determines which data source to use to satisfy the request depending on the algorithm selected in the multi data source configuration: load balancing or failover.

A multi data source can be thought of as a pool of data sources. Multi data sources are best used for failover or load balancing between nodes of a highly available database system, such as redundant databases or Oracle Real Application Clusters (RAC).

3.14 Cluster Configuration and config.xml

The `config.xml` file is an XML document that describes the configuration of a WebLogic Server domain. The `domain` element in `config.xml` is the top-level element, and all elements in the domain descend from the `domain` element. The `domain` element includes child elements such as the `server`, `cluster`, and `application` elements. These child elements may have children of their own. For example, the `server` element can include the child elements WebServer, SSL and Log. The Application element includes the child elements EJBComponent and WebAppComponent.

Each element has one or more configurable attributes. An attribute defined in config.dtd has a corresponding attribute in the configuration API. For example, the Server element has a ListenPort attribute, and likewise, the `Weblogic.management.configuration.ServerMBean` has a `ListenPort` attribute. Configurable attributes are readable and writable, that is, `ServerMBean` has a `getListenPort()` and a `setListenPort()` method.

To learn more about `config.xml`, see the *Oracle Fusion Middleware Using Clusters for Oracle WebLogic Server*.

3.15 About Singleton Services

A singleton service is a service that must run on only a single server instance at any given time, such as JMS and the JTA transaction recovery system, when the hosting server instance fails. A managed server configured for automatic migration will be automatically migrated to another machine in the event of failure.

3.16 WebLogic Server and LDAP High Availability

In a high availability environment, WebLogic Server needs to be able to access LDAP for these reasons:

- To access users and groups stored in LDAP for which WebLogic Server supports failover.

  For information about configuring failover for LDAP authentication providers, see "Configuring Failover for LDAP Authentication Providers" in the *Oracle Fusion Middleware Securing Oracle WebLogic Server* manual.

- To access the LDAP-based policy store and credential store.

  For information about configuring a domain to use an LDAP-based policy store, see "Configuring a Domain to Use an LDAP-Based Policy Store" in the *Oracle Fusion Middleware Security Guide* manual.

  For information about configuring a domain to use an LDAP-based credential store, see "Configuring a Domain to Use an LDAP-Based Credential Store" in the *Oracle Fusion Middleware Security Guide* manual.

4 Considerations for High Availability Oracle Database Access

This chapter describes considerations for high availability Oracle database access.

The sections in this chapter are as follows:

4.1 Oracle Real Application Clusters and Fusion Middleware

Most Fusion Middleware components use a database as the persistent store for their data. The Oracle database back end can be configured in any number of high availability configurations, including Cold Failover Clusters, Real Application Clusters, Oracle Data Guard, or Oracle Streams. For more information on these high availability configurations, see the *Oracle Database High Availability Overview*. This chapter describes considerations for Oracle Fusion Middleware configured with a high availability Oracle database, Oracle Real Application Clusters.

Oracle Real Application Clusters (Oracle RAC) is a computing environment that harnesses the processing power of multiple, interconnected computers. Along with a collection of hardware, called a cluster, it unites the processing power of each component to become a single, robust computing environment. A cluster comprises two or more computers, also called nodes. Oracle Real Application Clusters simultaneously provides a highly scalable and highly available database for Oracle Fusion Middleware.

Every Oracle RAC instance in the cluster has equal access and authority, therefore, node and instance failure may affect performance, but does not result in downtime, since the database service is available or can be made available on surviving server instances.

For more information on Oracle Real Application Clusters, see the *Oracle Real Application Clusters Administration and Deployment Guide*.

Oracle Fusion Middleware provides the best integration with an Oracle database in a high availability environment. When Oracle Fusion Middleware behaves as a client for the database (either as a java or system client) it uses special communication and monitoring capabilities that provide fast failover and minimal middle tier disruption in reaction to database failure scenarios.

Oracle Fusion Middleware components that access the database can be categorized in three ways:

- Java-based Oracle Fusion Middleware components deployed to Oracle WebLogic Server
- Java-based Oracle Fusion Middleware components that are standalone Java Clients
- Non-Java Oracle Fusion Middleware components

This chapter contains the following sections:

4.1.1 Java-Based Oracle Fusion Middleware Components Deployed to Oracle WebLogic Server

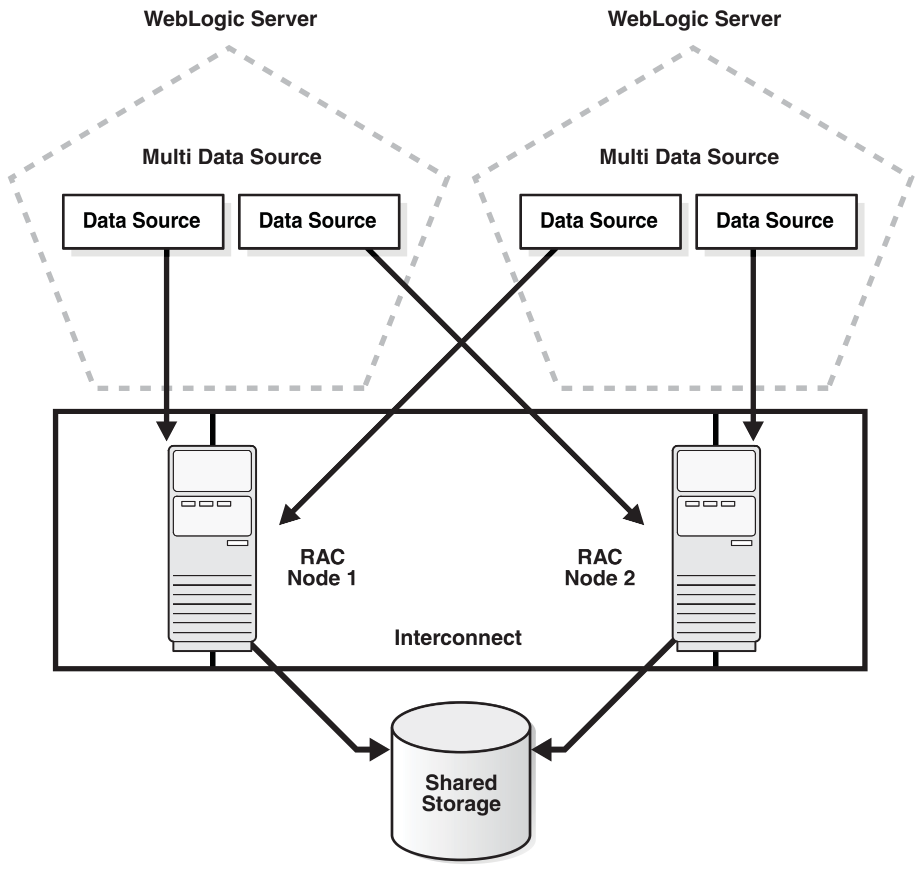
All Oracle Fusion Middleware components deployed to Oracle WebLogic Server support Oracle Real Application Clusters (RAC). For establishing connection pools, Oracle Fusion Middleware supports only multi data sources for the Oracle RAC back end for both XA and non-XA JDBC drivers. For connection pooling, Oracle Fusion Middleware deployments do not support other connection failover features supported by Oracle JDBC drivers for Oracle RAC. Oracle RAC multi data sources are configured by Oracle WebLogic Server Configuration Wizard. You can also configure multi data sources using the Oracle Fusion Middleware Administration console, or WLST commands. Please refer to component specific guides for multi data source configuration details

When an Oracle RAC node or instance fails, session requests are redirected to another node in the cluster, either by Oracle WebLogic Server or by the Oracle Thin driver. There is no failover of existing connections, however, new connection requests from the application managed using existing connections in the Oracle WebLogic pool or by new connections to the working Oracle RAC instance. In-flight transactions are typically rolled back when the database is the transaction manager. When the WebLogic Server is the Transaction Manager, in-flight transactions are failed over, meaning they are driven to completion or rolled back, based on the state of the transaction at the time of the failure. If the application requires load balancing across Oracle RAC nodes, WebLogic Server supports this capability through use of JDBC multi data sources configured for load balancing. The data sources that form a multi data source are accessed using a round-robin scheme (the Oracle recommended configuration for deployments against RAC databases). When switching connections, WebLogic Server selects a connection from the next data source in the order listed. The next section briefly describe configuration of multi data sources with Oracle RAC.

4.1.2 Using Multi Data Sources with Oracle RAC

To connect Oracle WebLogic Server to multiple Oracle RAC nodes using multi data sources, configure a JDBC data source for each Oracle RAC instance in your Oracle RAC cluster with the Oracle Thin driver. Then configure a multi data source, using either the algorithm for load balancing or the algorithm for failover, and add the data sources to it.

Figure 4–1 Multi Data Source Configuration



4.1.2.1 Configuring Multi Data Sources for MDS Repositories

Applications that use an MDS database-based repository can be configured for high availability Oracle database access. With this configuration, failure detection, recovery, and retry by MDS, as well as by the WebLogic infrastructure result in the application's read-only MDS operations being protected from Oracle RAC database planned and unplanned downtimes.

MDS multi data sources are exposed as MDS repositories in the Fusion Middleware Control navigation tree. These multi data sources can be selected during deployment plan customization of application deployment, and can be used with MDS WLST commands.

- Configuring an application to retry read-only operations

  To configure an application to retry the connection, you can configure the RetryConnection attribute of the application's MDS AppConfig MBean. For information about MDS configuration, see section 8.7 of the *Oracle Fusion Middleware Administrator's Guide*.

- Registering an MDS multi data source

  In addition to the steps specified in section Section 4.1.3, "Configuring Multi Data Sources with Oracle RAC," consider the following for MDS:

  – The child data sources that constitute a multi data source used for an MDS Repository must be configured as non-XA data sources.

  – The multi data source's name must be pre-fixed with `mds-`. This is required so the multi data source can be recognized as an MDS repository that can be used for MDS management functionality through Fusion Middleware Control, WLST, and JDeveloper.

Note: When an MDS data source is added as a child of a multi data source, this data source is no longer exposed as an MDS repository. For example, it is not displayed under the Metadata Repositories folder in the Fusion Middleware Control navigation tree, no MDS repository operations can be performed on it, and it does not appear in the list of selectable repositories during deployment.

- Converting a data source to a multi data source

  There are two considerations when converting an MDS data source to a multi data source to make sure the application is configured correctly:

  – If you are creating a new multi data source with a new, unique name, redeploy the application and select this new multi data source as the MDS repository during deployment plan customization.

  – If you want to avoid redeploying the application, you can delete the data source and recreate the new multi data source using the same name and jndi-name attributes.

4.1.2.2 Oracle RAC Configuration Requirements

This section describes requirements for Oracle RAC configuration:

- **XA Requirements**: Many Oracle components participate in distributed transactions, or are part of container managed transactions. These components require the back-end database setup for XA recovery by Oracle WebLogic Transaction Manager. For repositories created using RCU, this is done automatically. For other databases participating in XA transactions, ensure that XA pre-requisites are met:

  1. Log on to sqlplus as a system user, for example:

     ```
     sqlplus "/ as sysdba"
     ```

  2. Grant select on `sys.dba_pending_transactions` to `public`.
  3. Grant execute on `sys.dbms_xa` to `public`.
  4. Grant force on any transaction to `user`.

Note: Ensure that the `distributed_lock_timeout` parameter for the Oracle database is set to a value higher that the JTA timeout (This should be higher than the highest value on the middle tier - between the default for WebLogic Server, a specific configuration for a data source, or one used by a component for a transaction.

- Server-side Load Balancing: If the server-side load balancing feature has been enabled for the Oracle RAC back end (using remote_listeners), the JDBC URL used in the data sources of a multi data source configuration should include the INSTANCE_NAME. For example, you can specify the URL in the following format:

  ```
  jdbc:oracle:thin:@(DESCRIPTION=(ADDRESS=(PROTOCOL=TCP)(HOST=host-vip)
  (PORT=1521))(CONNECT_DATA=(SERVICE_NAME=dbservice)(INSTANCE_NAME=inst1)))
  ```

By default, the out-of-box installation assumes that `remote_listener` has been configured and creates the URL for data sources in a multi data source accordingly. Any multi data source created outside of the typical installation and configuration should follow the format described in this section.

If `remote_listeners` cannot be specified on the Oracle RAC side, and server side load balancing has been disabled, specifying the INSTANCE_NAME in the URL is not necessary. To disable remote listeners, delete any listed remote listeners in `spfile.ora` file on each Oracle RAC node. For example:

```
*.remote_listener="
```

In this case, the recommended URL that you use in the data sources of a multi data source configuration is:

```
jdbc:oracle:thin:@host-vip:port/dbservice
```

Or

```
jdbc:oracle:thin:@(DESCRIPTION=(ADDRESS=(PROTOCOL=TCP)(HOST=host-vip)(PORT=1521
))(CONNECT_DATA=(SERVICE_NAME=dbservice)))
```

- **Services**: When configuring Oracle Fusion Middleware for the Oracle database and specifically for Oracle RAC, Oracle recommends using the Oracle Services feature. Create the `service_name` provided as part of the database service location specifically for the application.

4.1.3 Configuring Multi Data Sources with Oracle RAC

Oracle Middleware 11g components support only multi data source configuration for Oracle RAC. You can configure multi data sources using the following:

- Oracle WebLogic Server Configuration Wizard during WebLogic Server domain creation
- Oracle Universal installer Java EE component configuration for Identity Management or Oracle Portal, Forms, Reports, and Discoverer.
- Oracle WebLogic Server Administration Console
- WLST Commands

Multi data sources support load balancing for both XA as well as non-XA data sources. This applies to all Oracle database versions supported by Oracle Fusion Middleware components.

Multi data sources encapsulate individual data sources that pool connection to specific instances of Oracle RAC. For multi data sources created manually, or modified after initial configuration, Oracle strongly recommends the following XA and Non-XA data source property values for optimal high availability behavior. If your environment so demands, changes to these should be done after careful consideration and testing:

Table 4–1 Recommended Multi Data Source Configuration

Property Name	Value
test-frequency-seconds	5
algorithm-type	Load-Balancing

For the individual data sources, Oracle recommends the following for high availability environments. Any other parameters should be set according to application requirements.

Table 4–2 XA Data Source Configuration

Property Name	Value
Driver	oracle.jdbc.xa.client.OracleXADataSource
Property command	<property> <name>oracle.net.CONNECT_TIMEOUT</name> <value>10000</value> </property>
initial-capacity	0
connection-creation-retry-frequency-seconds	10
test-frequency-seconds	300
test-connections-on-reserve	true
test-table-name	SQL SELECT 1 FROM DUAL
seconds-to-trust-an-idle-pool-connection	0
global-transactions-protocol	TwoPhaseCommit
keep-xa-conn-till-tx-complete	true
xa-retry-duration-seconds	300
xa-retry-interval-seconds	60

Table 4–3 Non-XA Data Source Configuration

Property Name	Value
Driver	oracle.jdbc.OracleDriver
Property to set	<property> <name>oracle.net.CONNECT_TIMEOUT</name> <value>10000</value> </property>
initial-capacity	0
connection-creation-retry-frequency-seconds	10
test-frequency-seconds	300
test-connections-on-reserve	true
test-table-name	SQL SELECT 1 FROM DUAL
seconds-to-trust-an-idle-pool-connection	0
global-transactions-protocol	None

For examples of recommended multi data sources, see Appendix B, "Recommended Multi Data Sources."

Increasing Transaction Timeout for XA Data Sources

If you see WARNING messages in the server logs that include the following exception:

javax.transaction.SystemException: Timeout during commit processing

```
[ javax.transaction.SystemException: Timeout during commit processing
```

This may indicate the XA timeout value you have in your setup must be increased. XA timeout can be increased for individual data sources when these warnings appear.

To increase this setting, use Oracle WebLogic Server Administration Console:

1. Access the data source configuration.
2. Select the **Transaction** tab.
3. Set XA Transaction Timeout to a larger value, for example, **300**.
4. Click **Save**.

Repeat this configuration for all individual data sources of an XA multi data source.

4.1.4 Oracle RAC Failover with WebLogic Server

Although Oracle RAC offers JDBC connect-time failover features, for most configurations, Oracle recommends using WebLogic JDBC multi data sources to manage failover. Connect-time failover does not provide the ability to pre-create connections to alternate Oracle RAC nodes, however, multi data sources have multiple connections available at all times to manage failover. For more information see Using Multi Data Sources with Oracle RAC.

4.1.5 JDBC Clients

Java J2SE-based Oracle Fusion Middleware components are optimized to work with the high availability features of Oracle RAC. The components can be deployed to use both the Oracle thin JDBC driver, or the OCI based JDBC drivers.

The JDBC Thin client is a pure Java, Type IV driver. It is lightweight and easy to install. It provides high performance, comparable to the performance provided by the JDBC Oracle Call Interface (OCI) driver. The JDBC Thin driver communicates with the server using TTC, a protocol developed by Oracle to access data from Oracle database. The driver allows a direct connection to the database by providing an implementation of TCP/IP those implements Oracle Net and TTC on top of Java sockets. The JDBC OCI client is a Type II driver and provides connections to JDBC clients over the Oracle Net. It uses the client side installation of Oracle Net and a deployment can customize behavior using Oracle Net configuration on the middle tier.

Note: These JDBC clients are used as part of standalone java J2SE programs.

Oracle Virtual Directory

When used with database adapters, Oracle Virtual Directory connects to a database, and the connections are not pooled. For details about configuring database adapters for Oracle RAC, see section Creating database Adapters, in the *Oracle Fusion Middleware Administrator's Guide for Oracle Virtual Directory*.

Database URL

To configure the an OVD database adapter for an Oracle RAC database using the Oracle Directory Services Manager:

1. In the **Connection** screen, select **Use Custom URL** from the **URL Type** list.
2. In the **Database URL** field, enter the URL to connect to the Oracle RAC database;, for example:

   JDBC Thin

   ```
   jdbc:oracle:thin:@(DESCRIPTION=(ADDRESS_LIST=(LOAD_
   BALANCE=ON)(ADDRESS=(PROTOCOL=TCP)(HOST=host-name-1)(PORT=1521))(ADDRESS=
   (PROTOCOL=TCP)(HOST=host-name-2)(PORT=1521)))(CONNECT_
   DATA=(SERVER=DEDICATED)(SERVICE_NAME=database-service-name)))
   ```

   JDBC OCI

   ```
   jdbc:oracle:oci:@(DESCRIPTION=(ADDRESS_LIST=(LOAD_
   BALANCE=ON)(ADDRESS=(PROTOCOL=TCP)(HOST=host-name-1)(PORT=1521))(ADDRESS=
   (PROTOCOL=TCP)(HOST=host-name-2)(PORT=1521)))(CONNECT_
   DATA=(SERVER=DEDICATED)(SERVICE_NAME=database-service-name)))
   ```

Connection Timeout Configuration

To configure the connection timeout for an Oracle RAC database using the Oracle Directory Services Manager:

1. In the **Connection** screen, for **JDBC Thin**, specify the database adapter parameter **oracleNetConnectTimeout** for the timeout parameter in seconds.
2. For JDBC OCI, specify `TCP.CONNECT_TIMEOUT=n` in the `sqlnet.ora in ORACLE_INSTANCE/config` directory.

4.1.6 System Clients

Oracle Fusion Middleware 11g includes some non-Java components. These components are primarily C-based and include Oracle Internet Directory (OID), Oracle Forms, Oracle Reports, Oracle Discoverer, and Oracle Portal. These components use Oracle Call Interface layer to interact with Oracle databases. For Oracle RAC-based systems, some of components integrate with the Oracle high availability Event Notification database feature.

High availability Event Notification provides a signal to the non-Java application if database failure occurs. The applications can register a callback on the environment to monitor the database connection. When a database failure related to the non-Java client occurs, the callback is invoked. This callback contains information about the database failure, including the event payload), a list of connections (server handles) that were disconnected as a result of the failure.

If another instance, for example, instance C, of the same database goes down, the client is not notified since it does not affect any of the client's connections. If another instance, for example, instance C, of the same database, goes down, the client will is not notified, since it does not affect any of the client's connections.

High availability Event Notification improves the response time of the application if database failure occurs. Without Event Notification, database failure would result in the connection being broken only after the TCP time out expired, which could take minutes. With high availability Event Notification, standalone, connection pool, and session pool connections are automatically broken and cleaned up by OCI and the application callback is invoked within seconds of failure. If any of these server handles are TAF-enabled, failover is automatically engaged by OCI.

The following section describes the recommended setting for non-Java client connections to Oracle RAC databases.

4.1.6.1 Oracle Internet Directory

Oracle Internet Directory integrates with high availability Event Notification. Oracle recommends using the Oracle Enterprise Manager Cluster Managed Services Page to create database services that client applications use to connect to the database.

You can also use SQL*Plus to configure your Oracle RAC database service.

To enable high availability event motivation for an Oracle RAC database connection:

1. Set the `AQ_HA_NOTIFICATIONS` attribute to **TRUE** and server-side Transparent Application Failover (TAF) settings are enabled. The failover retries and failover delay can be adjusted based on the requirements of the deployment. So for the database service used by OID, Oracle recommends setting Oracle Oracle RAC `DBMS_SERVICE` property values according to Table 4–4.

Table 4–4 OID Database Services Property Settings

Property Name	Value
AQ_HA_NOTIFICATIONS	TRUE
FAILOVER_METHOD	DBMS_SERVICE.FAILOVER_METHOD_BASIC
FAILOVER_TYPE	DBMS_SERVICE.FAILOVER_TYPE_SELECT
FAILOVER_RETRIES	5
FAILOVER_DELAY	5

2. Oracle also recommends setting TCP connect timeouts for the Oracle Net configuration. To configure this setting, specify `TCP.CONNECT_TIMEOUT=n` in the `sqlnet.ora` file in the `ORACLE_INSTANCE/config` directory.

4.1.6.2 Oracle Forms

Oracle Forms also integrates with high availability event notification. To enable this feature for Oracle Forms:

1. Use the Oracle Enterprise Manager Cluster Managed Services Page to create database services. For Oracle Forms, set the Oracle RAC `DBMS_SERVICE` property values according to Table 4–5the following is recommended to be set using the package of Oracle database.

Table 4–5 Oracle Forms Database Services Property Settings

Property Name	Value
AQ_HA_NOTIFICATIONS	TRUE
FAILOVER_METHOD	DBMS_SERVICE.FAILOVER_METHOD_NONE
FAILOVER_TYPE	DBMS_SERVICE.FAILOVER_TYPE_NONE

2. Oracle also recommends setting TCP connect timeouts for the Oracle Net configuration. To configure this setting, specify `TCP.CONNECT_TIMEOUT=n` in the `sqlnet.ora` file in the `ORACLE_INSTANCE/config` directory.

4.1.6.3 Oracle Portal

To configure Oracle Portal for optimal behavior in a high availability environment, set TCP connect timeouts for the Oracle Net configuration. To configure this setting, specify `TCP.CONNECT_TIMEOUT=n` in the `sqlnet.ora` file in the *ORACLE_INSTANCE*/config directory.

Oracle Portal also uses the death detection feature of `mod plsql`.

`mod_plsql` maintains a pool of connections to the database, and reuses established database connections for subsequent requests. If there is no response from a database connection in a connection pool, `mod_plsql` detects this, discards the dead connection, and creates a fresh database connection for subsequent requests.

The dead database connection detection feature of `mod_plsql` eliminates the occurrence of random errors when a database node or instance goes down. This feature is also extremely useful in high availability configurations, such as Real Application Cluster (RAC). If a node in an Oracle RAC cluster goes down, `mod_plsql` detects this and immediately starts servicing requests using the other Oracle RAC nodes.

By default, when an Oracle RAC node or database instance goes down and `mod_plsql` had previously pooled connections to the node, the first `mod_plsql` request which uses a dead connection in its pool results in a failure response of HTTP-503 being sent back to the end-user. This failure is then used by `mod_plsql` to trigger the detection and removal of all dead connections in its pool. `mod_plsql` pings all connection pools that were created before the node failure, and this ping operation is performed at the time of processing the next request that uses a pooled connection. If the ping operation fails, the database connection is discarded, and a new connection is created and processed.

Note: If after node failure, multiple `mod_plsql` requests come in concurrently, and `mod_plsql` has not yet detected the first dead connection, there could be multiple failures at that instant.

`mod_plsql` provides two configuration options for tuning the dead database connection detection feature:

- Specifying the Option to Detect Dead Database Connections
- Specifying the Timeout Period

Specifying the Option to Detect Dead Database Connections.

`mod_plsql` corrects connections after it detects a failure that could be caused by a database node going down. This is controlled by the `PlsqlConnectionValidation` parameter. For details on the PlsqlConnectionValidation parameter, refer to the "mod_plsql" section in the *Oracle HTTP Server Administrator's Guide*.

Setting the `PlsqlConnectionValidation` parameter to **Automatic** causes the `mod_plsql` module to test all pooled database connections that were created before a failed request. This is the default configuration.

Setting the `PlsqlConnectionValidation` parameter to **AlwaysValidate** causes `mod_plsql` to test all pooled database connections before issuing any request. Although the AlwaysValidate configuration option ensures greater availability, it also introduces additional performance overhead.

You can specify the timeout period for mod_plsql to test a bad database connection in a connection pool. The `PlsqlConnectionTimeout` parameter, which specifies the maximum time `mod_plsql` should wait for the test request to complete before it assumes that a connection is not usable.

Specifying the Connection Validation and Timeout Period

When the `PlsqlConnectionValidation` parameter is set to **Automatic** or **AlwaysValidate**, `mod_plsql` attempts to test pooled database connections.

You can specify the timeout period for `mod_plsql` to test a bad database connection in a connection pool. This is controlled by the `PlsqlConnectionTimeout` parameter, which specifies the maximum time mod_plsql should wait for the test request to complete before it assumes that a connection is not usable.

For details on the `PlsqlConnectionTimeout`, `PlsqlConnectionValidation`, and `PlsqlConnectionTimeout` parameter, refer to the "mod_plsql" section in the *Oracle HTTP Server Administrator's Guide*.

4.1.6.4 Oracle Reports and Oracle Discoverer

To configure Oracle Reports and Oracle Discovery for optimal behavior in a high availability environment, set TCP connect timeouts for the Oracle Net configuration. To configure this setting, specify `TCP.CONNECT_TIMEOUT=n` in the `sqlnet.ora` file in the `ORACLE_INSTANCE/config` directory.

Oracle Discoverer also uses a tns entry to connect to the RAC database:

```
frdisco = (DESCRIPTION = (LOAD_BALANCE = ON) (ADDRESS_LIST =
(ADDRESS = (PROTOCOL = TCP)(HOST = stajo05-vip)(PORT = 1521))
(ADDRESS = (PROTOCOL = TCP)(HOST = stajo06-vip)(PORT = 1521)))
(CONNECT_DATA = (SERVICE_NAME = orcl.us.oracle.com)))
```

4.2 Protecting Idle Connections from Firewall Timeouts

Most production deployments involve firewalls and database connection are made across firewalls, Oracle recommends configuring the firewall not to timeout the database connection. For Oracle RAC case, this specifically means not timing out the connections made on Oracle RAC VIPs and the database listener port.

If such a configuration is not possible, on the database server side, set `SQLNET.EXPIRE_TIME=n` in `ORACLE_HOME/network/admin/sqlnet.ora`. For Oracle RAC, this needs to be set on all the Oracle Homes. The `n` is in minutes. It should be set to less than the known value of the network device (firewall) timeout. Since the order of these times is normally more than ten minutes, and in some cases hours, the value should be set to the highest possible value.

4.3 Troubleshooting Real Application Clusters

Fusion Middleware components use multi data sources when connecting to an Oracle RAC database. If an Oracle RAC instances goes down, WebLogic Server attempts to determine the status of the of the database using the SELECT 1 FROM DUAL query. This query typically takes less than few seconds to complete. However, if the database response is slow, WebLogic Server gives up and assumes the database is unavailable. The following is an example of the type of exception that results in the logs:

```
<Mar 30, 2009 2:14:37 PM CDT> <Error> <JDBC> <BEA-001112> <Test "SELECT 1 FROM
 DUAL" set up for pool SOADataSource-rac1" failed with exception:
 oracle.jdbc.xa.OracleXAException".> [TopLink Warning]: 2009.03.30
 14:14:37.890--UnitOfWork(14568040)--Exception [TOPLINK-4002] (Oracle TopLink -
 11g Release 1 (11.1.1.1.0) (Build 090304)):
 oracle.toplink.exceptions.DatabaseException Internal Exception:
 java.sql.SQLException: Internal error: Cannot obtain XAConnection Creation of
 XAConnection for pool SOADataSource failed after waitSecs:30 :
 weblogic.common.ResourceException: SOADataSource(SOADataSource-rac1): Pool
 SOADataSource-rac1 has been @ disabled because of hanging connection tests,
 cannot allocate resources to applications. We waited 10938 milliseconds. A
 typical test has been taking 16.
```

You can set the WebLogic Server parameter, `-Dweblogic.resourcepool.max_test_wait_secs=30` to increase the time WebLogic Server waits for a response from the database. This parameter is located in the setDomainEnv.sh file. By setting this parameter, WebLogic Server waits 30 seconds for the database to respond to the SELECT 1 FROM DUAL query before giving up.

5 Configuring High Availability for Oracle Fusion Middleware SOA Suite

Oracle Fusion Middleware SOA Suite provides a complete set of service infrastructure components for designing, deploying, and managing composite applications. Oracle Fusion Middleware SOA Suite enables services to be created, managed, and orchestrated into composite applications and business processes. Composites enable you to easily assemble multiple technology components into one SOA composite application. Oracle Fusion Middleware SOA Suite plugs into heterogeneous IT infrastructures and enables enterprises to incrementally adopt SOA.

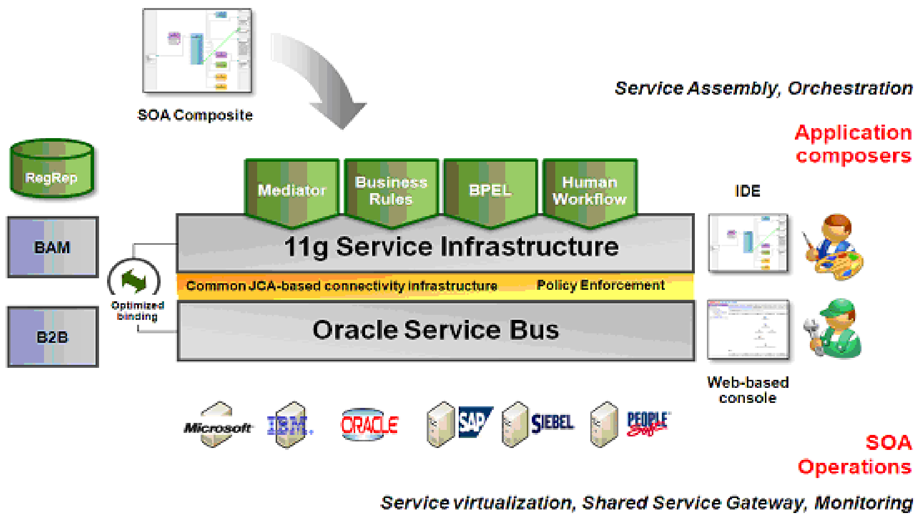
This chapter provides a description of Oracle SOA Suite components from a high availability perspective. The sections in this chapter outline the single-instance concepts that are important for designing a high availability deployment.

This chapter includes the following topics:

5.1 Introduction to Oracle Fusion Middleware SOA Suite

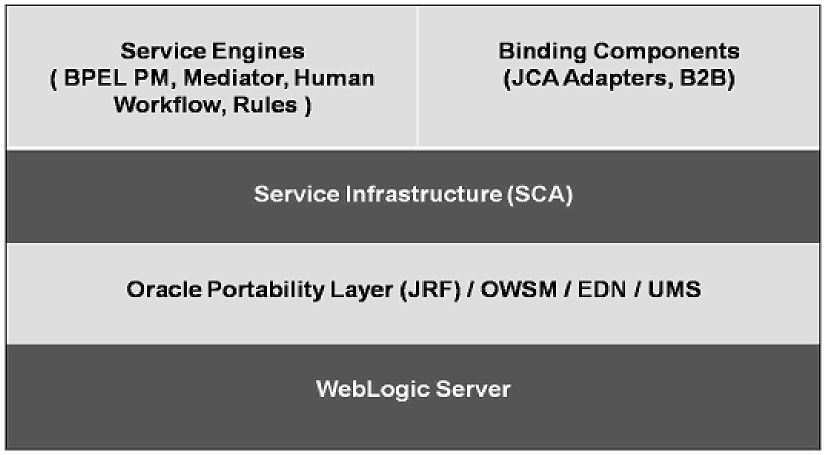
As illustrated in Figure 5–1, Oracle SOA Suite provides a comprehensive suite of products for developing, securing, and monitoring service-oriented architecture (SOA). Oracle SOA Suite 11g provides a unified runtime based on the SCA standard. The runtime consists of Service Engines (BPEL Process Manager, Human Workflow, Mediator, Rules) and Binding Components (JCA Adapters, B2B) that are managed and inter-connected by a common Service Infrastructure. Service infrastructure also provides common services such as lifecycle management and deployment.

Figure 5–1 Oracle Fusion Middleware SOA Suite and Components



A SOA composite application is the basic unit of deployment to the SOA runtime. Service components (BPEL process, business rule, human task, and mediator routing rule) are the building blocks of a SOA composite. Components are targeted to Service Engines during deployment while Services and References are enabled using the Binding Components. At runtime, messages are received by the Binding Component and are then routed to the appropriate Service Engine(s) by the Service Infrastructure.

Figure 5–2 Oracle SOA Infrastructure Stack Diagram


Service Engines (BPEL PM, Mediator, Human Workflow, Rules)
Binding Components (JCA Adapters, B2B)
Service Infrastructure (SCA)
Oracle Portability Layer (JRF) / OWSM / EDN / UMS
WebLogic Server

The SOA runtime executes within the context of an application server such as the Oracle WebLogic Application Server. It leverages the underlying application server capabilities for load balancing and high availability.

This guide provides high availability information for the following SOA Suite components:

- Oracle Service Infrastructure
- Oracle BPEL Process Manager
- Oracle Mediator
- Oracle Human Workflow
- Oracle JCA Adapters
- Oracle B2B
- Oracle Web Service Manager
- Oracle User Messaging Service
- Oracle Business Activity Monitoring

5.2 Oracle SOA Service Infrastructure High Availability

Oracle SOA Service Infrastructure is a Java EE application that provides the foundation services for running Oracle Fusion Middleware SOA Suite. This Java EE application is a runtime engine that is automatically deployed when Oracle Fusion Middleware SOA Suite is installed. You deploy *composites* (the basic artifacts in a Software Component Architecture) to the Oracle SOA Infrastructure and it provides the required services for the composites to run. Oracle SOA Infrastructure provides deployment, wiring, and thread management services for the composites. These services sustain the composite’s lifecycle and runtime operations.

The information in this section guides you through the issues and considerations necessary for designing a SOA Service Infrastructure high availability cluster. Later sections of this chapter describe the same issues and considerations for the following SOA Service Infrastructure-related components.

- Oracle BPEL Process Manager (Oracle BPEL PM)
- Oracle Mediator
- Oracle JCA Adapters
- Oracle Human Workflow
- Oracle B2B
- Oracle Web Services Management and Security (Oracle WMS)
- Oracle User Messaging Service
- Oracle Business Activity Monitoring

Backup and Recovery Considerations

For general information on backing up SOA Service Infrastructure files, see the "Introducing Backup and Recovery" and "Backing Up Your Environment" chapters of the *Oracle Fusion Middleware Administrator’s Guide*.

5.2.1 Oracle SOA Service Infrastructure Single-Instance Characteristics

The SOA Service Infrastructure is a Java EE-compliant application running on Oracle WebLogic Server. It provides the required services for running composites. A composite is basic unit of deployment for Service Component Architectures (SCA). The SCA Assembly Model is made up of a series of artifacts, which are defined by elements contained in XML files.

Composites are software packages made up of components, wires, services and references. For example, an Oracle BPEL process is a component, an inbound adapter is a service, an outbound adapter is a reference. Wires provide connections between service engines.

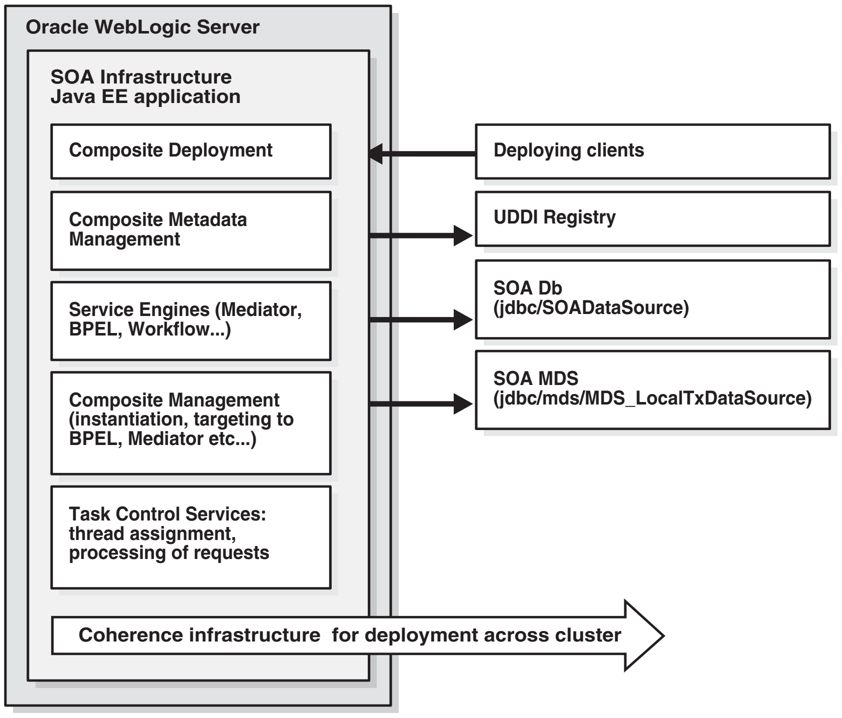
Individual components are targeted to specific service engines, such as Oracle BPEL PM, or Oracle Mediator. Oracle SOA Service Infrastructure connects to a SOA database to maintain composite state, and to the SOA Oracle Metadata Services (MDS) Repository to maintain composite metadata, such deployments, and version tracking. These two databases may be the same physical database, but the schemas used for each purpose are different. SOA infra provides a servlet for remote deployment of composites. The metadata and artifacts for remote deployments are also stored in the MDS repository. For more information on the MDS repository, see Section 4.1.2.1, "Configuring Multi Data Sources for MDS Repositories."

Oracle SOA Service Infrastructure application is also responsible for targeting the individual components to their specific engine and for instantiating these composites when requests reach the SOA system. After targeting and instantiation, Oracle SOA Service Infrastructure controls thread and resource assignment. This happens in the JVM, where the composite runs.

As illustrated in Figure 5–3, Oracle SOA Service Infrastructure integrates SOA composite applications with UDDI registries. UDDI registries provide a standards-based foundation for locating published services, invoking services, and managing service metadata. Oracle SOA Service Infrastructure is also the central hub used by the service engines to deliver messages through Oracle User Messaging Service infrastructure to communication channels, such as email and voice.

SOA Service Infrastructure provides the required services that sustain the different pieces in a Service Component Architecture, and allows the communication between them. For more details of the different components in an SCA system, refer to the *Oracle Fusion Middleware Administrator's Guide for Oracle SOA Suite.*

Figure 5–3 Basic Single-Node SOA Service Infrastructure Architecture


Oracle WebLogic Server
SOA Infrastructure Java EE application
Composite Deployment
Composite Metadata Management
Service Engines (Mediator, BPEL, Workflow...)
Composite Management (instantiation, targeting to BPEL, Mediator etc...)
Task Control Services: thread assignment, processing of requests
Coherence infrastructure for deployment across cluster
Deploying clients
UDDI Registry
SOA Db (jdbc/SOADataSource)
SOA MDS (jdbc/mds/MDS_LocalTxDataSource)

5.2.1.1 Oracle SOA Service Infrastructure Application Characteristics

Oracle SOA Service Infrastructure Services are contained in the `soa-infra-wls.ear` file. None of the services provided by Oracle SOA Service Infrastructure system are singletons, therefore, Oracle SOA Service Infrastructure can run in full active-active mode. The SOA Service Infrastructure Java EE application contains a Web module that provides browsing of the deployed composites as well as links to the test pages for these composites. This Web module uses `/soa-infra` as the associated URL context. This Web module is stateless and does not have any specific session replication requirements.

Other modules in Oracle SOA Service Infrastructure application provide task control for process instantiation and process tracking, as well as client services for accessing User Messaging System (UMS).

A task service controls instantiating and tracking processes asynchronously. In addition, there are multiple EJBs used by Oracle SOA Service Infrastructure system. However, all of the EJBs are stateless, and there are no requirements for stateful session bean replication in an Oracle SOA cluster. The processing of transactions by these EJBs relies on Oracle WebLogic Server transaction control service. Configure the appropriate transaction stores as recommended in the basic Oracle WebLogic Server guidelines to guarantee recovery across failures in Oracle WebLogic Server container.

5.2.1.2 Oracle SOA Service Infrastructure Startup and Shutdown Lifecyle

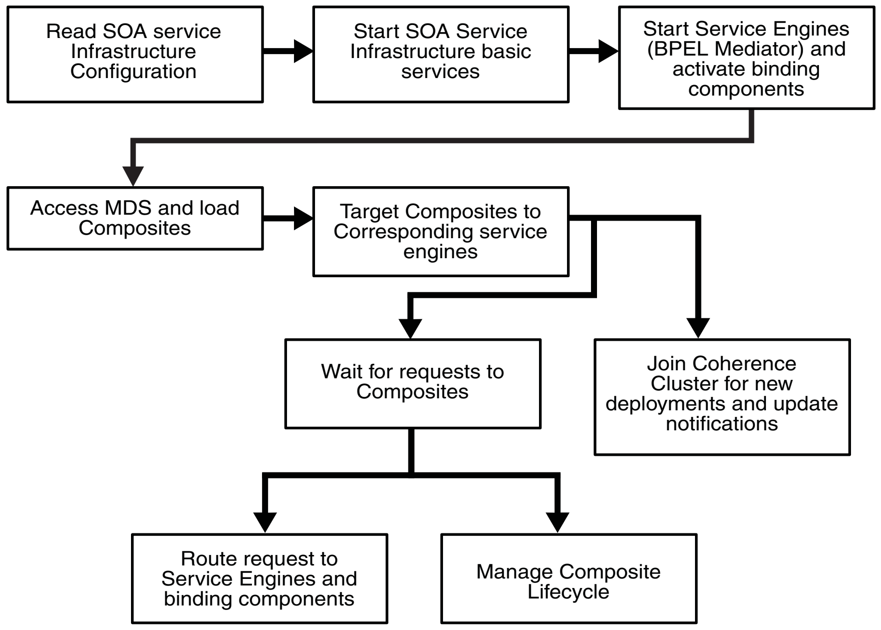
An Oracle SOA composite consists of the following:

- Components such as a BPEL process, Human Workflow task, a Mediator routing rule or Business Rules.
- Services and References for connecting Oracle SOA composite applications to external services, applications, and technologies.

These components are assembled together into an Oracle SOA composite application. This application is a single unit of deployment that simplifies the management and lifecycle of Oracle SOA applications.

When Oracle SOA Service Infrastructure application starts, it initializes the different service engines and loads the present composites from the MDS repository. It targets the individual components to their specific engines. Once the composite is loaded, the system is available to receive requests. At runtime, Oracle SOA Service Infrastructure manages all communication across service components. These calls between service engines are in-process calls. The following diagram reflects the sequence for Oracle SOA Service Infrastructure start, and processing of work:

Figure 5–4 Oracle SOA Service Infrastructure Application Startup and Shutdown Lifecyle


Read SOA service Infrastructure Configuration
Start SOA Service Infrastructure basic services
Start Service Engines (BPEL Mediator) and activate binding components
Access MDS and load Composites
Target Composites to Corresponding service engines
Wait for requests to Composites
Join Coherence Cluster for new deployments and update notifications
Route request to Service Engines and binding components
Manage Composite Lifecycle

Oracle SOA Service Infrastructure Application Startup and Shutdown Lifecyle

****

5.2.1.3 Oracle SOA Service Infrastructure External Dependencies

As described in the previous sections, Oracle SOA Service Infrastructure systems depends on the following components:

- Instance manager service depends on the runtime SOA database schema (soa-infra).
- Composite meta-data is stored in the MDS database schema which acts as a repository.
- In a clustered environment, the deployment coordinator service depends on the underlying Coherence cluster for signal propagation.

All three of these components must be available for Oracle SOA Service Infrastructure to start and run properly.

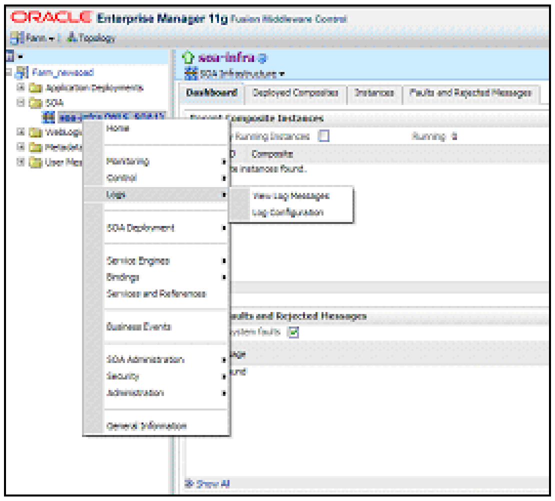
5.2.1.4 Oracle SOA Service Infrastructure Startup and Shut Down of Processes

Oracle SOA Service Infrastructure application is started by default when ever any Oracle WebLogic Managed Server to which Oracle SOA Service Infrastructure has been deployed is started. Normally, you should not need to stop Oracle SOA Service Infrastructure or any of its components by themselves. Some operations may required Oracle WebLogic Managed Server where the SOA Service Infrastructure runs to be rebooted. Only some patching scenarios could require stopping the application.

You can use Oracle WebLogic Server Administration Console to verify status and to start and stop Oracle WebLogic Server. You can also use WebLogic Server WLST command line to control the application. Oracle Enterprise Manager Fusion Middleware Control also allows multiple operations and configuration of Oracle SOA Service Infrastructure application as well as monitoring its status. See the *Oracle Fusion*

Middleware Administrator's Guide for Oracle SOA Suite for details on monitoring and controlling Oracle SOA Service Infrastructure application.

Figure 5–5 Monitoring and Controlling Oracle SOA Service Infrastructure Application



5.2.1.5 Oracle SOA Service Infrastructure Configuration Artifacts

The main controls of Oracle SOA Service Infrastructure can be configured using the soa-infra-config.xml file located in the `DOMAIN_HOME/config/soa-infra/configuration` directory. In this file you can specify:

- The data source JNDI name for process dehydration.

> **Note:** Once the JNDI names are read from this file, the databases used by the system are determined by the data sources that matched those JNDI names in WebLogic Server JDBS resources configuration.

- The server and callback URL

  A CallBack URL is the address that asynchronous services specify to be notified of a response to the service they invoked.

Note: If a request to an external or internal asynchronous service originates from SOA, the callback URL is determined using the following in decreasing order of preference:

- Use `callbackServerURL` specified as a binding property for the specific reference. (You can set this when modeling the composite or at runtime using the MBeans). This allows different service calls to have different callback URLs. At runtime this property is set using the System MBean Browser, through the corresponding binding Mbean. To add a specific URL add a `callbackServerURL` property to its `Properties` attribute, then invoke the save operation.
- Use the callback URL as specified in the `soa-infra-config.xml` file. In this case, only one address can be specified and an address that works well for all possible services must be used.
- Use the callback URL as the frontend host specified in WebLogic Server for the SOA_Cluster. In this case as well, only one address can be specified, and the recommendation is same as for `soa-infra-config.xml` file.
- Use the local host name as provided by WebLogic Server MBean APIs.This is not recommended in high availability environments.

- The audit level of information to be collected by the message tracking infrastructure.

These options are available also from Oracle Enterprise Manager Fusion Middleware Control and are specific to each Oracle SOA Service Infrastructure installation. Other configuration options at the container level, such as data sources, JTA configuration, and persistent stores, are maintained as part of the WebLogic Server Domain configuration and are synchronized across a cluster of Oracle WebLogic Servers by Oracle WebLogic Server core infrastructure.

5.2.1.6 Oracle SOA Service Infrastructure Log File Locations

The operations performed by Oracle SOA Service Infrastructure and it's components are logged by Oracle WebLogic Managed Server where the SOA Service Infrastructure is running. You can find these logs at the following location:

```
DOMAIN_HOME/servers/WLS_ServerName/logs/WLS_ServerName.log
```

The log files for the different Oracle WebLogic Sever managed server are also available from Oracle WebLogic Server Administration Console. To verify the logs, access Oracle WebLogic Server Administration Console using the following URL: `admin_server_host:port/console`. Click **Diagnostics-Log Files**.

It is also important to verify the output of Oracle WebLogic Managed Server where Oracle SOA Service Infrastructure is running. This information is stored at the following location:

```
DOMAIN_HOME/servers/WLS_ServerName/logs/WLS_ServerName.out
```

Additionally, a diagnostic log is produced in the log directory for the managed server. This log's granularity and logging properties can be changed through the `domain_dir/config/fmwconfig/servers/WLS_ServerName/logging.xml` file. The properties in this file can also be modified from Oracle Enterprise Manager Fusion

Middleware Control by selecting **Farm**, **SOA**, **SOA Server**. Right-click, and select **Logs**, and then **Log Configuration**.

Oracle Enterprise Manager Fusion Middleware Control allows performing selective searches in all the logs in the SOA domain. To do a selective search, access Oracle Fusion Middleware Control and click on **Farm-Logs** and enter the search criteria that pertain to `soa-infra` or deployed composites. See the *Oracle Fusion Middleware Administrator's Guide for Oracle SOA Suite* for more details on the logs and information reported for a SOA System in Oracle Enterprise Manager Fusion Middleware Control.

5.2.2 Oracle SOA Service Infrastructure High Availability Architecture and Failover Considerations

Figure 5–6 illustrates a two-node Oracle SOA Service Infrastructure cluster running on two Oracle WebLogic Servers. Oracle WebLogic Servers are front ended by an Oracle HTTP Server, which load balances incoming requests to the servers.

Note: The diagram describes high availability with a focus only on Oracle SOA without addressing other entities, such as Oracle HTTP Server and load balancers.

Figure 5–6 Oracle SOA Service Infrastructure High Availability Architecture

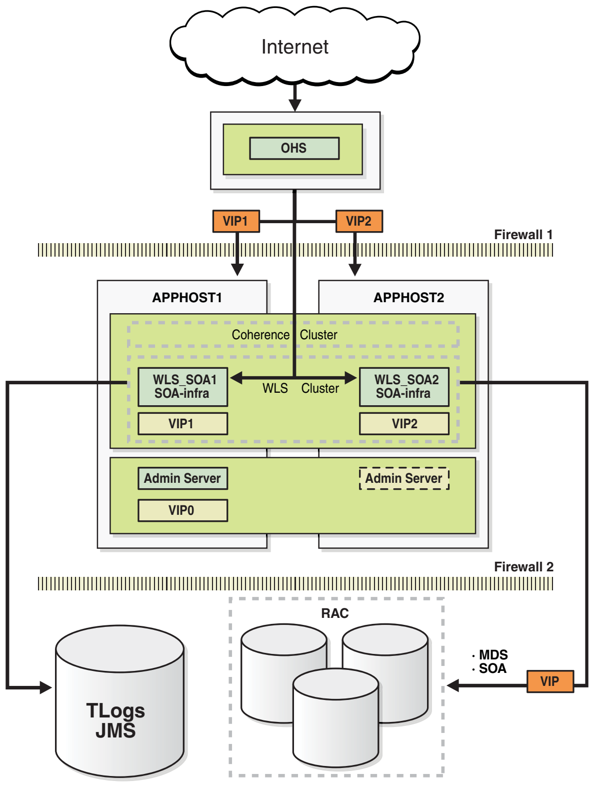


Figure 5–6 illustrates the following main characteristics of this high availability configuration:

The SOA Service Infrastructure runs in a Oracle WebLogic Server managed servers that are part of an Oracle WebLogic Server cluster. Oracle WebLogic Server cluster synchronizes the configuration for common artifacts of WebLogic Server cluster used by Oracle SOA Service Infrastructure, for example, JTS configuration, data sources, and persistent store definitions.

Oracle SOA Service Infrastructure uses the front end host and port information configured for Oracle WebLogic Server cluster as the server and callback URL. This information is defined using Oracle WebLogic Server Administration Console. To define this information, select **Clusters**, **SOA_Cluster_Name**, **HTTP/HTTPS frontend host** and then **Port**. If there is no address specified for Oracle WebLogic Server cluster where Oracle SOA Service Infrastructure is running, the system uses the physical host name as the server and callback URL.

For SOA high availability installations frontended by Oracle HTTP Server, monitoring should be done on the Listen ports of Oracle HTTP Server. This is the case when a deployment is using all the components deployed to the SOA Managed Server. A simple HTTP monitor that pings the HTTP/HTTPS port and expects a pre-determined response in turn should suffice. If only a specific SOA component is being used (such as B2B), then a monitor that does a deeper level check all the way to the Managed server can be considered to validate the health of the component in use. Please check with your load balancer vendor on setting up the HTTP monitors with your load balancer.

For more information about the server and callback URLs see the *Oracle Fusion Middleware Administrator's Guide*. Changing the HTTP Front End address for a cluster requires a restart of the managed servers that are part of the cluster.

The deployment coordinator is configured and used for deployment and updates of composites. The deployment coordinator sends notifications to the members of the deployment coordinator cluster to retrieve new artifacts from the MDS repository, when they are updated by the group leader. A leader node performs singleton operations for the cluster, such as updating the MDS after deployments, or changes are made to the composites.

Oracle SOA Service Infrastructure system uses Oracle WebLogic Server cluster name as its key to confirm a deployment coordinator group. If all nodes in a WebLogic Server cluster can communicate (over multicast or unicast) the deployment coordinator cluster is the same as the WebLogic Server cluster in which the SOA Service Infrastructure runs.

The Administration Server runs in Active-Passive mode. Whenever a failure occurs in APPHOST1, the Admin Server can be restarted in APPHOST2. Therefore it uses a virtual IP or virtual hostname as listen address.

For information about configuring virtual IPs for the administration server and configuring the administration server for high availability, see Chapter 10.2.2.3, "Transforming the Administration Server for Cold Failover Clusters."

Oracle WebLogic Server Whole Server Migration

Although SOA Service Infrastructure can run in full active-active mode, the architecture uses Oracle WebLogic Server Migration feature to protect some SOA components against failures. This implies that each of the WebLogic Managed Server in which the SOA Service Infrastructure runs is listening on a Virtual IP that is migrated to another box upon failover. The Administration Server and Enterprise Manager run in AdminServer, not the managed server.

As shown in Figure 5–6, WLS_SOA1 listens on VIP1, and WLS_SOA2 listens on VIP2. Each managed server uses the other node as a failover resource, therefore, the system is configured in a cross manner. In other words, WLS_SOA1 fails over to APPHOST2, and WLS_SOA2 fails over to APPHOST1. The appropriate capacity planning must be done to anticipate the scenario where the two SOA managed servers are running on the same node. For more information on Server Migration features, see Chapter 3, High Availability for WebLogic Server in this guide.

To resume transactions after a server migration, configure the transaction and JMS stores in a shared storage. In case of failure in one of the server infrastructure instances, other instances can resume transactions and JMS operation by reading the persistent stores from that shared storage.

The Metadata store is configured in an Oracle RAC database to protect from database failures. Similarly, the SOA process state information is also stored in an Oracle RAC database. In this example, both Oracle RAC databases are the same.

About Oracle SOA Service Infrastructure Components

These high availability characteristics apply to most of Oracle SOA components contained in the composite applications deployed across the cluster. For specific two-node high availability characteristics of the individual components, see the specific component sections the follow in this chapter.

5.2.2.1 Oracle SOA Service Infrastructure Protection from Failures and Expected Behavior

This section describes how an Oracle SOA high availability cluster deployment protects components from failure. This section also describes expected behavior in the event of component failure.

The SOA Service Infrastructure system is protected from all process failures by the WebLogic Server infrastructure.

5.2.2.1.1 WebLogic Server Crash If a WLS_SOAx server crashes, Node Manager attempts to restart it locally. If repeated restarts fail, the WebLogic Server infrastructure attempts to perform a server migration of the WLS_SOAx server to the other node in the cluster. While the failover takes place, the other SOA Service Infrastructure instance becomes the leader for deployments and composite updates and provides the basic services required by the service engines in the system.

Ongoing requests from the HTTP Server time out and new requests are directed to the other WLS_SOAx server. Once the server's restart completes on the other node, Oracle HTTP Server resumes routing any incoming requests to the server. The migrated server reads MDS repository for any updates that might have taken place during restart, and joins the deployment coordinator cluster to listen for new updates. The migrated server also resumes any pending transactions from the transaction logs in shared storage.

In the server migration scenario, the service engines, such as Oracle BPEL PM and Oracle Mediator, are failed over together with the SOA Service Infrastructure. They do not re-issue any requests to the other SOA Service Infrastructure instances by themselves. They resume operations together with the SOA Service Infrastructure once failover is complete.

Oracle SOA Service Infrastructure application may be down due to failure in accessing resources, errors caused by the deployment coordinator infrastructure, or other issues unrelated to whether the managed server is running. Therefore, Oracle recommends administrators monitoring the soa-infra application for errors caused by the application in the managed server logs. For information about log file locations, see Section 5.2.1.6, "Oracle SOA Service Infrastructure Log File Locations".

5.2.2.1.2 Node Failure In the event of a node failure, server migration is triggered after the available server verifies the time stamp in the database leasing system. If the failed server was the deployment coordinator cluster master, the available server becomes the new master and the SOA Service Infrastructure remains available for deployment and for composite lifecycle. After the time stamp for leasing is verified, the Node Manager in the node that still remains available attempts to migrate the VIP used by the failed managed server, and restarts it locally. This effectively results in the SOA Service Infrastructure application having two instances running in the same node. For more information on the failover process, see Section 3.9, "Whole Server Migration".

Service engines are deployed to the container as a part of the Service Infrastructure application. These service engines contain all of the ear files and library calls.

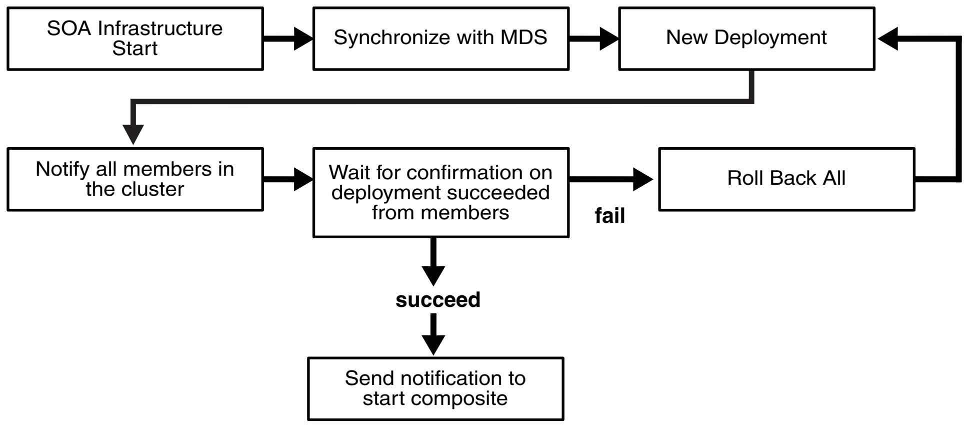
5.2.2.1.3 Database Failure The SOA Service Infrastructure system is protected against failures in the database by using multi data sources. These Multi Data sources are typically configured during the initial set up of the system (Oracle Fusion Middleware Configuration Wizard allows you to define these multi-pools directly at installation time) and guarantee that when an Oracle RAC database instance fails, the connections are reestablished with available database instances. The Multi Data source allows you to configure connections to multiple instances in an Oracle RAC database.

For information about multi data source configuration with Oracle RAC and the MDS repository, see Section 4.1.2, "Using Multi Data Sources with Oracle RAC."

5.2.2.2 Oracle SOA Service Infrastructure Cluster-Wide Deployment

As explained in previous sections, composite deployments are stored centrally by the SOA Service Infrastructure in the MDS repository. Each time the SOA Service Infrastructure is started, it synchronizes itself with the MDS repository and SOA store to get the deployment and process state. The deployment coordinator infrastructure orchestrates the notifications for composites deployments and updates. When a new deployment or update takes place, deployment coordinator notifies all members in the cluster. When all members in the cluster confirm that the deployment has succeeded, the master sends a notification to start the composite. If a deployment fails on any one of the nodes, it is rolled back to the rest of the cluster. An error message in the deployment coordinator master (WebLogic Server managed server), indicates the node on which the deployment failed. Figure 5–7 explains this process:

Figure 5–7 Cluster-Wide Deployment of Oracle SOA Composites


SOA Infrastructure Start
Synchronize with MDS
New Deployment
Notify all members in the cluster
Wait for confirmation on deployment succeeded from members
fail
Roll Back All
succeed
Send notification to start composite

5.2.2.3 Online Redeployment of Oracle SOA Service Infrastructure Composites in Cluster

When SOA Service Infrastructure performs an update or redeployment of a composite, it can overwrite an existing version or create a new version (v+1). All composites are uniquely identified based on the composite name and revision. Clients accessing a composite use by default, the version that is identified in the MDS repository (also visible from Oracle Enterprise Manager Fusion Middleware Control), as the default version. It is possible to manage the lifecycle of each version separately and perform online redeployments of composites, even with one single instance of SOA Service Infrastructure. The steps for performing this operation would be as follows:

1. Deploy version x.
2. Mark version x as default.
3. Deploy a new version of the composite (x+1) and mark it again `version x` as default (this is done to elude people accessing the new version in default access).
4. Test the new version by accessing from test client specifying the version of the composite (x+1)
5. Once verifications are complete, mark x+1 as the default version
6. New clients are routed to version x+1, old clients complete their work in version x

It is possible to leave the previous version deployed, or undeploy it. If you undeploy a composite which has in flight instances, they become stale and do not complete. The un-deployment of a composite removes the composite from the MDS repository.

5.2.2.4 Oracle SOA Service Infrastructure Cluster-Wide Configuration Changes

The standard Java EE artifacts that SOA Service Infrastructure uses are configured as part of Oracle WebLogic Domain in which SOA is installed. Oracle WebLogic Clusters provide automatic configuration synchronization for artifacts such as data-sources, persistent stores, and JMS modules, across the WebLogic Server domain. At the same time, the WebLogic Server cluster is in charge of synchronizing the deployments and libraries used by the SOA Service Infrastructure.

As explained in the single instance section, SOA Service Infrastructure-specific aspects are configured individually per WebLogic Server domain directory (typically one per node) through the soa-infra-config.xml file. This file is located in the `DOMAIN_HOME/config/soa-infra/configuration` directory.

This file is included in the jar file that is created when using Oracle WebLogic Server pack utility, therefore, it is propagated to other nodes when users run the pack/unpack procedure to set up high availability for SOA Service Infrastructure. However, ongoing changes to `soa-infra-config.xml` are not automatically replicated to the other nodes. Subsequently, you must make modifications in each node. An example of this would be updating the serverURL, or callbackURL fields for a SOA System. This modification needs to be manually performed in all the domain directories across a high availability topology. Users can edit the `soa-infra-config.xml` file directly, or use Oracle Enterprise Manager Fusion Middleware Control to make the configuration change.

If you are using Oracle SOA Suite in a clustered environment, any configuration property changes you make in Oracle Enterprise Manager on one node must be made on all nodes. Configuration properties are set in Oracle Enterprise Manager through the following options of the SOA Infrastructure menu:

Administration > **System MBean Browser**

SOA Administration > any property selections

5.3 Oracle BPEL Process Manager and High Availability Concepts

The information in this section guides you through the issues and considerations necessary for configuring Oracle BPEL PM for high availability.

5.3.1 Oracle BPEL Process Manager Single-Instance Characteristics

Service engines are containers that host the business logic of service components in a SOA composite application. Each service component, such as Oracle BPEL PM, Oracle Human Workflow, Decision Service, or Oracle Mediator, is executed in its own service engine (Decision Service execute in the business rules service engine). A service engine plugs into Oracle SOA Service Infrastructure. Oracle BPEL Process engine is the service engine running in SOA Service Infrastructure that allows the execution of BPEL Processes.

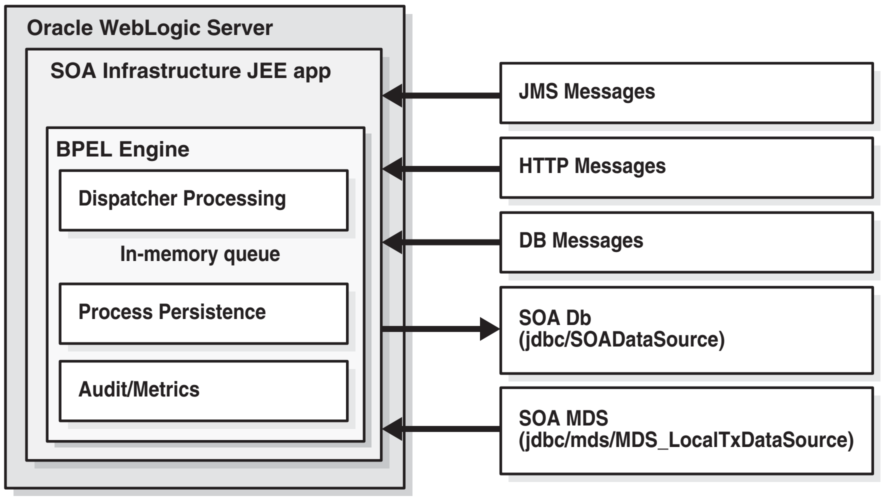
A BPEL process provides the standard for assembling a set of discrete services into an end-to-end process flow, and developing synchronous and asynchronous services into end-to-end BPEL process flows. It provides process orchestration and storage of long running, asynchronous processes.

As illustrated in Figure 5–8, the BPEL engine is a stateless part of Oracle SOA Service Infrastructure. It is started with the SOA Service Infrastructure application and contains different modules that take care of different aspects required to execute BPEL processes. If a deployed composite contains a BPEL process, SOA Service Infrastructure invokes BPEL Process Manager to get the component from the MetaDataStore (MDS).

BPEL Process Manager uses a Dispatcher Module that maintains an in-memory logical queue containing units of work to process the incoming messages from binding components (JMS, database, Web server).

The BPEL Process Manager engine saves processes' execution state in the SOA database though a persistence module based on Oracle Toplink. The Audit framework Continuously audits the work being processed by storing process execution information in the SOA database.

Figure 5–8 Oracle BPEL PM Single-Instance Architecture



5.3.1.1 BPEL Process Manager Component Characteristics

The BPEL Service Engine runs inside the SOA Service Infrastructure Java EE application (soa-infra.ear). The BPEL Service Engine does not have any singleton services. All state associated with a BPEL process is stored in a database (dehydration store) and there is no in-memory state replication required.

The processing of work by SOA Service Infrastructure Java EE EJBs is transactional and relies on Oracle WebLogic Service transaction control service. You configure the appropriate transaction stores as recommended in the basic WebLogic Server guidelines to guarantee recovery across failures in the WebLogic Server container. Additionally the BPEL engine system does not contain any Web modules, therefore, session replication is not required in the servlet layer when running BPEL in active-active.

In this release of Oracle Fusion Middleware, BPEL Service Container does not rely on JMS for asynchronous message dispatching. Therefore, there is no dependency on a distributed JMS infra-structure.

External Dependencies

The BPEL engine system depends on the following components:

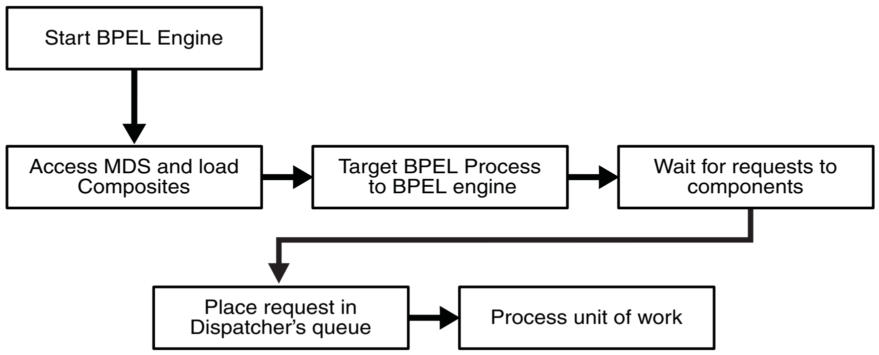
- SOA Service Infrastructure database for BPEL process state persistence
- MDS repository for BPEL process metadata store

Both components must be available for the BPEL engine system to start and run properly.

5.3.1.2 Oracle BPEL Process Manager Startup and Shutdown Lifecycle

As illustrated in Figure 5–9, when Oracle SOA Service Infrastructure application starts, it initializes the BPEL engine and loads the composites from the MDS repository. If the composite contains any BPEL processes, it targets those individual components to the BPEL engine. Once the process is loaded, the system is available to receive requests. At runtime, the BPEL engine waits for requests from different channels, such as JMS, the database, and HTTP.

Figure 5–9 Startup and Shutdown Lifecycle of Oracle BPEL PM


Start BPEL Engine
Access MDS and load Composites
Target BPEL Process to BPEL engine
Wait for requests to components
Place request in Dispatcher's queue
Process unit of work

A detailed startup and shutdown lifecycle is as follows:

1. Start SOA Server.
2. Start BPEL Engine.
3. Composites are loaded from MDS repository by SOA Service Infrastructure.
4. BPEL components are dispatched to the BPEL engine to be loaded.

5. Composite-binding components are activated.
6. The BPEL engine services requests.
7. The shutdown signal is received by SOA Service Infrastructure.
8. SOA Service Infrastructure starts undeploying loaded composites.
9. Composite-binding components are disabled.
10. BPEL components are dispatched to the BPEL engine to be unloaded.
11. The BPEL engine shuts down.

5.3.1.3 Oracle BPEL Process Manager Request Flow and Recovery

Recoverable activities are activities that have failed and can be recovered. For example, if you are using the file adapter to initiate an asynchronous BPEL process and your system crashes while the instance is processing, you can manually perform recovery when the server restarts, to ensure that all message records are recovered.

There are 2 types of BPEL Processes based on the invocation interface:

- **One-Way**: Most commonly asynchronous fire-and-forget pattern.
- **Two-Way**: Synchronous request-response pattern.

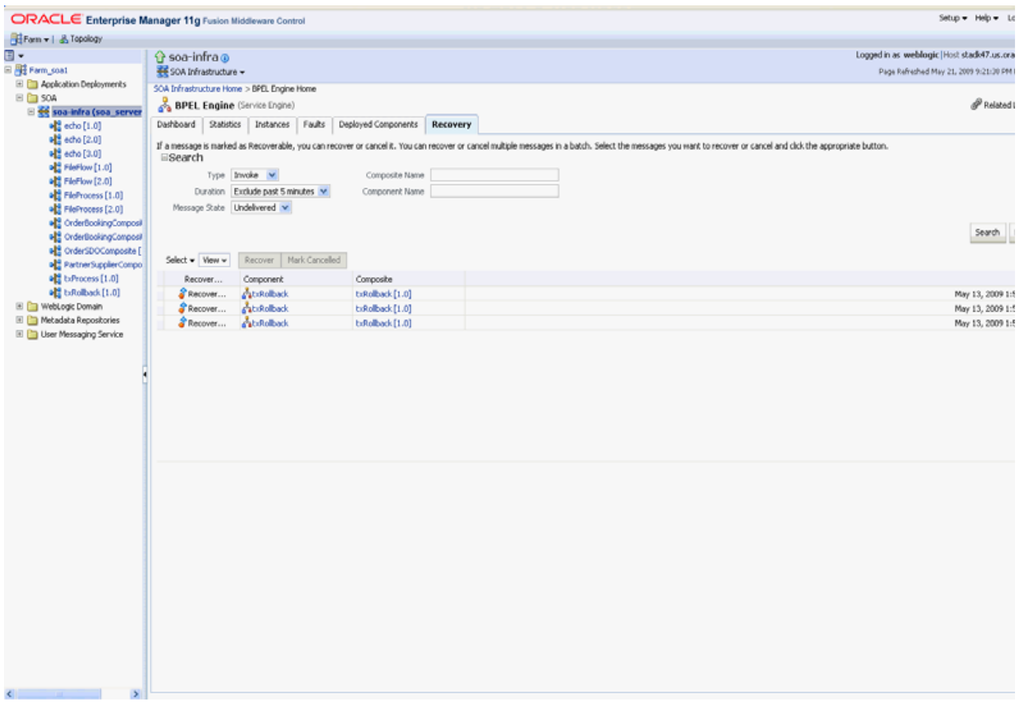
The recovery semantics after a server failure are based on whether the process is invoked synchronously and asynchronously. The following describes the behavior based on invocation and process type:

- **Synchronous Invocation (Request)**: For synchronous requests, an error is thrown back to the client in the case of a server failure. It is the client's responsibility to handle the error message and take appropriate action such as retrying the request. This holds true for both Transient and Durable processes.

  Note: In the case of durable processes, the message is persisted to the dehydration store at certain points. It is possible to recover the message from the dehydration store and replay the process using Enterprise Manager, however, the client cannot be notified of the response. Therefore, this is not a recommended option. It is preferable to handle all recovery from the client.

- **Asynchronous Invocation (Post)**: There are two types of asynchronous invocations - one that starts a new process and one that is a *callback* to an existing process. In the case of a callback, the engine (after recognizing that it is a continue operation) attempts to resolve the subscribing process, first by conversation ID, and then by a correlation set if defined. Messages associated with asynchronous requests are persisted to the dehydration store as part of the client call. If there is a failure prior to the persistence, the client receives an error message and has to handle the error in the same way as an error for a synchronous invocation. If the persistence is successful, the client call returns and further processing is done outside the context of the client call. In the case of a server failure, one-way processes that were invoked asynchronously can be restarted using Enterprise Manager.

Figure 5–10 Enterprise Manager BPEL Engine Recovery



Logged in as weblogic | Host stadk47.us.ora

Page Refreshed May 21, 2009 9:21:30 PM

If a message is marked as Recoverable, you can recover or cancel it. You can recover or cancel multiple messages in a batch. Select the messages you want to recover or cancel and click the appropriate button.

Type: Invoke
Duration: Exclude past 5 minutes
Message State: Undelivered
Composite Name:
Component Name:

Recover...	Component	Composite	
Recover...	txRollback	txRollback [1.0]	May 13, 2009 1:5
Recover...	txRollback	txRollback [1.0]	May 13, 2009 1:5
Recover...	txRollback	txRollback [1.0]	May 13, 2009 1:5

See the *Oracle Fusion Middleware Administrator's Guide for Oracle SOA Suite* and Developers Guide for details on the synchronous and asynchronous models supported by BPEL engine.

5.3.1.4 Oracle BPEL Process Manager Configuration Artifacts

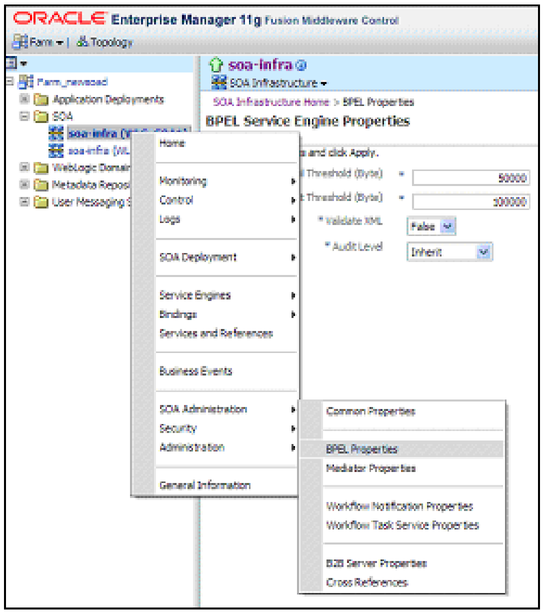
The main controls of Oracle Fusion Middleware BPEL engine are configured through the `bpel-config.xml` file located in the `DOMAIN_HOME/config/soa-infra/configuration` directory. This file allows specifying, among other properties, the following aspects:

- Number of Dispatcher Threads
- Timeout period for synchronous process wait.
- Number of retries for invocations that expire

Some of these parameters are configurable from Oracle Enterprise Manager Fusion Middleware Control. These properties are specific to each WebLogic domain directory that contains WebLogic Servers running Oracle Fusion Middleware SOA. Other configuration options at the container level, such as data sources, JTA configuration, and persistent stores are maintained as part of the WebLogic Server Domain configuration, and are synchronized across a cluster of WebLogic Servers by the WebLogic Server core infrastructure. See the *Oracle Fusion Middleware Administrator's Guide for Oracle SOA Suite* for more details on the configuration for the BPEL engine.

Figure 5–11, Shows Oracle WebLogic Server Administration Console where you can configure Oracle BPEL PM properties

Figure 5–11 Oracle BPEL PM Configuration Properties



s and click Apply.

l Threshold (Byte) * 50000

t Threshold (Byte) * 100000

* Validate XML False

* Audit Level Inherit

If you are using Oracle SOA Suite in a clustered environment, any configuration property changes you make in Oracle Enterprise Manager on one node must be made on all nodes. Configuration properties are set in Oracle Enterprise Manager through the following options of the SOA Infrastructure menu:

Administration > **System MBean Browser**

SOA Administration > any property selections.

5.3.2 Oracle BPEL Process Manager High Availability Architecture and Failover Considerations

Figure 5–6 describes a Oracle SOA Service Infrastructure two-node cluster running on two WebLogic Servers. Oracle BPEL PM is deployed as part of Oracle SOA Service Infrastructure.

5.3.2.1 Oracle BPEL Process Manager Protection from Failures and Expected Behavior

For information about Oracle SOA Service Infrastructure protection from failures and expected behavior, see Section 5.2.2.1, "Oracle SOA Service Infrastructure Protection from Failures and Expected Behavior".

The BPEL engine system is protected from all process failures by the WebLogic Server infrastructure. The following process failure considerations apply to Oracle BPEL:

- If the managed servers crash, Node Manager attempts to restart them locally. If Whole Server Migration is configured and repeated restarts fail, the WebLogic Server infrastructure attempts to perform server migration of the managed server to the other node in the cluster, if it is configured. Once the server on the other node is restarted, Oracle HTTP Server resumes routing any incoming requests to it. The migrated server reads the SOA database, resumes any pending processing, and resume transactions from the transaction logs in shared storage.
- The BPEL PM Service Engine or the entire SOA Service Infrastructure may be unavailable because of errors related to JDBC data-sources, or Coherence

configuration, even after a successful managed server startup. Therefore, it is necessary to monitor SOA Service Infrastructure logs for errors. For the location of server logs, see Section 5.2.1.6, "Oracle SOA Service Infrastructure Log File Locations".

In addition, for the handling of failures in the BPEL Engine itself with the WebLogic Server infrastructure, you can define and perform fault recovery actions on BPEL process faults identified as **recoverable** in Oracle Enterprise Manager Fusion Middleware Control. The recovery actions you perform on faults are based on actions you defined in your fault recovery policy files for BPEL process service components.

Three types of faults can be displayed in Oracle Enterprise Manager Fusion Middleware Control:

- Business: Application-specific faults that are generated when there is a problem with the information being processed (for example, a social security number is not found in the database).
- System: Network and other types of errors, such as a database server being completely unavailable, or a Web service being unreachable.
- Oracle Web Service Manager (OWSM): Errors on policies attached to SOA composite applications, service components, or binding components.

Fault recovery policies are defined at design time. The fault policies files included with the composites, define the actions to take should a failure occur. For more information on how to define and use Fault recovery, see the *Oracle Fusion Middleware Administrator's Guide for Oracle SOA Suite*.

Missing Instances

The inbound payload for one-way invocation requests are stored in the dehydration store and committed with the current global transaction context. If the caller has already started a global transaction, the invocation message is saved when the caller commits the transaction; if no incoming transaction is present, a new transaction is started to commit the invocation message. If a BPEL instance cannot be found in Enterprise Manager, the invocation message may be present in the manual recovery console (the recovery console lists all of the incoming messages that have not been delivered yet). If the invocation message is not found in the recovery console, check the status of the transaction from the caller side.

Transactional issues with endpoints

BPEL asynchronously saves the audit trail for instances that cannot be dehydrated within the global transaction in which the request was started. If any transactional services that are referenced from a BPEL component happen to rollback the transaction, the BPEL instance audit trail should still appear in Enterprise Manager. If there is a problem with the dehydration store, the entire global transaction is rolled back and no audit trail is saved. In this case, a message or activity can be recovered to pick up processing from the last known dehydrated point.

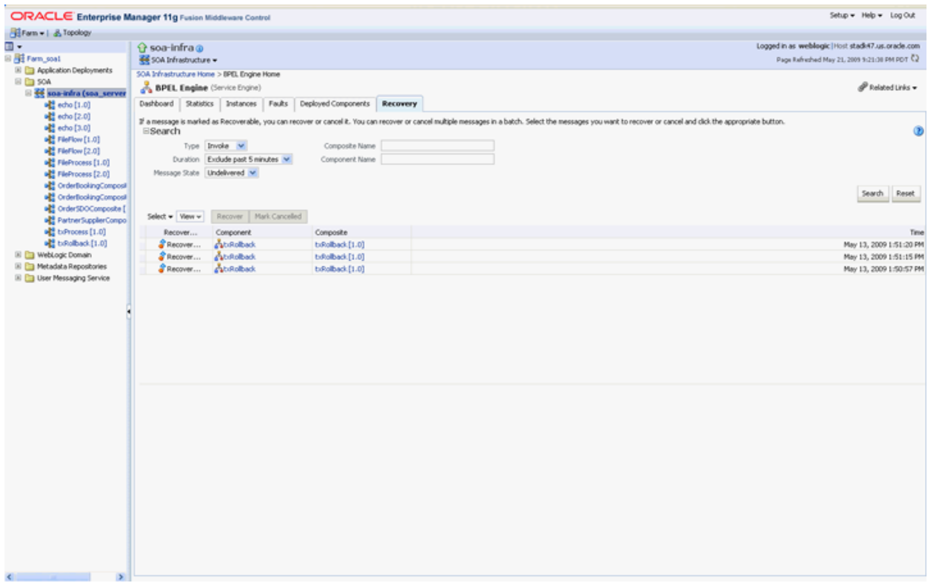
If the dehydration point is an Oracle RAC database with multiple nodes and the Java EE server on which the BPEL engine is hosted loses its JDBC connection to the Oracle RAC node, the BPEL engine attempts to retry the transaction (thereby rolling back whatever changes were made in the current transaction). If a new connection cannot be established to a new Oracle RAC node, the BPEL message or activity can be recovered using the recovery console.

Logging

BPEL engine loggers are set to INFO level by default. When trying to track down a message or instance, you may find it helpful to set loggers to TRACE level to enable more output to the WebLogic Server logs. For dehydration log messages, enable `oracle.soa.bpel.engine.data`. For thread or dispatcher log messages, enable `oracle.soa.bpel.engine.dispatcher`. To capture all engine log messages, enable `oracle.soa.bpel.engine`.

5.3.2.1.1 Recovering Failed BPEL and Mediator Instances In the case of a server failure, in-flight one-way processes that were invoked asynchronously may require manual recovery. Such processes are marked as **Recoverable** and can be restarted using Enterprise Manager as shown in Figure 5–12.

Figure 5–12 BPEL PM Instance Recovery using Oracle Enterprise Manager



Logged in as weblogic | Host stadk47.us.oracle.com
Page Refreshed May 21, 2009 9:21:38 PM PDT

SOA Infrastructure Home > BPEL Engine Home

BPEL Engine (Service Engine)

If a message is marked as Recoverable, you can recover or cancel it. You can recover or cancel multiple messages in a batch. Select the messages you want to recover or cancel and click the appropriate button.

Search

Type: Invoke
Duration: Exclude past 5 minutes
Message State: Undelivered
Composite Name:
Component Name:

Recover...	Component	Composite	Time
Recover...	txRollback	txRollback [1.0]	May 13, 2009 1:51:20 PM
Recover...	txRollback	txRollback [1.0]	May 13, 2009 1:51:15 PM
Recover...	txRollback	txRollback [1.0]	May 13, 2009 1:50:57 PM

5.3.2.2 Oracle BPEL Process Manager Cluster-Wide Configuration Changes

The standard Java EE artifacts that BPEL engine uses are configured as part of Oracle WebLogic Domain in which SOA is installed. Oracle WebLogic Clusters provide automatic configuration synchronization for artifacts, such as data-sources, persistent stores, and JMS modules across the WebLogic Server domain.

As explained in the single instance section, BPEL engine-specific configuration is stored in the bpel-config.xml file located in the domain directory associated with each managed server (typically one domain directory per node).

5.4 Oracle Mediator and High Availability Concepts

The information in this section guides you through the issues and considerations necessary for configuring Oracle Mediator for high availability.

5.4.1 Oracle Mediator Singe-Instance Characteristics

Oracle Mediator is a service engine within the Oracle SOA Service Infrastructure. Oracle Mediator provides the framework to mediate between various providers and consumers of services and events. The Mediator service engine runs in-place with the

SOA Service Infrastructure Java EE application. The runtime state for execution started by asynchronous interactions or involving parallel routing rules is maintained in the SOA runtime database. For details about administering Oracle Mediator, see the *Oracle Fusion Middleware Administrator's Guide for Oracle SOA Suite*.

- Synchronous and Asynchronous Interactions

  Oracle Mediator provides support for synchronous and asynchronous request-response interactions. In a synchronous interaction, the client making the request remains blocked, awaiting the response. In an asynchronous interaction, the client invokes the service but does not wait for the response. You can specify a time out period for an asynchronous interaction.

- Sequential and Parallel Message Routing

  Oracle Mediator can route messages to their destinations either sequentially, or in parallel.

  – In a sequential routing, processing of data takes place in one single transaction.

  – In a parallel routing scenario, one transaction is used for en-queueing information, and another one for de-queuing it.

5.4.1.1 Oracle Mediator Component Characteristics

If a composite contains a Mediator component, SOA Service Infrastructure targets the component to the Mediator engine for deployment. None of the services provided by the Mediator engine system are singletons, therefore, Oracle Mediator engines can run in full active-active mode. The processing of messages by the worker threads in Oracle Mediator is transactional and relies on Oracle WebLogic Server transaction control service. Configure the appropriate transaction stores as recommended by WebLogic Server guidelines to guarantee recovery across failures in the WebLogic Server container. Additionally, Oracle Mediator's engine does not contain any stateful Web modules or stateful session beans, therefore you are not required to configure any sort of session replication when running Oracle Mediator in active-active mode. The state of work and work-to-be processed is maintained by Oracle Mediator in the database. Therefore, it is critical that Oracle Mediator's database be highly available. This requires configuring multi data sources for the SOA data source as described in Section 5.11.4, "Running Oracle Fusion Middleware Configuration Wizard on APPHOST1 to Create the SOA Domain". For information about multi data source configuration with RAC and the MDS repository, see Section 4.1.2, "Using Multi Data Sources with Oracle RAC."

External Dependencies

Oracle Mediator depends on the following components:

- SOA database for Mediator message and message state persistence
- MDS repository for composite metadata store

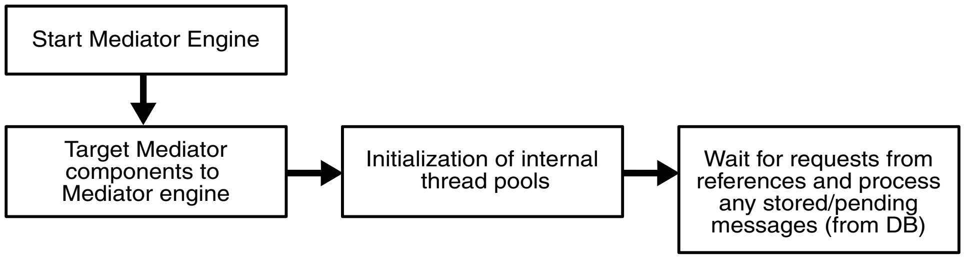
Both components must be available for Oracle Mediator to start or run properly.

5.4.1.2 Oracle Mediator Startup and Shutdown Lifecycle

When the Oracle SOA Service Infrastructure application starts, it initializes the Oracle Mediator engine and loads the composites from the MDS repository. If the composite contains any Oracle Mediator components, it targets them to the Oracle Mediator engine. At runtime, Mediator routing rules can be invoked through an inbound binding component or by another service engine. Graceful shutdown of the Mediator engine is initiated by the SOA Service Infrastructure and involves sending signals to in-flight instances and unloading of loaded components.

Figure 5–13 illustrates Oracle Mediator Startup lifecyle.

Figure 5–13 Oracle Mediator Startup Lifecyle


Start Mediator Engine
Target Mediator components to Mediator engine
Initialization of internal thread pools
Wait for requests from references and process any stored/pending messages (from DB)

5.4.1.3 Oracle Mediator Request Flow

The recovery semantics of Oracle Mediator after a server failure are based on client interaction type and routing rule type.

Oracle Mediator provides support for synchronous and asynchronous request-response interactions. The following describes the behavior based on interaction type:

- Synchronous Interaction: For synchronous interactions, an error is thrown back to the client in the case of a server failure. It is the client's responsibility to handle the error message and take appropriate action such as retrying the request.
- Asynchronous Interaction: There are two types of asynchronous invocations - one that starts a new routing rule execution and one that is a callback to an existing rule execution. In the case of a callback, the engine attempts to resolve the subscribing instance through a correlation id. If there is a failure in handling the callback, the client receives an error message and has to handle the error appropriately. The client invocation must be transactional in order to guarantee reliable handling of a callback in the case of a server failure.

Oracle Mediator can route messages either sequentially or in parallel. Only messages processed in parallel need manual recovery in the case of a server failure. The following describes the behavior based on routing rule type:

- Sequential Routing Rule: For sequential routing rule, complete rule is executed in a single transaction. In the case of server failure, Mediator relies on the underlying transaction manager for recovery.
- Parallel Routing Rule: For parallel routing rule, mediator uses store-and-forward paradigm that involves two separate transactions - one transaction for persisting the message and a second one for executing the routing rule. If there is a failure prior to the persistence, the client receives an error message and has to handle the error in the same way as in a sequential routing rule. If the persistence is successful, the client call returns and further processing is done outside the context of the client call. In the case of a server failure, Mediator will initiate recovery upon server restart after a configurable time interval specified by `ContainerIdLeaseRefresh`

5.4.1.4 Oracle Mediator Configuration Artifacts

The main controls of the Oracle Mediator engine are configured through the `mediator-config.xml` file located in the *DOMAIN_*

HOME/config/soa-infra/configuration directory. Relevant properties for high availability specified by this file include:

- Batch size of rows locked for Parallel Routing
  - `DeferredMaxRowsRetrieved`
- Rate of heartbeat messages across Oracle Mediator instances
  - `ContainerIdLeaseRefresh`
  - `ContainerIdLeaseTimeout`
- Audit level

You can configure some of these parameters from the Oracle Enterprise Manager Fusion Middleware Control. These properties are specific to each WebLogic domain directory that contains WebLogic Servers running Oracle Fusion Middleware SOA.

5.4.2 Oracle Mediator High Availability Architecture and Failover Considerations

Mediator being an embedded service engine leverages the same high availability architecture as the SOA Service Infrastructure. Figure 5–6 describes a two-node Oracle SOA Service Infrastructure cluster running on two WebLogic Servers. Oracle Mediator is deployed as part of Oracle SOA Service Infrastructure composite application.

In a clustered configuration, Oracle Mediator can run in an active-active mode as there are no singleton services and all state is stored in the SOA runtime database.

5.4.2.1 Oracle Mediator Protection from Failures and Expected Behavior

For information about Oracle SOA Service Infrastructure protection from failures and expected behavior, see Section 5.2.2.1, "Oracle SOA Service Infrastructure Protection from Failures and Expected Behavior."

During the execution of a Parallel Routing Rule, Mediator obtains a logical lock on a batch of messages in the SOA runtime database. This lock contains a reference to the container ID that uniquely identifies the Oracle Mediator engine. The container ID is assigned at startup time. The user can configure the batch size using `DeferredMaxRowsRetrieved`. A smaller batch size ensures that there are a lower number of locked rows requiring recovery (as explained above) in the case of a server failure.

In case of unplanned outages, you must wait as much time as specified as the `ContainerIdLeaseRefresh` interval, after restarting the server. This enables the server to complete the instances still in the running state.

In a multi-node cluster environment, if Mediator is used for an asynchronous message exchange pattern, there could be a possibility that the callback will not be handled by Mediator if it arrives before the request has completed. This could happen in a scenario where the request initiated by Mediator to the target service takes longer to complete and callback from the target service arrives before the request completes.

Process Failure

Oracle Mediator is protected from all process failures by the WebLogic Server infrastructure. The following process failure considerations apply to Oracle Mediator:

- If the managed servers crash, Node Manager attempts to restart them locally. If Whole Server Migration is configured and repeated restarts fail, the WebLogic Server infrastructure attempts perform server migration of the managed server to the other node in the cluster, if it is configured. Once the server on the other node is restarted, Oracle HTTP Server resumes routing any incoming requests to it. The migrated server read the SOA database, resume any pending processing, and resume transactions from the transaction logs in shared storage.

- The SOA Service Infrastructure application where Mediator Engine runs may be down due to failure in accessing resources, errors caused by the coherence infrastructure, or other issues unrelated to the status of the managed server where it is located. Therefore, monitor Oracle SOA Service Infrastructure application to watch for errors caused by the application in the managed server logs. For the location of server logs, see Section 5.2.1.6, "Oracle SOA Service Infrastructure Log File Locations").
- During recovery after a server crash, Oracle Mediator attempts to redeliver messages that were partially processed. If the automatic retry fails, Oracle Mediator enqueues the message to the error hospital for manual recover.

Node Failure

In case of a node failure, server migration is triggered after the available server verifies the time stamp in the database leasing system. If the crashed server was the coherence cluster master, the available server become the new master, and the Mediator Engine remains available for processing messages from binding components. After the time stamp for leasing is verified, the Node Manager in the remaining node attempts to migrate the VIP used by the failed managed server, restarts it locally. This effectively result in the Mediator Engine having two instances running in the same node.

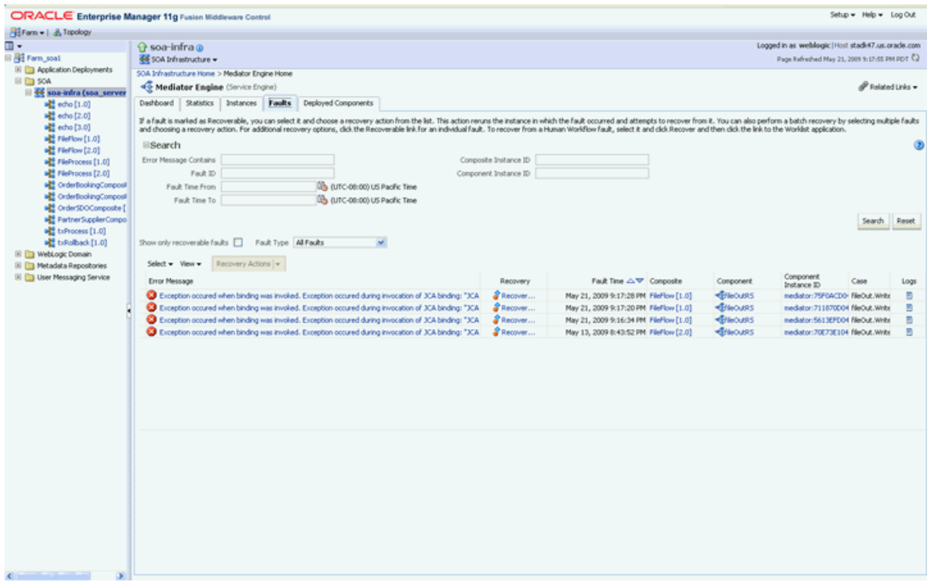
Cluster-Wide Configuration Changes

The standard Java EE artifacts that Oracle Mediator engine uses are configured as part of Oracle WebLogic's domain in which SOA is installed. Oracle WebLogic Clusters provide automatic configuration synchronization for artifacts, such as data-sources, persistent stores, and JMS modules across the WebLogic Server domain. At the same time, the WebLogic Server cluster is in charge of synchronizing the deployments, libraries used by Oracle Mediator's engine.

As explained in the single instance section, Oracle Mediator engine-specific aspects are configured individually per WebLogic Server domain directory (typically one per node) through the `mediator-config.xml` file. this file is located in the `DOMAIN_HOME/config/soa-infra/configuration` directory.

5.4.2.1.1 Recovering Failed Mediator Instances During recovery from a server failure, under certain circumstances, messages with in-flight routing rules may be placed in the error hospital for manual recovery. These messages can be recovered using Enterprise Manager as shown in Appendix 5–14.

Figure 5–14   Mediator Instance Recovery using Oracle Enterprise Manager



If a fault is marked as Recoverable, you can select it and choose a recovery action from the list. This action reruns the instance in which the fault occurred and attempts to recover from it. You can also perform a batch recovery by selecting multiple faults and choosing a recovery action. For additional recovery options, click the Recoverable link for an individual fault. To recover from a Human Workflow fault, select it and click Recover and then click the link to the Worklist application.

Error Message	Recovery	Fault Time	Composite	Component	Component Instance ID	Case	Logs
Exception occured when binding was invoked. Exception occured during invocation of JCA binding: "JCA	Recover...	May 21, 2009 9:17:28 PM	FileFlow [1.0]	fileOutRS	mediator:75F0ACD0	fileOut.Write	
Exception occured when binding was invoked. Exception occured during invocation of JCA binding: "JCA	Recover...	May 21, 2009 9:17:20 PM	FileFlow [1.0]	fileOutRS	mediator:711870D04	fileOut.Write	
Exception occured when binding was invoked. Exception occured during invocation of JCA binding: "JCA	Recover...	May 21, 2009 9:16:34 PM	FileFlow [1.0]	fileOutRS	mediator:5613EFD04	fileOut.Write	
Exception occured when binding was invoked. Exception occured during invocation of JCA binding: "JCA	Recover...	May 13, 2009 8:43:52 PM	FileFlow [2.0]	fileOutRS	mediator:70E73E104	fileOut.Write	

5.4.2.2 Troubleshooting Oracle Mediator High Availability

To debug Mediator failures, check the database tables to determine which container failed. To identify requests that were in progress when Mediator failed, find the rows that are still locked, and unlock them. You may also view the payload as it is stored as a blob

The poll interval for Mediator Instance Manager should be the same across all the nodes. The timed stamp used is that database time stamp.

The possible stages of messages in the database are:

- Ready
- Locked
- Completed
- Error

5.5 Oracle Human Workflow and High Availability Concepts

The information in this section guides you through the issues and considerations necessary for configuring Oracle Human Workflow for high availability.

5.5.1 Oracle Human Workflow Singe-Instance Characteristics

Oracle Human Workflow is a service engine running in Oracle SOA Service Infrastructure that allows the execution of interactive human driven processes. A human workflow provides the human interaction support such as approve, reject, and reassign actions within a process or outside of any process. The Human Workflow service consists of a number of services that handle various aspects of human interaction with a business process.

All human task metadata is stored and managed in the Metadata Service (MDS) repository. The Human Workflow engine consists of a Service Engine running within the SOA Service Infrastructure and additional Java EE applications for

DefaultToDoTaskFlow and Worklist applications. A human workflow leverages an internal JMS queue for notifications related to a human task.

For details about administering Oracle Human Workflow, see the *Oracle Fusion Middleware Administrator's Guide for Oracle SOA Suite*.

5.5.1.1 Oracle Human Workflow Startup and Shutdown Lifecycle

The Human Workflow startup consists of two phases, loading of the Java EE applications, and initialization of the service engine. The Java EE applications are loaded as part of the application server startup. The service engine initialization and shutdown is controlled by SOA Service Infrastructure. Post initialization, composites that contain Human Workflow components are targeted to the Human Workflow service engine by SOA Service Infrastructure.

5.5.1.2 Oracle Human Workflow Request Processing

A human workflow can be initiated by an invocation from another SOA engine (such as Oracle BPEL PM). The message is routed to the engine by SOA Service Infrastructure and is persisted by the Workflow engine within its runtime schema. The message becomes available for human actions through the browser based UI after the client transaction that is associated with the invocation is committed. Each update on the message or the runtime state through a user action is a separate transaction.

As soon as Workflow commits its transaction, the control passes back to BPEL which almost instantaneously commits its transaction. Between this window, if the RAC instance goes down, on failover, the message is retried and can cause duplicate tasks.

5.5.1.3 Oracle Human Workflow Configuration Artifacts

The configuration parameters for Human Workflow are defined in the `workflow-config.xml`, `workflow-identity-config.xml` and `workflow-notification-config.xml` files located in the config/soa-infra/configuration directory under the domain root. Key configuration parameters include:

- taskAutoReleaseConfigurations
- worklistApplicationURL
- HWFMailerConfiguration

If you are using Oracle SOA Suite in a clustered environment, any configuration property changes you make in Oracle Enterprise Manager on one node must be made on all nodes. Configuration properties are set in Oracle Enterprise Manager through the following options of the SOA Infrastructure menu:

Administration > **System MBean Browser**

SOA Administration > any property selections.

5.5.1.3.1 Managing the URI of the Human Task Service Component Task Details Application You can add or remove the URI of the task details application used in human workflow.

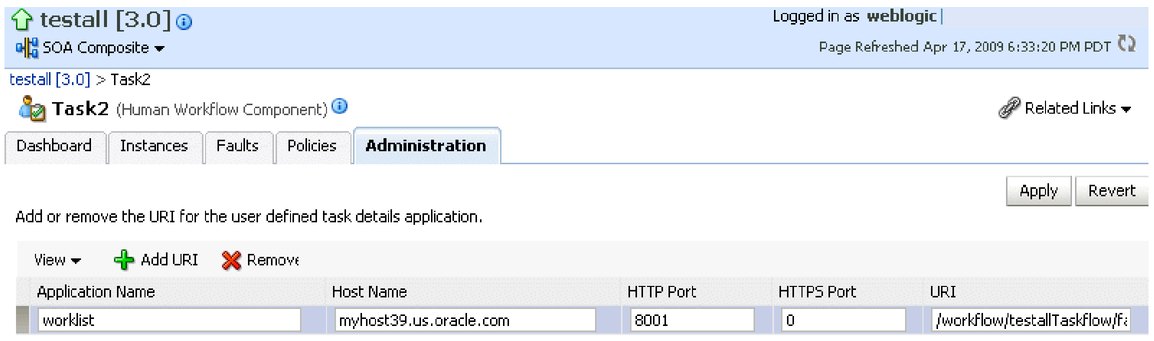
To manage the URI of the human task service component task details application:

1. Access this page through one of the following options:

From the SOA Infrastructure Menu...	From the SOA Folder in the Navigator...
1. Select **Home**. 2. Select the **Deployed Composites** tab. 3. In the **Composite** section, select a specific SOA composite application.	1. Under **soa-infra**, select a specific SOA composite application.

2. Select the human task service component in the **Component Metrics** table.
3. Click **Administration**.

   The Administration page shows the URI for the task details application.



Add or remove the URI for the user defined task details application.

Application Name	Host Name	HTTP Port	HTTPS Port	URI
worklist	myhost39.us.oracle.com	8001	0	/workflow/testallTaskflow/fa

4. Click the **Add** icon to specify the following details for the URI:
   - Application name
   - Host name
   - HTTP port
   - HTTPS port (optional)
   - URI
5. Click **Apply**.

5.5.2 Oracle Human Workflow High Availability Architecture and Failover Considerations

Figure 5–6 describes a two-node Oracle SOA Service Infrastructure cluster running on two WebLogic Servers. Oracle Human Workflow is deployed as part of Oracle SOA Service Infrastructure composite application.

5.5.2.1 Oracle Human Workflow Protection from Failures and Expected Behavior

Oracle Human Workflow's engine uses transactional EJBs for persistence and JMS queues for user notification. All state is stored in the database and the JMS queue, and there is no in-memory session state to be replicated for recovery. Therefore, Oracle Human Workflow's service engine and the associated Java EE applications are run in an active-active topology on a WebLogic cluster. In the case of a server or hardware crash, whole server migration must be configured for recovering pending transactions and JMS messages stored on the local queue. Notifications are not sent out until the server is restarted.

5.5.3 Troubleshooting Oracle Human Workflow High Availability

Human Workflow works like a standard Java EE application with a browser based client. Once a message is persisted to its runtime schema, it can be processed using the workflow UIs. If a server crashes in the middle of the transaction, you must recover the server for successful recovery or rollback. The following loggers are relevant for debugging errors associated with human workflow:

- oracle.soa.services.common
- oracle.soa.services.notification
- oracle.soa.services.workflow
- oracle.soa.services.workflow.common
- oracle.soa.services.workflow.evidence
- oracle.soa.services.workflow.metadata
- oracle.soa.services.workflow.persistency
- oracle.soa.services.workflow.query
- oracle.soa.services.workflow.runtimeconfig
- oracle.soa.services.workflow.soa
- oracle.soa.services.workflow.task
- oracle.soa.services.workflow.task.dispatch
- oracle.soa.services.workflow.task.routing

5.6 Oracle B2B and High Availability Concepts

The information in this section guides you through the issues and considerations necessary for configuring Oracle B2B for high availability.

5.6.1 Oracle B2B Singe-Instance Characteristics

Oracle B2B connects SOA composite applications to external services, applications, and technologies. Oracle B2B offers a multi-protocol gateway that supports industry-recognized B2B standards. Oracle B2B extends Oracle SOA Suite with business protocol standards, such as electronic data interchange (EDI), ebXML, HL7, and RosettaNet. For details about Oracle B2B's functionality, see the *Oracle Fusion Middleware Administrator's Guide for Oracle SOA Suite*.

Oracle B2B is a binding component within SOA Service Infrastructure. B2B meta-data consists of trading partner details along with their supported documents and delivery channels which is stored within the MDS repository. The SOA composite has a reference to this meta-data.

Oracle B2B receives messages from different channels. A listener thread logs the messages in a table in the SOA database and sends the corresponding events to a JMS queue. Event handlers listen for events to process the requests in the JMS queues.

Oracle B2B supports the following transport protocols for communication with business entities:

- HTTP(S)
- Oracle Advanced Queue
- Email (SMTP 1.0, IMAP 1.0, POP3)

- File
- FTP and SFTP (SSH FTP)
- TCP/IP
- JMS

For HTTP protocol, Oracle B2B uses the system's HTTP listener. For FTP and Email, Oracle B2B uses the external FTP and email server configured in the system. Even in high availability mode, only one B2B instance polls for incoming message for FTP, File, and email protocols.

Note: MLLP & TCP/IP protocols are not supported in a clustered environment.

5.6.1.1 Oracle B2B Component Characteristics

Oracle B2B runs in-place with SOA Service Infrastructure Java EE application. Oracle B2B UI, is a Web application that is deployed as a standalone war file on the same managed server as the SOA Service Infrastructure. Oracle B2B UI application is stateful and stores information in the HTTP Session.

Oracle B2B stores state information within JMS queues and the SOA run-time database. Whole Server Migration is required for automatic JMS message and transaction recovery after a server failure.

Oracle B2B uses JMS intensively. Oracle B2B uses Oracle WebLogic Server Migration feature for protecting Oracle B2B JMS resources against failures. Oracle B2B's engine uses EJBs and Servlets, which are stateless. All state information is persisted in JMS queues and the database.

High availability of B2B depends on the high availability of the run-time database. Hence, the SOA runtime data sources should be configured as mulit-data sources for high availability. For information about multi data source configuration with RAC and the MDS repository, see Section 4.1.2, "Using Multi Data Sources with Oracle RAC."

External Dependencies

Oracle B2B depends on the following components:

- Oracle SOA database for Oracle B2B message and message state persistence
- MDS repository for Oracle B2B metadata store
- FTP and email servers if the corresponding adapters are used

The SOA database and the MDS repository must be available for Oracle B2B to start and run properly.

5.6.1.2 Oracle B2B Startup and Shutdown Lifecycle

When Oracle SOA Service Infrastructure application starts, it initializes Oracle B2B's engine. Oracle B2B meta-data deployment should occur before composites are deployed to the SOA Service Infrastructure. Oracle B2B end-points are defined as Channels which are started as part of the engine initialization. Based on the protocol, each configured channel has external dependencies, such as:

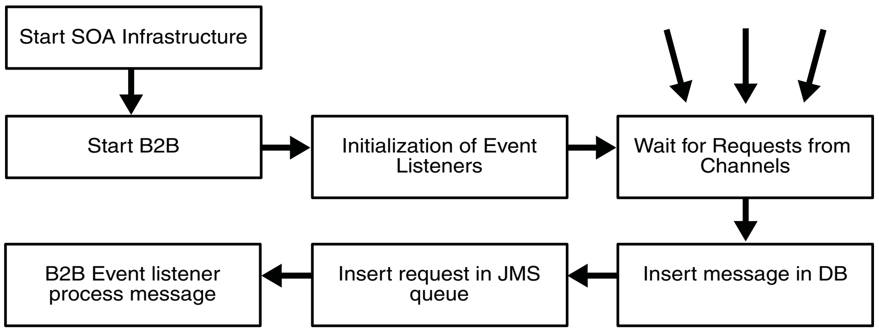
- If FTP is used, the appropriate FTP server must be started.
- If email is used, the appropriate email server must be started.

- If HTTP/HTTPS is used, the HTTP server front ending the B2B system must be started

5.6.1.3 Oracle B2B Request Flow

In an Oracle B2B system all processing is done asynchronously. When a request arrives, the corresponding message is inserted in the SOA database. A notification is inserted in a JMS queue as a placeholder for the pending work. The event handler threads listen for events in these queues and process the work sequentially.

Figure 5–15 Oracle B2B Request Flow


Start SOA Infrastructure
Start B2B
Initialization of Event Listeners
Wait for Requests from Channels
Insert message in DB
Insert request in JMS queue
B2B Event listener process message

5.6.1.4 Oracle B2B Configuration Artifacts

Oracle Enterprise Manager Fusion Middleware Control allows enabling and disabling metrics for Oracle B2B's Server. This is the only property exposed by Oracle Enterprise Manager Fusion Middleware Control. The property is also present in the `b2b-config.xml` file located in the `DOMAIN_HOME/config/soa-infra/configuration` directory. In this file you can also enter the JTA timeout value. For example:

```
<property>
              <name>b2b.jtaTimeout</name>
              <value>300</value>
              <comment>Set JTA Timeout to 300 seconds.</comment>
</property>
```

Most of the configuration for Oracle B2B's Server is maintained in the SOA database and exposed by Oracle B2B User Interface Application. For details about Oracle B2B configuration, see the *Oracle Fusion Middleware User's Guide for Oracle B2B*.

Certain configuration options, stored in the `b2b-config.xml` file need must be kept in sync manually across the cluster.

If you are using Oracle SOA Suite in a clustered environment, any configuration property changes you make in Oracle Enterprise Manager on one node must be made on all nodes. Configuration properties are set in Oracle Enterprise Manager through the following options of the SOA Infrastructure menu:

Administration > **System MBean Browser**

SOA Administration > any property selections.

5.6.2 Oracle B2B High Availability Architecture and Failover Considerations

Figure 5–6 describes a two-node Oracle SOA Service Infrastructure cluster running on two WebLogic Servers. Oracle B2B is deployed as part of Oracle SOA Service

Infrastructure composite application. The following high availability characteristics are specific to an Oracle B2B high availability deployment:

- Oracle B2B User Interface application runs inside each one of Oracle WebLogic Server servers, and as part of the same cluster as Oracle SOA Service Infrastructure application.
- Oracle B2B's server maintains the partners, documents, and channels definitions in the SOA database using an Oracle RAC database.

5.6.2.1 Oracle B2B Protection from Failures and Expected Behavior

This section describes how an Oracle B2B high availability cluster deployment protects components from failure. This section also describes expected behavior in the event of component failure.

Oracle B2B UI Failure

Oracle B2B's User Interface application maintains some navigation information in memory. When a failure happens in one of the managed servers running Oracle B2B's User Interface, users' requests are redirected to another active WebLogic Server running the application. Ongoing requests from Oracle HTTP Server time out according the time out setting in Oracle HTTP Server's configuration. Failover requires those users accessing the failed instance to re-login, since the application is not enabled for serializing the session information.

B2B UI in-memory data is non-transactional and some steps may need to be revisited in the case of failover.

For general information about Oracle SOA Service Infrastructure process failure, see Section 5.2.2.1, "Oracle SOA Service Infrastructure Protection from Failures and Expected Behavior"

Node Failure

If a node failure occurs, or if the local oracle WebLogic Server Node Manager reaches the maximum restart tries on the failed managed server, whole server migration is triggered after the available server verifies the time stamp in the database leasing system. The other oracle B2B engine remains available for processing new messages from the different channels. At the same time, the remaining Oracle B2B server should resume the pending operations for singleton channels, such as FTP, email, or file, after the default timeout period is reached in the time stamp the failed node sets in the database (two minutes by default). For Oracle B2B's User Interface application, ongoing requests from Oracle HTTP Server time out according to Oracle HTTP Server configuration.

Database Failure

For information about Oracle B2B database failure, see Section 5.2.2.1, "Oracle SOA Service Infrastructure Protection from Failures and Expected Behavior"

5.6.2.2 Oracle B2B Cluster-Wide Configuration Changes

The standard Java EE artifacts that Oracle B2B engine uses are configured as part of Oracle WebLogic's domain in which Oracle SOA Service Infrastructure is installed. Oracle WebLogic Clusters provide automatic configuration synchronization for artifacts, such as data-sources, persistent stores, and JMS modules across the WebLogic Server domain. At the same time, the WebLogic Server cluster is in charge of synchronizing the deployments and libraries used by Oracle B2B's engine.

In an active-active cluster, with Oracle B2B UI running on two or more nodes, the load balancer fronting the B2B UI persistence settings must be configured so that multiple requests (in the same session) from the client are directed to the same B2B node.

As explained in the single instance section, most of Oracle B2B server-specific configuration is maintained in the database and are applicable to all the B2B Servers running in a WebLogic Server domain. Therefore, configuration properties, such as Payload Size, and Outbound Dispatcher Counts are applied cluster-wide, meaning they are used by all instances in Oracle SOA's cluster.

Just the enabling of DMS metrics is configured individually per WebLogic Server domain directory through the `b2b-config.xml` file. this file is located in the `DOMAIN_HOME/config/soa-infra/configuration` directory. This file should be kept in sync manually across the cluster.

To set up file, FTP, or email transports in a high availability environment, specify a unique name for each instance in `b2b.HAInstance`. If you use `ServerName` for the value, Oracle B2B retrieves the WebLogic Server name as the `HAInstanceName`.

For information about setting up file, FTP, or email transports, see Section B.4, Setting Up File, FTP, or Email in an HA Environment, in the *Oracle Fusion Middleware User's Guide for Oracle B2B*

5.6.2.3 Oracle B2B deployments in a Cluster

Using the command line utility for deploying, purging, or importing metadata in B2B may cause inconsistencies and errors in the B2B system. For B2B, deploy agreements and purge or import metadata ONLY from the GUI available in the B2B console, instead of using the command line utility.

5.6.2.4 Troubleshooting Oracle B2B Active-Active Configuration

This section provides troubleshooting tips and possible resolutions for Oracle B2B active-active configurations.

5.6.2.4.1 Purge, Import, or Deployment of B2B Metadata When performing purge, import, or deployment of B2B metadata in a cluster, error timing and load balancing may cause exceptions that are unlikely to happen if a retry of the operation is performed.

There is no clean up nor other additional steps required. If errors, such as `[java] MDS-02202: Content of the metadata object` appear for deployment or "`postTransfer: MDS-00521: error while reading document...` appear for purge or import, retry the operation.

5.6.2.4.2 Error While Retrieving Oracle B2B Document Definitions **Problem:** Error happens when trying to retrieve a document definition XSD from Oracle B2B. B2B resides in a cluster and is accessed through a load balancer. B2B console report the following:

```
An error occured while loading the document definitions.
 java.lang.IllegalArgumentException: Cluster address must be set when clustering
 is enabled.
```

Solution: This occurs if you do not set the frontend HTTP host and port for the Oracle WebLogic cluster where Oracle B2B resides. To eliminate this error, set the front end address for the SOA Cluster:

1. In the WebLogic Server Administration Console, in the Change Center section, click **Lock & Edit**.

2. In the left pane, choose the **Environment in the Domain Structure** window and then choose **Clusters**. The Summary of Clusters page appears.
3. Select the `WLS_SOA` cluster.
4. Select **HTTP**.
5. Set the values for the following:
   - **Frontend Host:** `soa.mycompany.com`
   - **Frontend HTTPS Port:** `443`
   - **Frontend HTTP Port:** `80`
6. Click **Save**.
7. To activate the changes, click **Activate Changes** in the Change Center section of the Administration Console.
8. Restart the servers to make the Frontend Host directive in the cluster effective.

5.7 Oracle Web Service Manager and High Availability Concepts

The information in this section guides you through the issues and considerations necessary for configuring Oracle WSM for high availability.

5.7.1 Oracle WSM Single-Instance Characteristics

Oracle Web Services Manager (Oracle WSM) provides a policy framework to manage and secure Web services consistently across your organization. It provides capabilities to build, enforce, execute and monitor WebService policies including security, WSRM, MTOM and addressing policies. It typically gets deployed along with SOA Service Infrastructure.

Oracle Web Services Manager is made up of the following components:

- **Policy Manager** reads and writes security and management policies including predefined and custom policies from the MDS repository. It is typically deployed on Oracle SOA Service Infrastructure managed servers. However, you can deploy Policy Manager on separate managed servers.

  Policy Manager is a stateless Java EE application. It exposes its capabilities through stateless session beans. Policy Manager does not cache any data, (all accesses go to the MDS repository).

- **Agent** is responsible for policy enforcement, execution and gathering of runtime statistics. Agent is available on all Oracle Fusion Middleware managed servers. It is configured on the same server as the application which it protects.

  Agent is made up of a set of jar files, which are a part of underlying WebService stack. It does not have any session state. Agent maintains an in-memory policy cache, which is populated at the Agent startup time. It does not use any JTA or JMS.

  Agent is made up of the following two pieces:.

  – **Policy Access Point (PAP)** interacts with Policy Manager. Agent communicates with the Policy Manager through EJB invocations over T3 (or T3s if ssl is enabled) protocol.

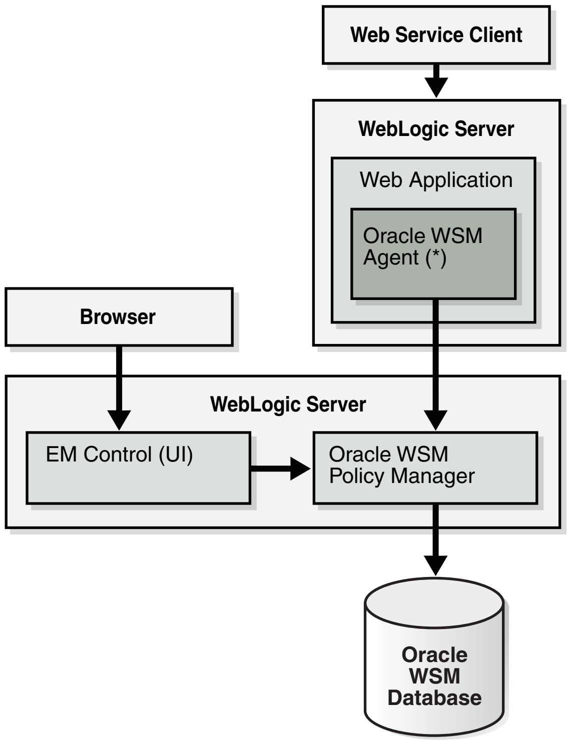
  – **Policy Interceptor** is generated when a WebService is deployed and activated, or when a policy is attached to a WebService using Enterprise Manager. If new

WebServices are protected using Oracle WSM, an additional instance of the interceptor is generated for each new WebService. Interceptor is responsible for policy enforcement.

- **Metadata Store** Policies are stored in the MDS repository. It is typically backed by an Oracle database. For high availability purposes, Oracle recommends using an Oracle RAC database as the back end for MDS repository.
- **Enterprise Manager** is used to configure Oracle WSM. It also displays different WebServices metrics gathered by Oracle WSM.

For more details of Oracle WSM architecture, please refer to the *Oracle Fusion Middleware Administration Guide Oracle WSM*.

Figure 5–16 Oracle WSM Single-Instance Architecture



5.7.1.1 Oracle WSM Component Characteristics

Oracle WSM Agent is a set of.jar files available on every Oracle Fusion Middleware managed server in a WebServices stack.

Policy Manager is contained in the `wsm-pm.ear` file. None of the services provided by Oracle WSM are singletons, therefore, it can run in full active-active mode. Oracle WSM services can be validated by `http://APPHOSTx:port/wsm-pm/validator`. This validator displays Oracle WSM policies.

Agent and Oracle Enterprise Manager interact with Policy Manager using the EJB interfaces. The EJBs used in Oracle WSM are stateless and can be deployed in a clustered environment. Therefore, there is no requirement to enable state replication in the cluster.

The agent and Policy Manager need not be co-located. However, for proper working the agent would expect policy manager to be deployed on at least one node of the domain. The WSM agent has capabilities to auto-discover Policy Managers deployed in the domain.

Neither Agent nor Policy Manager use JTA or JMS messaging. The MDS-based policy store also does not support JTA.

External Dependencies

Oracle WSM Policy Manager depends on the following components:

- MDS repository for storing the policies
- Oracle WSM Agent depends only on Oracle WSM Policy Manager.

Both components must be available for Oracle WSM to start and run properly.

5.7.1.2 Oracle WSM Startup and Shutdown Lifecycle

The following key Oracle WSM components are involved in the startup lifecycle for Oracle WSM

- Oracle WSM Policy Manager
- Oracle WSM Agent

Policy Manager is a stateless application which does not perform any caching. There is no special application level startup sequence performed when managed server where Policy Manager is deployed starts up. Policy Manager communicates with the MDS repository to retrieve policies. The MDS repository can be stored in an Oracle RAC database to provide MDS high availability.

When a managed server on which an Agent is configured comes up, the Agent connects to Policy Manager to get latest revision of policies. If it succeeds, the changes to the policies are downloaded and cached. Once the Agent is up and running, it periodically attempts a cache refresh at a configurable interval. The default time is every one minute.

Oracle WSM Agent communicates with the Policy Manager through EJB invocations over T3 (or T3s if SSL is enabled) protocol. If Policy Manager is deployed on different nodes and some of them have SSL enabled and others don't, Agent communicates only with the nodes with SSL connections.

If the Policy Manager to which Agent is connected becomes unavailable, the underlying infrastructure automatically connects to another Policy Manager instance running elsewhere in the cluster. This is achieved through Oracle WebLogic EJB clustering.

If a managed server has Web services deployed, which are protected by Oracle WSM, and Agent is not able to communicate with any of the Policy Managers at startup time, Web service invocation fails.

5.7.1.3 Oracle WSM Request Flow

When a protected WebService is accessed by a client application, Agent queries the policy cache and enforces the applicable policies. Based on the policies, the request is authenticated, encrypted, decrypted, authorized or logged. It does not connect to Policy Manager for any of these operations. Runtime availability of Policy Manager does not affect the functioning of Agent, unless there is a configuration change, such as new Webservices, which are protected by Oracle WSM, being deployed, or new policies attached to existing Webservices. If there is such a configuration change, then Agent must connect to Policy Manager to get the applicable policies. If it cannot connect after initial startup, it continues to operate based on the cached policies.

5.7.1.4 Oracle WSM Configuration Artifacts

Oracle WSM Agent configuration resides in `policy-accessor-config.xml`. This file is located in the `DOMAIN_HOME/config/fmwconfig` directory. With this configuration file you can specify:

- Policy Manager URL (if configured)
- Cache Refresh Interval
- Clock skew, to allow for difference in system clock of the client and servers

These options are also available from Oracle Enterprise Manager Fusion Middleware Control and are specific to each Oracle WSM Agent installation.

Policy Manager also depends on `adf-config.xml` for MDS repository location.

Other configuration options at the container level, such as data sources for MDS repository location, and application targeting, are maintained as part of Oracle WebLogic Server Domain configuration, and are synchronized across a cluster of Oracle WebLogic Servers by Oracle WebLogic Server core infrastructure.

5.7.2 Oracle WSM High Availability Architecture and Failover Considerations

Figure 5–6 describes a Oracle SOA Service Infrastructure two-node cluster running on two WebLogic Servers. Oracle WSM is deployed as part of Oracle SOA Service Infrastructure composite application. The following high availability characteristics are specific to an Oracle WSM high availability deployment:

- Oracle WSM Policy Manager, as well as Oracle WSM Agents are deployed on WLS_SOA_INFRA. In addition, Oracle WSM Agents are available on any custom WLS cluster, in case there is a need to protect any custom WebServices deployed on them.
- Oracle WSM Policy Manager and Agents runs in a WebLogic Server Infrastructure managed servers, that are part of a WebLogic Server cluster. The WebLogic Server cluster synchronizes configuration for common artifacts of WebLogic Server used by Oracle SOA Service Infrastructure (JDBC).
- Oracle WSM Policy Manager EJBs leverage clustering and high availability capabilities of the WebLogic Server cluster.
- All Oracle WSM Policy Manager instances in the cluster point to the same MDS repository.
- The MDS repository where Oracle WSM policies are stored is configured in an Oracle RAC database to protect from database failure.

5.7.2.1 Oracle WSM Protection from Failures and Expected Behavior

Since OWSM Agent is deployed on the same managed server as the application is deployed, hence it will be available again, as soon as the application becomes available due to server restart/migration. Following two sections describe the failover for Policy Manager.

Process Failure

Oracle WSM components are protected from process failures by the WebLogic Server infrastructure:

- If the managed servers crash, Node Manager attempts to restart them locally. If Whole Server Migration is configured and repeated restarts fail, the WebLogic Server infrastructure attempts perform server migration of the managed server to the other node in the cluster, if it is configured. Once the server on the other node is restarted, Oracle HTTP Server resumes routing any incoming requests to it. At startup time, Oracle WSM Agent and Policy Manager go through the startup lifecycle as described in Section 5.7.1.2, "Oracle WSM Startup and Shutdown Lifecycle".

- If the managed server running Policy Manager is restarted or migrated, the failover is transparent to the Agents connected to it. Policy Manager leverages underlying EJB clustering and the failover infrastructure of Oracle WebLogic Server.

Node Failure

If node failure occurs, and Whole Server Migration is not configured, Oracle WSM Agent fails over transparently to the other Policy Manager instance.

If Whole Server Migration is configured, server migration is triggered after the available server verifies the time stamp in the database leasing system. After the time stamp for leasing is verified, the Node Manager in the node that still remains available attempts to migrate the VIP used by the failed managed server, and restarts it locally. This effectively results in Oracle WSM Policy Manager application having two instances running in the same node. Refer to the Section 3.9, "Whole Server Migration" for more details on the failover process. The agents in this situation load balance against both Policy Manager instances running on the same Node.

5.7.2.2 Oracle WSM Cluster-Wide Configuration Changes

The standard Java EE artifacts that Policy Manager uses are configured as part of Oracle WebLogic Domain in which Oracle SOA Service Infrastructure is installed. Oracle WebLogic Clusters provide automatic configuration synchronization for artifacts, such as data-sources, across the WebLogic Server domain. At the same time, the WebLogic Server cluster control synchronizing the deployments and libraries used by different Oracle WSM components.

As explained in the single-instance section, Oracle WSM instance specific aspects are configured individually per WebLogic Server domain directory (typically one per node) through the `policy-accessor-config.xml` file, as well as the `adf-config.xml` file. These files are included in the .jar file that is created when using Oracle WebLogic Server Pack utility, therefore it is propagated to other nodes when you run pack/unpack to set up high availability for oracle SOA Service Infrastructure. However, ongoing changes to `policy-accessor-config.xml` or `adf-config.xml` are not replicated automatically to the other nodes. This implies that for updates, you must modify each node individually, for example, if you updated the Policy Manager URL or Cache Refresh Interval. These modifications must be manually performed in all the domain directories across a high availability topology. You can edit the `policy-accessor -config.xml` or `adf-config.xml` files directly, or use Oracle Enterprise Manager Fusion Middleware Control to make the configuration change on each managed server.

Oracle WSM Agents are capable of automatically discovering the deployed Oracle WSM Policy Manager deployments. If you want to over-ride the behavior and some point to a specific Policy Manager instance, you can do so by editing the `policy-accessor-config.xml` file.

5.8 Oracle User Messaging Service and High Availability Concepts

The information in this section guides you through the issues and considerations necessary for configuring Oracle User Messaging Service for high availability.

5.8.1 Oracle User Messaging Service Single-Instance Characteristics

Oracle User Messaging Service (Oracle UMS) enables two way communication between users and deployed applications. It has support for a variety of channels, such as email, IM, SMS, and text-to-voice messages. Oracle UMS is integrated with Oracle Fusion Middleware components, such as Oracle BPEL PM, Oracle Human Workflow, Oracle BAM and Oracle WebCenter. It is typically deployed along with Oracle SOA Service Infrastructure.

Oracle UMS is made up of the following components:

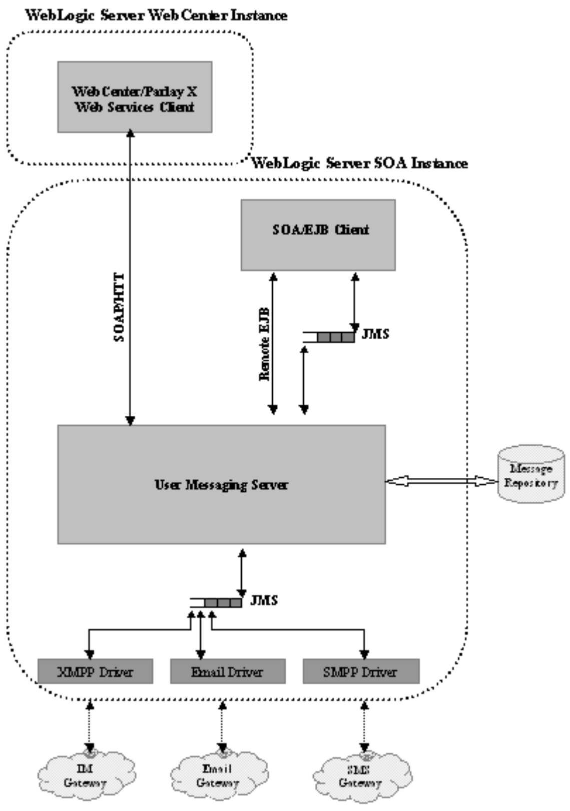
- **UMS Server** is a Java EE application that runs on Oracle WebLogic Server. The UMS Server orchestrates message flows between applications and users. The server routes outbound messages from a client application to the appropriate driver, and routes inbound messages to the correct client application. The server also maintains a repository of previously sent messages in a persistent store, and correlates delivery status information with previously sent messages.
- **UMS Drivers** connect UMS to the messaging gateways, adapting content to the various protocols supported by UMS. Drivers can be deployed or undeployed independently of one another depending on the messaging channels available in a given installation.

  UMS Drivers adapt sent and received content to and from external protocols, such as email, XMPP, and SMPP. UMS Drivers are also Java EE applications deployed to an Oracle WebLogic Server.
- **UMS Client applications** implement the business logic of sending and receiving messages. Examples of a UMS client application include an Oracle SOA application that sends messages as one step of an Oracle BPEL PM workflow, or an Oracle WebCenter Spaces application that can send messages from a Web interface.

  UMS client applications have either a UMS-specific EJB module embedded in them, or interact as standard Web service clients.

Figure 5–17 illustrates the services and dependencies of an Oracle UMS single-instance architecture.

Figure 5–17 Oracle User Messaging Service Single-Instance architecture



5.8.1.1 Oracle User Messaging Service Component Characteristics

This section describes the characteristics of Oracle UMS components.

UMS Server is made up of the following:

- Message Driven Beans (MDBs) that dequeue messages from JMS queues
- Stateless Session Beans to implement messaging business logic
- JAX-WS servlets to implement the messaging Webservices
- A simple Oracle ADF Faces User Interface component for managing end user messaging preferences.

UMS Drivers typically contain the following:

- JCA resource adapter (embedded within the EAR) in order to interface with external protocol gateways
- Two MDBs, one for inbound, and one for outbound, that interface between the resource adapter and the JMS queues
- Some UMS Drivers that implement an HTTP-based protocol, such as VoiceXML, have a servlet module as well.

UMS Client Applications that use the EJB interface have a stateless session bean that provides the API used by client business logic, and an MDB to asynchronously receive inbound messages.

UMS depends heavily on JMS. JMS queues are also used to buffer content between clients and servers, and between servers and drivers.

Each typical messaging operation (sending an outbound message or receiving an inbound message), involves two JMS queues, one between the client and server, and one between the server and driver.

UMS messaging state is stored in the database and in persistent JMS queues.

External Dependencies

UMS depends on an external database repository to maintain message and configuration state. It shares this state between clustered instances. The UMS Server accesses the database using a Java EE JDBC data source provisioned at installation time. This data source is a non-XA data source. Therefore, UMS does not depend on the JTA capabilities of the underlying infrastructure.

UMS uses JMS to deliver messages among messaging applications. By default it is configured to use a file-based persistent JMS store, therefore it depends on the storage device where those files are located.

5.8.1.2 Oracle User Messaging Service Startup and Shutdown Lifecycle

As Java EE applications, all UMS components are started by Oracle WebLogic Server container.

UMS Server startup

At server startup time, UMS Server initializes a TopLink session that creates a connection to the database repository. UMS Server then begins listening on Web service endpoints. EJBs also become available for invocation of functionality such as sending messages or retrieving delivery status.

UMS Driver startup

When a UMS driver is deployed to a cluster, it sends a registration message to the local UMS server. The UMS server records the registration information in the database, making it available to all UMS servers in the cluster. Once this occurs, any UMS server can route messages to any UMS driver in the cluster. This happens when new drivers are deployed, or when existing drivers are restarted following a configuration change.

At the time of server startup, UMS drivers typically establish a connection to the external gateway for which they are configured. Some of these are persistent socket-level connections, such as SMPP and XMPP. Some connections are established and torn down for each request, such as SMTP and IMAP connections. Drivers then make a remote EJB call to the UMS server to register themselves.

UMS Client Application startup

When the managed server where the clients are deployed starts up, EJB clients typically make a remote EJB call to register themselves with the UMS server. Web service clients do not have explicit registration mechanisms.

5.8.1.3 Oracle User Messaging Service Request Flow

UMS clients make a series of short-lived requests to the UMS server. Web service clients make requests using SOAP or HTTP. EJB clients make remote EJB calls to the server for initialization and configuration management, in addition to interacting using JMS.

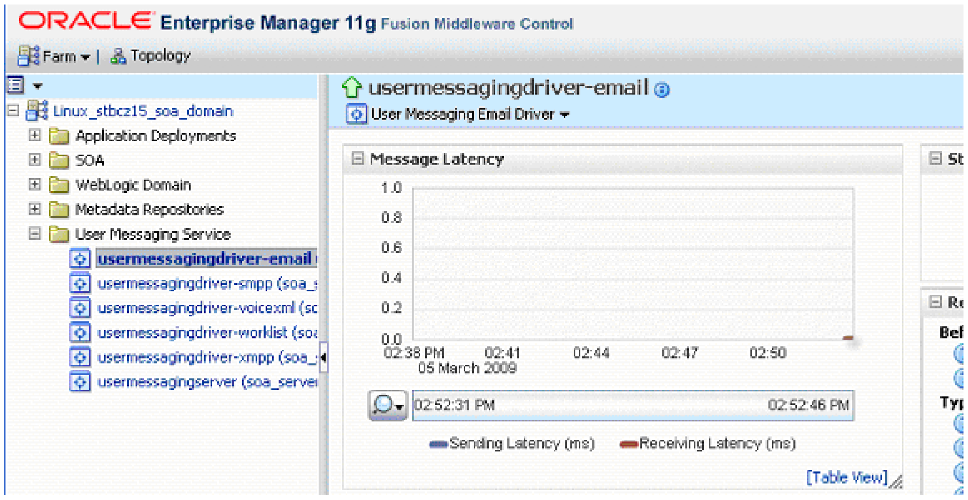
After initial startup and registration, subsequent UMS request flows can be categorized as follows:

- **Outbound messaging**: For an outbound message, a client application wishes to send a message to a user. The client application either makes a Web service request to the UMS Server, or enqueues a message in the UMS Server's sending JMS queue. The UMS Server receives the request, records the messaging state, and selects an appropriate driver. The UMS Server enqueues the message in a different outbound JMS queue. The corresponding driver, which is listening to the outbound queue, dequeues the message, converts it to the appropriate format for the given protocol, and delivers it to an external gateway (such as an SMTP server or IM gateway.)
- **Inbound messaging**: For an inbound message, the end user wishes to send a message back to the client application. The user sends a message from a device, such as an IM client or phone. The message passes from the external gateway to the driver. The driver adapts the message to the UMS format, and enqueues it in a JMS queue. The server dequeues the message from the JMS queue, records the message state, and enqueues the message in the appropriate application's inbound JMS queue. The client application then dequeues the message and processes it as needed.

5.8.1.4 Oracle User Messaging Service Configuration Artifacts

UMS Server and Drivers each have an XML configuration file that is the artifact of a configuration Mbean, implemented using Oracle Fusion Middleware JMX Framework. You can perform configuration changes using Oracle Enterprise Manager Fusion Middleware Control.

Figure 5–18 Configuring Oracle User Messaging Service using Oracle Enterprise Manager Fusion Middleware Control


ORACLE Enterprise Manager 11g Fusion Middleware Control
Farm
Topology
Linux_stbcz15_soa_domain
Application Deployments
SOA
WebLogic Domain
Metadata Repositories
User Messaging Service
usermessagingdriver-email
usermessagingdriver-smpp (soa_s
usermessagingdriver-voicexml (sc
usermessagingdriver-worklist (soa
usermessagingdriver-xmpp (soa_s
usermessagingserver (soa_server
usermessagingdriver-email
User Messaging Email Driver
Message Latency
1.0
0.8
0.6
0.4
0.2
0.0
02:38 PM
05 March 2009
02:41
02:44
02:47
02:50
02:52:31 PM
02:52:46 PM
Sending Latency (ms)
Receiving Latency (ms)
[Table View]

All UMS components use standard Java EE deployment descriptors and Oracle WebLogic proprietary descriptors for configuring Java EE components. These descriptors can be configured using standard Oracle WebLogic tools, such as Oracle WebLogic Server Administration Console or WLST. For more information, see UMS administration topics in *Oracle Fusion Middleware Administrator's Guide for Oracle SOA Suite*.

If you are using Oracle SOA Suite in a clustered environment, any configuration property changes you make in Oracle Enterprise Manager on one node must be made on all nodes. Configuration properties are set in Oracle Enterprise Manager through the following options of the SOA Infrastructure menu:

Administration > **System MBean Browser**

SOA Administration > any property selections.

5.8.2 Oracle User Messaging Service High Availability Architecture and Failover Considerations

See section Section 5.2.2, "Oracle SOA Service Infrastructure High Availability Architecture and Failover Considerations" for a description of Oracle SOA Service Infrastructure two-node high availability characteristics. Figure 5–6 illustrates an Oracle SOA Service Infrastructure cluster running on two WebLogic Servers. Oracle User Messaging Service is deployed as part of Oracle SOA Service Infrastructure composite application. The following high availability characteristics are specific to an Oracle User Messaging Service high availability deployment:

- All UMS components are deployed to Oracle WebLogic Service Infrastructure managed servers that are part of an Oracle WebLogic Server cluster. Oracle WebLogic Server cluster synchronizes configuration for common artifacts of Oracle WebLogic Server used by UMS, such as JDBC data sources.
- All Oracle UMS components are stateless.
- UMS Server and Client stateless EJBs leverage clustering and high availability capabilities of Oracle WebLogic Server cluster
- UMS Server relies on a shared database repository for persistent storage.
- UMS relies on JMS distributed destinations for load-balancing and availability across cluster nodes. It also relies on the JMS connection factory's capability to failover to a different JMS server in the event of a failure.
- The user messaging preferences user interface does not require session stickiness. It remains available through the use of a basic load balancing. There are no sticky session routing requirements, as all session state is persisted in the database and shared across the clustered instances.
- UMS does not participate in any global transactions.
- UMS uses Oracle WebLogic Server multi data source to connect to the back-end Oracle RAC database.

5.8.2.1 Oracle User Messaging Service Protection from Failures and Expected Behavior

Oracle UMS is typically deployed on the same managed server as Oracle SOA Service Infrastructure. Oracle SOA Service Infrastructure is protected from process and node failures using Oracle WebLogic Server Whole Server Migration. Whole Server Migration also provides failover capabilities for JMS usage.

For information about Oracle UMS protection from failures and expected behavior, see Section 5.2.2.1, "Oracle SOA Service Infrastructure Protection from Failures and Expected Behavior".

Process Failure

This section describes specific considerations for process failure in an Oracle User Messaging Service high availability configuration:

- Node Manager attempts to restart the managed servers locally if a crash occurs.
- If Whole Server Migration is configured for Oracle WebLogic Server managed server to which Oracle UMS components are deployed, and the restart count

threshold is exceeded, Oracle WebLogic Server infrastructure attempts to perform a server migration of the managed server to another node in the cluster. After the server migration completes successfully, at the startup time, UMS Server and Drivers go through the startup cycle as previously described, including driver registration. Driver registration is an independent operation and does not have any affect on other available instances. See Chapter 3, "High Availability for WebLogic Server" for more information about server migration.

- At restart, (on the same or different node), the UMS JMS server in the managed server starts producing and consuming messages from its JMS store.
- If a managed server running UMS Server and Drivers is restarted or migrated, the failover is transparent to the connected UMS Clients. The failover is transparent because UMS components are stateless. Once the server's restart is finished, the Web server starts routing requests to it for Web service clients. Similarly, EJB clients become aware of the server availability and start routing requests to it. This is made possible by Oracle WebLogic clustering infrastructure.

Node Failure

For information about Oracle UMS node failure, see Section 5.2.2.1, "Oracle SOA Service Infrastructure Protection from Failures and Expected Behavior"

Database Failure

For information about Oracle UMS database failure, see Section 5.2.2.1, "Oracle SOA Service Infrastructure Protection from Failures and Expected Behavior"

Protection From External Messaging Gateway Failures

Before attempting a message delivery, UMS first persists the message to the database. If an external Messaging Gateway becomes unavailable, the corresponding UMS driver periodically attempts to reconnect to the gateway and deliver any undelivered messages persisted in the database. Alternatively, if the messages are not delivered, administrators can manually resend the messages using the UMS server's Message Status Page in Oracle Fusion Middleware Enterprise Manager.

5.8.2.2 Oracle User Messaging Service Cluster-Wide Configuration Changes

UMS configuration is file-based, standard Java EE deployment descriptors and JMX configuration Mbeans, using a standard JMX framework. Changes are propagated using standard Oracle WebLogic Server Mbean server mechanisms. There are no cluster-wide configuration capabilities. As a result, configuration changes must be repeated on every member of a cluster

UMS uses standard Java EE artifacts, configured as part of Oracle WebLogic's domain in which Oracle UMS is installed. Oracle WebLogic Server clusters provide automatic configuration synchronization for artifacts, such as data sources, across Oracle WebLogic Server domain. At the same time, Oracle WebLogic Server cluster controls synchronization of deployments and libraries used by different Oracle UMS components.

5.9 Oracle JCA Adapters and High Availability Concepts

The information in this section guides you through the issues and considerations necessary for configuring Oracle JCA Adapters for high availability.

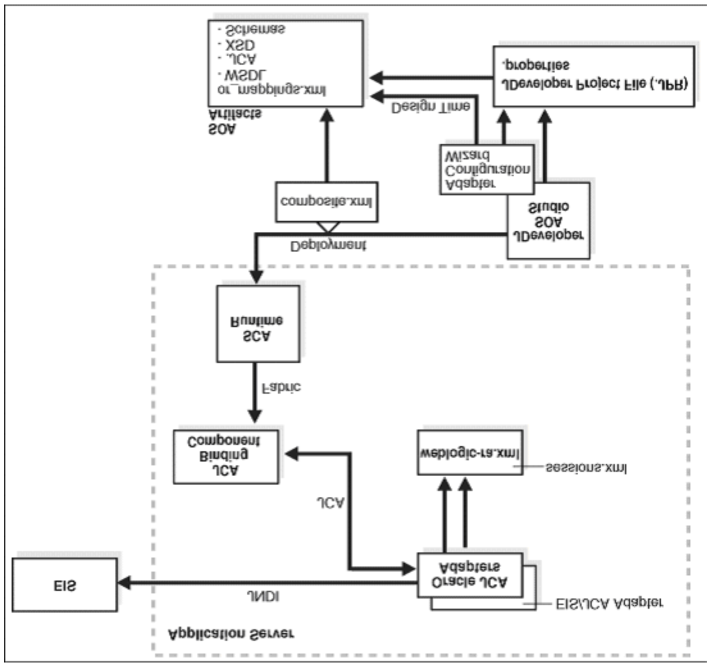
5.9.1 Oracle JCA Adapters Single-Instance Characteristics

Oracle JCA Adapters are JCA binding components that allow the Service Infrastructure to communicate to endpoints using different protocols. Oracle JCA Adapters are deployed as a JCA resource (RAR) and are not part of Oracle SOA Service Infrastructure.

The run-time component of Oracle JCA Adapters is the J2CA 1.5 resource adapter for the specific back-end application. Oracle JCA Adapters are deployed in J2CA container of Oracle WebLogic Server. Oracle Fusion Middleware integrates with these J2CA 1.5 adapters through the JCA Binding Component, which converts Web service messages into J2CA interactions and back. Adapters can, therefore, fully leverage the scalability and high availability of the underlying Oracle WebLogic Server platform.

Figure 5–19 illustrates the services and dependencies of Oracle JCA Adapters single-instance architecture.

Figure 5–19 Oracle JCA Adapters Single-Instance Architecture



5.9.1.1 Oracle JCA Adapters Component Lifecycle

The lifecycle of an adapter is controlled by Oracle SOA Service Infrastructure. There are three high-level steps in the initialization of an Oracle JCA Adapter:

- **Deployment**: Oracle JCA Adapters are deployed during installation as J2CA 1.5 resource adapters (RAR files) within the same Oracle WebLogic Server container as Oracle Fusion Middleware. The physical deployment of adapters involves using the RAR file and registering the adapters as connectors with the underlying Oracle WebLogic Server, or the middle tier platform. The RAR file contains the `ra.xml` file, which contains declarative information about the contract between Oracle WebLogic Server and the resource adapter. In addition, the RAR file contains `weblogic-ra.xml` which is the deployment descriptor file containing deployment-specific information about the resource adapter.
- **Loading**: Loading refers to the process of creating in-memory resources and creating connections to the configured end-point. Although Oracle JCA Adapters

are physically deployed as J2CA 1.5 resource adapters, their logical deployment involves creating the Connection Factory entries for the J2CA 1.5 resource adapter by editing the `weblogic-ra.xml` file. This file is Oracle WebLogic Server-specific deployment descriptor for a resource adapter. It contains configurations for deploying resource adapters to Oracle WebLogic Server, which includes the back-end application connection information as specified in the deployment descriptor of the resource adapter, Java Naming and Directory Interface (JNDI) name to be used, connection pooling parameters, resource principal mapping mechanism, and configurations.

- **Activation**: Activation refers to initiation of a JCA binding component (Service and Reference) within a Composite. Listeners are started for the endpoint referenced by the Adapter configuration within the Composite.

Property Changes During Oracle JCA Adapters Runtime

Oracle JCA Adapters expose the underlying back-end operation-specific properties as header properties and allow the manipulation of these elements within a business process.The underlying properties are as follows:

- interactionspec or activationspec Properties - These properties require the adapter endpoint to be recycled (re-activated).
- Endpoint Properties - Changes to these properties are notified to the adapter without requiring the endpoint to be restarted.

5.9.1.2 Oracle JCA Adapters Reliability and Transactional Behavior

Oracle JCA Adapters support global transactions based on the JCA 1.5 XA contracts that leverage the underlying application server transaction manager. Adapters supporting XA transactions include Oracle Adapters for Oracle Applications, database, Advanced Queuing, JMS and MQSeries. Non-transactional adapters include Oracle File Adapter and Oracle FTP Adapter.

Inbound Transactions

For a synchronous process, the global transaction initiated by the adapter spans message delivery and composite execution.

For an asynchronous service entry point, a transactional adapter initiates a global JTA transaction before sending an inbound message to a composite. When control returns to the adapter, it commits the JTA transaction, executing the following set of actions as an atomic unit of work:

1. Commit the removal of the message from the inbound adapter endpoint, for example, table and queue.
2. Commit the execution of the composite instance.

If anything fails during this process, both of these actions are rolled back based on XA guarantees.

Outbound Transactions

For transactional adapters, outbound JCA interactions (the invoke activities) are scoped with the global JTA transaction of the Composite instance. This means that all composite activities, including Oracle JCA adapter invocations, are part of a global transaction, and as such all activities are either committed or rolled back if an error occurs. Therefore, one can guarantee exactly-once message delivery when both inbound and outbound adapters are transactional and the connection factories have been configured to support XA global transactions.

Nontransactional

The Oracle File Adapter picks up a file from an inbound directory, processes the file, and sends the processed file to an output directory. However, during this process if a failover occurs in the Oracle RAC backend or in an SOA managed server, then the file is processed twice because of the nontransactional nature of Oracle File Adapter. As a result, there can be duplicate files in the output directory.

5.9.1.3 Oracle JCA Adapters - Rejected Message Handling

The messages that error out before being posted to Oracle SOA Service Infrastructure mesh are referred to as rejected messages. For example, Oracle File Adapter picks a file having data in CSV format and tries to translate it to the XML format using NXSD. Now, if there is any errors in the translation, the message is rejected and is not posted to the target composite.

All rejected messages are stored in the database with payload. There are two destinations for rejected messages:

- Adapter Rejection Table - Messages are stored in the Adapter Rejection table when there is a binding error related to a corrupt message.

  In case the database hosting the Rejected Message Table is offline, rejected messages are temporarily stored in the local file system (as a backup) and eventually loaded back into the database when it is detected as being online again.
- Composite Instance Failure Table - This table is part of instance tracking, populated by each component detecting a failure.

In the case of errors being thrown by the SOA Service Infrastructure layer, the Adapter framework behavior depends on whether or not the error is marked as retriable. If the error is not retriable, the message is treated as a rejected message and persisted to the Rejected Message Table.

Configuring Rejection Handlers

Rejection handlers are defined and configured by using fault policies as described in the following steps:

1. Define a fault policy for the rejected messages in the `fault-policies.xml` file, which is stored along with `composite.xml` in the Oracle JDeveloper project directory.
2. Associate the fault policy with a service endpoint of the composite in the `fault-bindings.xml` file.
3. Copy the `fault-policies.xml` file and the `fault-bindings.xml` file to Oracle SOA Composite project directory.
4. Deploy the Oracle SOA Composite project.

Retry frequency and retry interval are both configurable. Service engines can mark an invocation as retriable. In a clustered environment, if roll back occurs, it may be picked up by another instance. If another node picks up the message, it generates a unique ID. The number of retries are configured within the composite, under the `binding.jca` element. For more information, see the *Oracle Fusion Middleware User's Guide for Technology Adapters*.

System-Defined Rejected Message Handlers

The following system-defined error handlers can be configured in fault policies:

- **Web Service Handler** - Predefined WSDL interface must be implemented by the target service
- **Custom Java Handler** - A predefined Java interface must be implemented by the target class.
- **JMS Queue** - The rejected message is enqueued to a queue as a JMS message with the appropriate context and payload.
- **File** - The rejected message is stored in the file system with the proper context and payload.

Payload Persistence

For resubmitting rejected messages, payload persistence is imperative. Payloads are stored in the database and be viewed through Oracle Enterprise Manager Fusion Middleware Control. Error handlers, invoked during automatic error handling also receive the payload during their invocation. Error payload persistence in the database is enabled by default.

5.9.2 Oracle JCA Adapters High Availability Architecture and Failover Considerations

This section describes an Oracle JCA Adapters high availability considerations for configuring Oracle JCA Adapters to run on an Oracle SOA cluster.

5.9.2.1 Oracle JCA Adapters High Availability Error Handling

This section describes Oracle JCA High Availability error handling.

Connection Errors

Oracle JCA Adapters and Oracle Adapters for Oracle Applications handle connection errors for the following interactions:

- **Outbound Interaction**: Oracle JCA Adapters and Oracle Adapter for Oracle Applications raise the `oracle.tip.adapter.api.exception.PCRetriableResourceException` exception for transient connection errors, which are recoverable connection errors. For example, a database listener may not have started, resulting in connection errors. You can define the maximum number of attempts to reconnect made using the `fault-policy.xml` file. The parameters for attempts to reconnect can be specified using this file. After the configured number of retries is reached, the `fabricInvocationException` exception is thrown.

  Fault handling and retry properties are defined as part of the binding level configuration. When the binding level retry has expired, the fault is handled based on policies specified within the fault-policy.xml. For more information on binding properties, see the *Oracle Fusion Middleware User's Guide for Technology Adapters*.
- **Inbound Interaction**: Oracle JCA Adapters, Oracle Adapter for Oracle Applications, and legacy adapters support a poll model for connecting to the back-end application for receiving events. In case of unrecoverable connection failures, the adapters recycle old connections, and send out alerts or notifications to Oracle SOA Service Infrastructure. The inbound interaction connection errors are written to a log, and can be viewed through Oracle WebLogic Server Administration Console.

5.9.2.2 Oracle File and FTP Adapters High Availability

The Oracle File and FTP Adapters can be configured for high availability using an active-active topology. A cluster-based high availability configuration is supported for both inbound and outbound operations. File and FTP Adapters do not support XA and hence can only guarantee at-least-once delivery of messages. Thus, it is possible to have duplicate messages after recovery from a server crash.

Prerequisites for High Availability

The following list describes prerequisites for Oracle JCA Adapters high availability:

- In a clustered configuration, inbound adapters across managed servers must point to the same physical directory, a directory on a shared drive.
- Connection-factories must specify the same shared folder as the control directory, and their names must match. For example, if the deployment descriptor for one connection-factory has /shared/control_dir as the value for controlDir, the other deployment descriptor must also have the same value.

The Oracle File and FTP Adapters must ensure that only one node processes a particular file in a distributed topology. You can use the database table as a coordinator to ensure that Oracle File and FTP Adapters are highly available for inbound operations. See the *Oracle Fusion Middleware User's Guide for Technology Adapters* for details on configuring a database table as a coordinator.

The Oracle File and FTP Adapters must ensure that if multiple references write to the same directory, then these do not overwrite each other. The following locking capabilities can be used to make Oracle File and FTP Adapters highly available for outbound operations:

- Database mutex
- User-defined mutex

See the *Oracle Fusion Middleware User's Guide for Technology Adapters* for details on configuring a database mutex.

Oracle JCA Adapters High Availability Configuration

A cluster-based high availability configuration is supported for both inbound and outbound operations. However, consider the following:

- Inbound Operations - Oracle File and FTP Adapters must ensure that only one node processes a particular file in a distributed topology.
- Outbound Operations - Oracle File and FTP Adapters must ensure that if multiple references write to the same directory, these do not overwrite each other.

Database-based mutexes are used as coordinators to ensure these behaviors in a clustered topology.

Configuring a Database Mutex

Configure a database table as a coordinator by Modifying Oracle File Adapter deployment descriptor for the `connection-instance` corresponding to `eis/HAFileAdapter` from Oracle WebLogic Server Administration Console:

1. Click **FileAdapter** under **Summary of Deployments** on Oracle WebLogic Server Administration Console.
2. Click the **Outbound Connection Pools** tab, and expand **javax.resource.cci.ConnectionFactory** to see the configured connection factories.

3. Click **eis/HAFileAdapter**. The **Outbound Connection Properties** for the connection factory corresponding to high availability is displayed.
4. Update the connection factory properties.

Update the Adapter configuration to use the connection factory as shown in the following example:

```
<adapter-config name="FlatStructureOut" adapter="File Adapter"
xmlns="http://platform.integration.oracle/blocks/adapter/fw/metadata">
              <connection-factory location="eis/HAFileAdapter" adapterRef=""/>
              <endpoint-interaction portType="Write_ptt" operation="Write">
            <interaction-spec
className="oracle.tip.adapter.file.outbound.FileInteractionSpec">
                  <property../>
                  <property../>
                </interaction-spec>
              </endpoint-interaction>
            </adapter-config>
```

Note: The location attribute is set to `eis/HAFileAdapter` for the connection factory.

The new parameters in connection factory for Oracle File and FTP Adapters are as follows:

- **controlDir** - Set this parameter to the directory structure where you want the control files to be stored. Set it to a shared location if multiple Oracle WebLogic Server instances are running in a cluster.
- **inboundDataSource** - Set this parameter to `jdbc/SOADataSource`. This is the data source where the schemas corresponding to high availability are pre-created. The pre-created schemas are located under `MW_HOME/AS11gR1SOA/rcu/integration/soainfra/sql/adapter/createschema_adapter_oracle.sql`. To create the schemas elsewhere, use this script. Set the `inboundDataSource` property accordingly if you choose a different schema.
- **outboundDataSource** - Set this parameter to `jdbc/SOADataSource`. This is the data source where the schemas corresponding to high availability are pre-created. The pre-created schemas can be found under `MW_HOME/AS11gR1SOA/rcu/integration/soainfra/sql/adapter/createschema_adapter_oracle.sql`. To create the schemas elsewhere, use this script. Set the `outboundDataSource` property if you do create the schemas elsewhere.
- **outboundLockTypeForWrite** - Set this parameter to `oracle` if you are using Oracle database. By default Oracle File and FTP Adapters use an in-memory mutex to lock outbound write operations. Choose one of the following values for synchronizing write operations:
  - **memory** - Oracle File and FTP Adapters use an in-memory mutex to synchronize access to the file system.
  - **oracle** - The adapter uses Oracle database sequence.
  - **db** - The adapter uses a pre-created database table (FILEADAPTER_MUTEX) as the locking mechanism. Use this option only if you are using a schema other than the Oracle database schema.

– **user-defined** - The adapter uses a user-defined mutex. In order to configure the user-defined mutex, implement the mutex interface: `oracle.tip.adapter.file.Mutex` and then configure a new binding-property with the name `oracle.tip.adapter.file.mutex` and value as the fully qualified class name for the mutex for the outbound reference.

Note: For large payloads, increase transaction timeout for the SOA DataSource by adding the following:

```
<xa-set-transaction-timeout>true</xa-set-transaction
-timeout>
```

```
<xa-transaction-timeout>1000</xa-transaction-timeout
>
```

Increase global transaction timeouts if a database is used as the coordinator.

5.9.2.3 Oracle Database Adapters High Availability

The Oracle Database Adapter supports high availability in an active-active setup. In an active-active setup, distributed polling techniques can be used for inbound Database Adapters to ensure that the same data is not retrieved more than once. For more information, see Section 9.3.7.1, "Distributed Polling Best Practice: MarkReservedValue." of the Adapter Guide. Similar to other adapters, an Oracle Database Adapter can also be configured for singleton behavior within an active-passive setup. This allows a high performance multi-threaded inbound Oracle Database Adapter instance running in an active-passive setup, to follow a fan out pattern and invoke multiple composite instances across a cluster. The Oracle Database Adapter also supports the high availability feature when there is a database failure or restart. The DB adapter picks up again without any message loss.

For information about how an inbound rejected message is handled by using fault policy, see the *Oracle Fusion Middleware User's Guide for Technology Adapters*.

Surviving Database Restart

The Oracle Database Adapter can survive a database down scenario and pick up again without any message loss. For avoiding data loss, the data-source needs to be XA enabled and configured for RAC (multi-data source). For information on configuring a datasource for high availability, see Appendix B, "Recommended Multi Data Sources."

5.9.2.4 Oracle JMS Adapters High Availability

Oracle JMS Adapters support multiple provider including WebLogic JMS, MQ Series and Oracle AQ. JMS Adapters support exactly-once message delivery with no message loss during a server crash. In order to guarantee exactly-once delivery of messages, the JMS connection factory should be configured to support XA. For details on configuring connection factories for different adapters please refer to Section 8.4 of the *Oracle Fusion Middleware User's Guide for Technology Adapters*.

For Oracle AQ, it is also necessary to configure the data-source for XA and RAC. For information on configuring a datasource for high availability, see Appendix B, "Recommended Multi Data Sources."

5.9.2.5 Oracle JCA Adapters Log File Locations

You can view the logs for Oracle JCA Adapters as follows:

Oracle JCA Adapters and Oracle Adapter for Oracle Applications: For both outbound and inbound interactions, the log files are redirected to the soa-diagnostic.log file.The log files for Oracle Fusion Middleware SOA Suite that is deployed to the server-soa managed server are located in:

```
MW_HOME/user_projects/domains/domain_
name/servers/server-soa/logs/soa-diagnostic.log
```

Packaged-application adapters: These adapters do not implement the LogManager interface because it is not part of the J2CA 1.5 standard. Therefore, for OPMN-managed components the log outputs are redirected to

```
ORACLE_INSTANCE\diagnostics\logs\component_type\component_name.
```

For outbound interactions, the logs are directed to the same location. On the other hand, for inbound interactions, logs are redirected to soa-diagnostic.log.

Legacy adapters: In addition to the J2CA resource adapter, legacy adapters consists of Oracle Connect, which consists of native adapters for communicating with the mainframe application and data stores. Oracle Connect logs can be viewed using Oracle Studio, which is the mainframe adapter design-time tool and Oracle Connect management tool. Oracle Connect generates various types of logs, such as the daemon log, workspace log, and server process log.

5.10 Oracle Business Activity Monitoring and High Availability Concepts

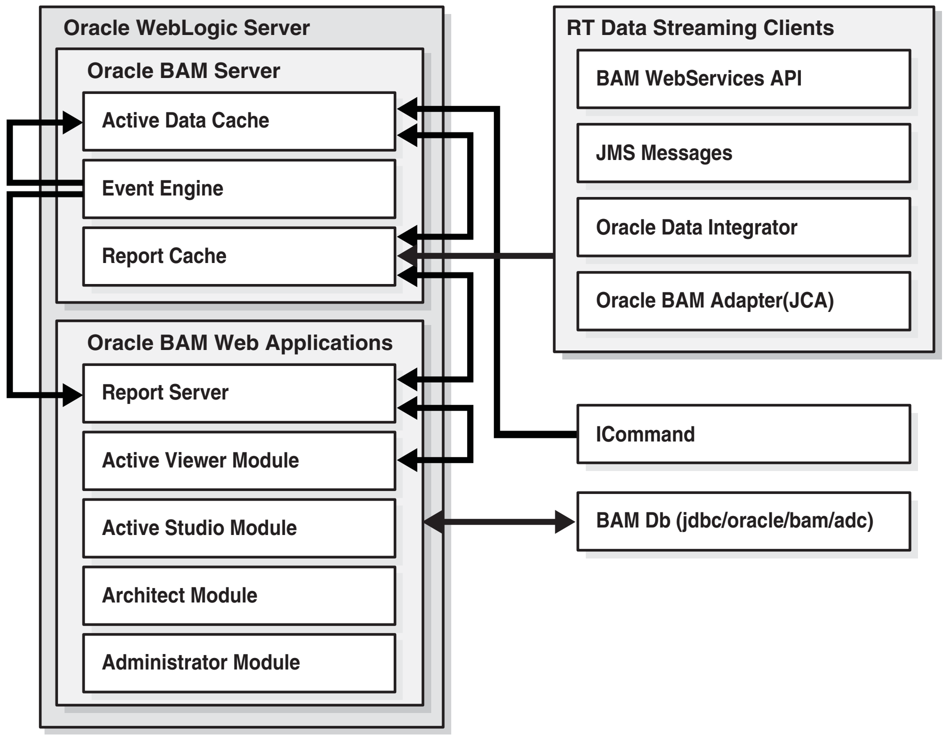
The information in this section guides you through the issues and considerations necessary for configuring Oracle BAM for high availability.

5.10.1 Oracle Business Activity Monitoring Single-Instance Characteristics

Oracle BAM provides the tools for monitoring business services and processes in the enterprise. It allows correlating of market indicators to the actual business process and to changing business processes quickly or taking corrective actions if the business environment changes. Oracle Business Activity Monitoring (Oracle BAM) provides the necessary tools and runtime services for creating dashboards that display real-time data inflow and define rules to send alerts under specified conditions.

Figure 5–20 illustrates the main services and dependencies that characterize an Oracle BAM instance.

Figure 5–20 Oracle Business Activity Monitoring Single-Instance Architecture



5.10.1.1 Oracle Business Activity Monitoring Component Characteristics

Oracle BAM is made up of the following components:

- **Oracle BAM Server** is a set of runtime components that handle incoming data from different data sources. Oracle BAM components are also used to evaluate conditions for sending alerts to users and triggering the required actions configured for these alerts. The following are the main components in an Oracle BAM Server:
  - **Active Data Cache** is designed and optimized to handle large amounts of data in a real-time solution. To make data readily accessible and deliverable, it maintains real-time views of the data. The data feed to the Active Data Cache is a combination of business data sources, from data warehouse information to transactional feeds and other enterprise sources. The various data streaming technologies integrated with Oracle BAM send this information to the Active Data Cache in a continuous stream as data changes occur.

    The Active Data Cache hosts and runs the data objects, the view sets and the active view sets. It receives transactions (insert, update, delete, and upsert) to its data objects, and these data objects notify other data objects which are linked to them through lookups. Active view sets which are monitoring these data objects are notified of the changes and produce active data.
  - **Event Engine**: Event Engine monitors complex data conditions and implements specified rules. Rules can include a series of conditions and actions attached to an event. The Event Engine continuously monitors the information in the Active Data Cache for certain conditions and executes the related actions defined in associated rules.

    The Event Engine is responsible for tracking events based on date, time or data changes. The design of the Event Engine uses a satellite concept, in which there are four different systems (satellites), with which event clauses can be registered and in which they can be tracked.

  - **Report Cache** off loads the burden of maintaining the view set snapshot in memory from the Active Data Cache. The Report Cache opens view sets and active view sets in the Active Data Cache for the Report Server in Oracle BAM Web Applications set of components. It then caches the snapshot (in chunks) and the active data before sending it to the Report Server. This allows for random access into the snapshot and recovery from losing internet connectivity. The Report Cache also allows the Report Server to be stateless, and with the Active Data Cache it supports view set sharing.

- **Real Time Data Streaming Clients**: In an Oracle BAM system clients can feed data into Oracle BAM server by using different types of protocols and APIs. The following are the available mechanisms to send data to the Oracle BAM server:

  - **Oracle BAM Adapter** is a JCA-compliant Adapter, which Java EE applications can use to send data to the Oracle BAM Server.

  - **Enterprise message sources** are used by applications to provide direct Java Message Service (JMS) connectivity to the Oracle BAM server by mapping messages directly to Oracle BAM data objects. The Oracle BAM server can read data directly from any JMS based message queue or topic. This option offers guaranteed messaging.

    In an Oracle BAM high availability environment, involving Enterprise Message Sources, the Enterprise Message Source property **Start when BAM Server starts** should be set to **Yes**.

    Durable Subscriber Name must be set in the Enterprise Message Source to ensure that no messages are lost during the failover.

  - **Oracle Data Integrator** is the ETL tool that is used with Oracle BAM to perform rigorous data transformations. The Oracle BAM Server has been implemented as an ODI Technology (for example, DB2, SQL Server are ODI Technologies) and Oracle BAM has ODI Knowledge Modules which let ODI perform all of the operations on the Oracle BAM Server to facilitate reading and writing data in various ways, including Change Data Capture.

  - **Web Services API** interacts directly with Oracle BAM data objects from a remote client.

- **Oracle BAM Web Applications** are the Web user interfaces (Java EE Web applications) for viewing Oracle BAM data. Oracle BAM Web Applications also allow for building data models and creating dashboards and alerts. Oracle BAM Web Applications are the browser based interfaces provided for building reports and administrating the users that have access to the BAM User Interface system. As part of the Web applications, a Report Server applies the report definitions to the data sets retrieved from the Active Data Cache for presentation in a browser.

- **ICommand** is a command line interface for administrating the data in the Active Data Cache. It provides a set of commands to export, import, rename, clear, and delete items from Active Data Cache.

The BAM application runs separately from the rest of Oracle Fusion Middleware SOA Suite components. Oracle BAM Web Applications are deployed together with the BAM Server. Oracle BAM Server uses EJBs and DAOs for implementing its services and persisting information to the database. All of the EJBs are stateless. However, the Oracle BAM Server is a singleton: only one active Oracle BAM server is used for maintaining fed data and for pushing it into the Web Application Reports. Oracle WebLogic Server Migration feature is used to protect Oracle BAM server from failures.

Oracle BAM server uses JMS intensively. However, all Oracle BAM JMS queues are internal and do not require to be configured in a high availability configuration. When failover occurs, the queues are recreated on the new active Oracle BAM instance.

Oracle BAM Web Application Modules can run in full active-active mode connecting to the available Oracle BAM server. The information is retrieved using RMI protocol to access the EJBs that provide the information to remote consumers. Oracle BAM Web Applications are stateless, except for the Reports Server, which maintains a session ID for reports or users accessing the systems. This requires enabling session replication for the WebLogic Managed Server where Oracle BAM Web Applications run.

The processing of work by the EJBs and threads in Oracle BAM is non transactional and does not require Oracle WebLogic Server special transaction logs configuration. At the same time, Oracle BAM uses the database intensively, therefore it is important that Oracle BAM's database access is prepared for failure in the database. This requires configuring multi data sources for Oracle BAM data source as described in Section 5.12, "Configuring High Availability for Oracle BAM". For information about multi data source configuration with RAC and the MDS repository, see Section 4.1.2, "Using Multi Data Sources with Oracle RAC."

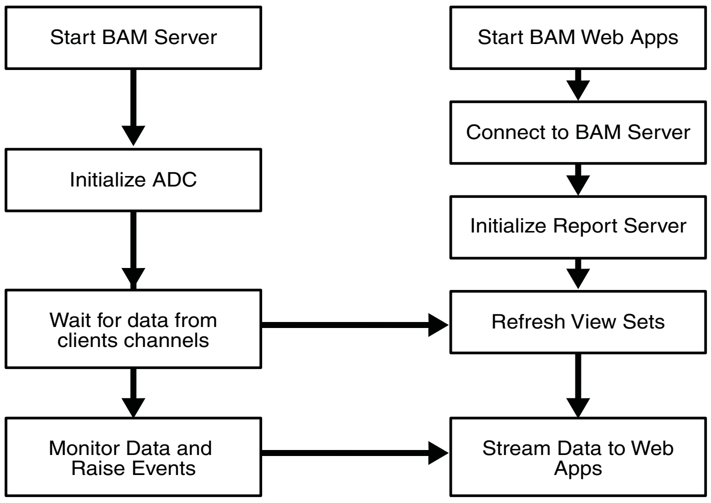
External Dependencies

Oracle BAM engine system depends only on the Oracle BAM database (which contains Oracle BAM schemas) for Oracle BAM data and report metadata. The database must be available for Oracle BAM system to start and run properly.

5.10.1.2 Oracle Business Activity Monitoring Startup/Shutdown Lifecycle

As shown in Figure 5–21, when the WebLogic Server is started, the Oracle BAM Server application is initialized. During startup, the Active Data Cache loads all the data from the repository. The Reports Cache is initialized with the data required for the available systems. The Event Engine starts analyzing the data and preparing notifications. Oracle BAM Web Applications are configured at boot time with an Oracle BAM Server's address. When Oracle BAM Web Applications are started, they connect to the Active Data Cache in Oracle BAM server and retrieve the corresponding viewsets through the Reports Server. Once initialization is done, the system is ready to receive data from clients and raise events and update reports dynamically.

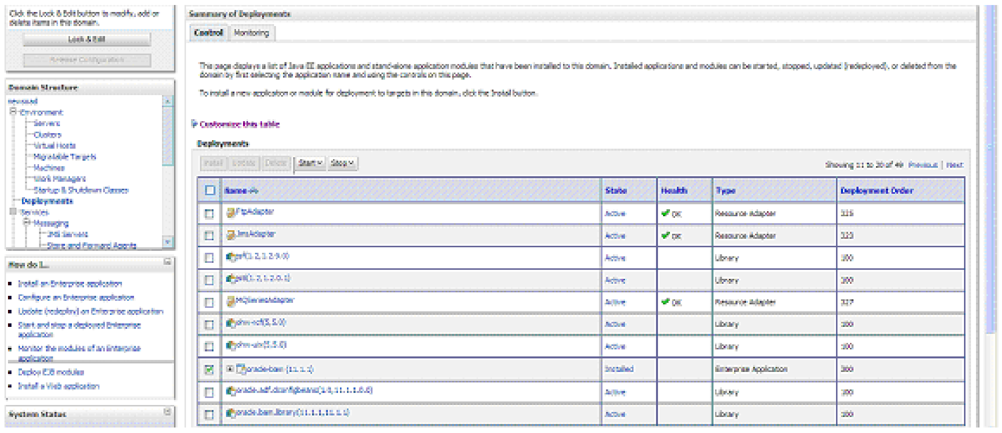
Figure 5–21 Oracle Business Activity Monitoring Startup/Shutdown Lifecycle



5.10.1.3 Oracle Business Activity Monitoring Startup and Shutdown of Processes

Oracle BAM Server is an application, `oracle-bam`, that operates independently from the rest of the SOA Service Infrastructure. It runs in the managed server where Oracle BAM is installed. It gets started by default with Oracle WebLogic Managed Server to which Oracle BAM is deployed. Use Oracle WebLogic Server Administration Console to verify Oracle BAM Server's status and to start and stop it. You can also use WebLogic Server WLST command line to control the application.

Figure 5–22 Startup and Shutdown of Oracle Business Activity Monitoring using WebLogic Server


Click the Lock & Edit button to modify, add or delete items in this domain.
Lock & Edit
Release Configuration
Domain Structure
newsoad
Environment
Servers
Clusters
Virtual Hosts
Migratable Targets
Machines
Work Managers
Startup & Shutdown Classes
Deployments
Services
Messaging
JMS Servers
Store and Forward Agents
How do I...
Install an Enterprise application
Configure an Enterprise application
Update (redeploy) an Enterprise application
Start and stop a deployed Enterprise application
Monitor the modules of an Enterprise application
Deploy EJB modules
Install a Web application
System Status
Summary of Deployments
Control
Monitoring
Customize this table
Install
Update
Delete
Start
Stop
Showing 11 to 20 of 49 Previous | Next

This page displays a list of Java EE applications and stand-alone application modules that have been installed to this domain. Installed applications and modules can be started, stopped, updated (redeployed), or deleted from the domain by first selecting the application name and using the controls on this page.

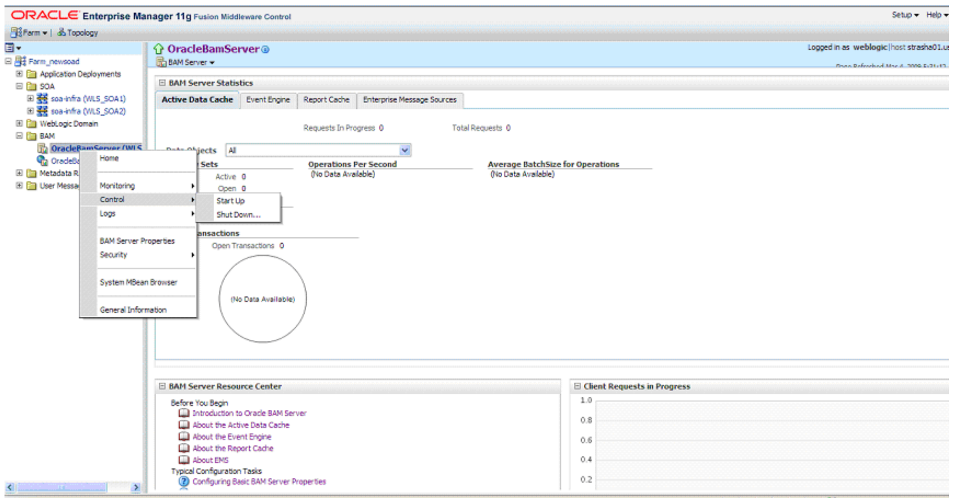
To install a new application or module for deployment to targets in this domain, click the Install button.

Deployments

	Name	State	Health	Type	Deployment Order
☐	FtpAdapter	Active	✓ OK	Resource Adapter	315
☐	JmsAdapter	Active	✓ OK	Resource Adapter	323
☐	jsf(1.2,1.2.9.0)	Active		Library	100
☐	jstl(1.2,1.2.0.1)	Active		Library	100
☐	MQSeriesAdapter	Active	✓ OK	Resource Adapter	317
☐	ohw-rcf(5,5.0)	Active		Library	100
☐	ohw-uix(5,5.0)	Active		Library	100
✓	oracle-bam (11.1.1)	Installed		Enterprise Application	300
☐	oracle.adf.dconfigbeans(1.0,11.1.1.0.0)	Active		Library	100
☐	oracle.bam.library(11.1.1,11.1.1)	Active		Library	100

Oracle Enterprise Manager Fusion Middleware Control allows multiple operations and configuration of the BAM Server as well as monitoring its status. For details about monitoring and controlling the Oracle BAM engine, *Oracle Fusion Middleware Administrator's Guide for Oracle SOA Suite*.

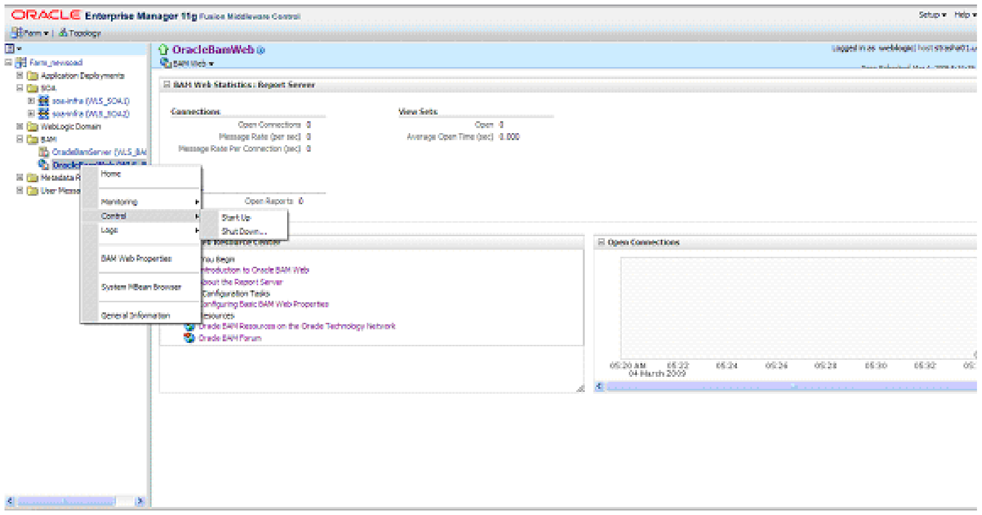
Figure 5–23 Startup and Shutdown of Oracle Business Activity Monitoring using Fusion Middleware Control



The BAM Web Applications run collocated by default in the same managed server as BAM Server. for example, the BAM server and BAM Web Applications are part of the same deployment in WebLogic Server Administration Console. They can't be stopped and managed separately from BAM Server by default. It is possible though, as explained in later sections, to manipulate WebLogic Server application targeting to separate both components. The Oracle WebLogic Administration Console allows starting and stopping Oracle BAM. This can also be done using WLST commands.

Oracle Enterprise Manager Fusion Middleware Control Console allows multiple operations and configuration of the BAM Web Applications

Figure 5–24 Oracle Enterprise Manager Fusion Middleware Control

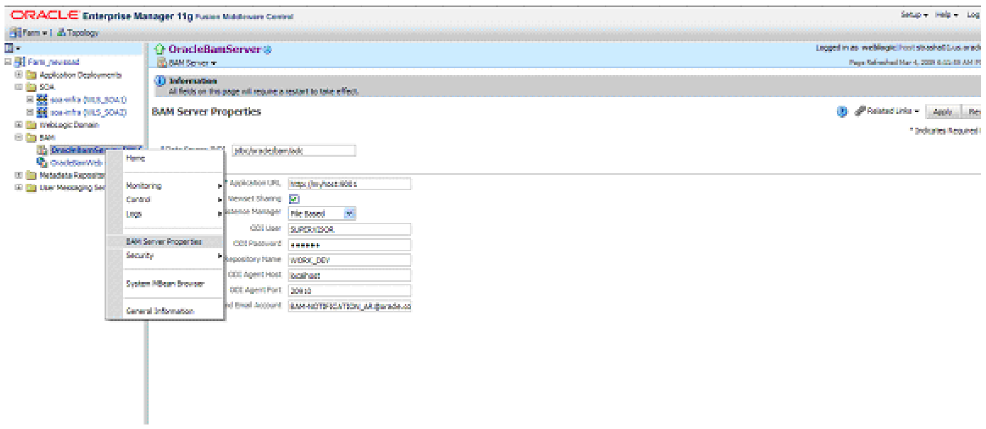


Note: Although Enterprise Manager offers separate start and stop for Oracle BAM Server and Oracle BAM Web Applications, this is in reality a stop operations that affects both components, meaning if Oracle BAM Server is stopped from Enterprise Manager, the corresponding Oracle BAM Web Applications are also stopped, and the reverse is also true. This also applies for start operations.

5.10.1.4 Oracle Business Activity Monitoring Configuration Artifacts

Oracle Enterprise Manager Fusion Middleware Control exposes some of the configuration options for Oracle BAM Server.

Figure 5–25 Configuring Oracle Business Activity Monitoring



Information
All fields on this page will require a restart to take effect.

BAM Server Properties

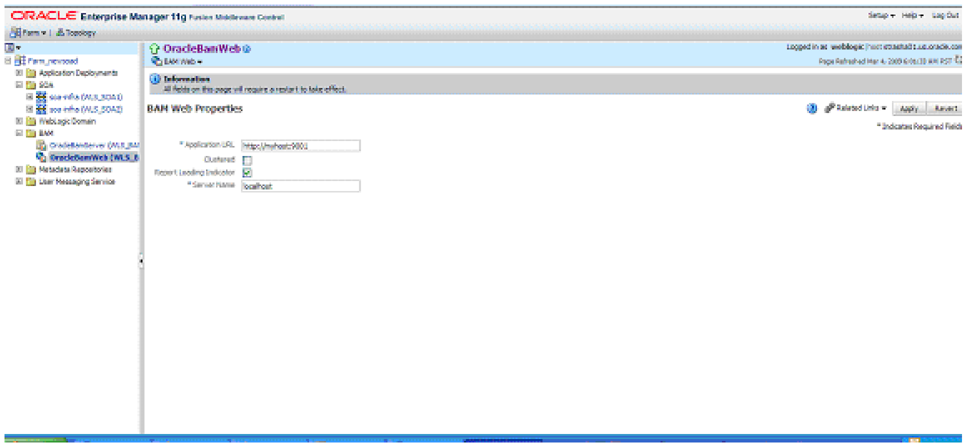
Data Source JNDI	jdbc/oracle/bam/adc
Application URL	http://myhost:9001
Viewset Sharing	✓
Licence Manager	File Based
ODI User	SUPERVISOR
ODI Password	******
Repository Name	WORK_DEV
ODI Agent Host	localhost
ODI Agent Port	20910
Email Account	BAM-NOTIFICATION_AR@oracle.co

The properties exposed by Oracle Enterprise Manager are a mix of the information contained in two different files: `BAMServerConfig.xml` and

`BAMCommonConfig.xml`. These files are located under the `DOMAIN_HOME/servers/BAM_Server_Name/tmp/_WL_user/oracle-bam_11.1.1/yhryfp/APP-INF/classes/config` directory (notice that `yhryfp` is the random directory generated at installation time for deploying BAM applications). This is the random directory generated at installation time for deploying Oracle BAM applications. The properties in these files are modifiable by using the Mbeans exposed in Oracle Enterprise Manager Fusion Middleware Control. For details on the configuration options for Oracle BAM Servers, refer to the *Oracle Fusion Middleware Administrator's Guide for Oracle SOA Suite* and *Oracle Fusion Middleware User's Guide for Oracle Business Activity Management*.

Similarly, some properties are exposed for Oracle BAM Web Applications by Oracle Enterprise Manager Fusion Middleware Control as shown in Figure 5–26.

Figure 5–26 Oracle Business Activity Monitoring Configuration Properties



Information
All fields on this page will require a restart to take effect.

BAM Web Properties

* Indicates Required Field

* Application URL	http://myhost:9001
Clustered	☐
Report Loading Indicator	✓
* Server Name	localhost

Configuration options at the container level, such as data sources and persistent stores, are maintained as part of Oracle WebLogic Server Domain configuration and are synchronized across a cluster of Oracle WebLogic Servers by Oracle WebLogic Server core infrastructure. See the *Oracle Fusion Middleware Administrator's Guide for Oracle SOA Suite* for details on configuring Oracle BAM.

5.10.2 Oracle Business Activity Monitoring High Availability Architecture and Failover Considerations

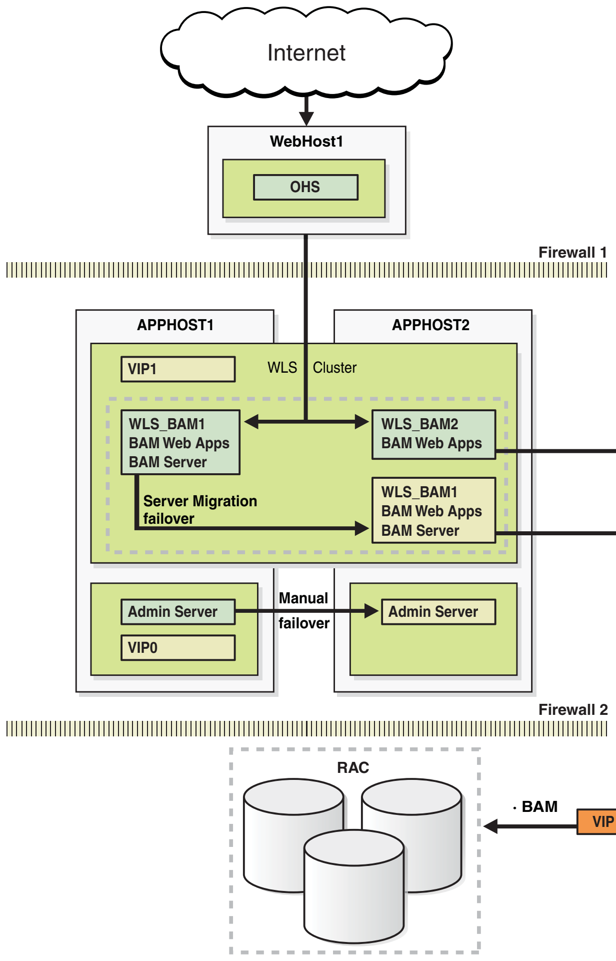
Figure 5–27 describes a two-node Oracle BAM cluster running on two Oracle WebLogic Servers. Oracle WebLogic Servers are front ended by Oracle HTTP Servers which load balances incoming requests to them. The following are the main characteristics of this configuration:

- Oracle BAM Web Applications run on two clustered WebLogic Server managed servers. The WebLogic Server Cluster synchronizes configuration for common artifacts of WebLogic Server used by Oracle BAM Web Applications, such as data sources, persistent store, and definitions. Oracle BAM Server is targeted by any of the servers where BAM Web applications are running. Only one WebLogic server runs Oracle BAM Server.
- The architecture uses Oracle WebLogic Server Migration feature to protect Oracle WebLogic Server that runs both Oracle BAM Web Applications and Oracle BAM Server (in APPHOST1) against failures. This provides protection for Oracle BAM Server which is a singleton. The WebLogic Managed Server in which Oracle BAM Server runs is listening on a Virtual IP that gets migrated to another node when

the failover occurs. This is the address that Oracle BAM Web Applications in APPHOST2 use to connect to an Oracle BAM Server. Plan appropriately to account for the scenario where Oracle BAM Server and two instances of Oracle BAM Web Applications are running on APPHOST2. For more information on Server Migration features, see Chapter 3, "High Availability for WebLogic Server."

- Oracle BAM's database is configured with Oracle Real Application Clusters (RAC) to protect from database failures. Oracle BAM Server performs the appropriate reconnection and operations retries if database instance failure occurs.

Figure 5–27 Oracle BAM High Availability Architecture



5.10.2.1 Oracle Business Activity Monitoring Protection from Failures and Expected Behavior

Oracle BAM Server and Oracle BAM Web applications are protected from all process failures by the WebLogic Server infrastructure. This section also describes expected behavior in the event of component failure.

Process Failure

Oracle BAM Server and BAM Web Applications are protected from all process failures by the WLS infrastructure. This section describes process failure considerations for Oracle BAM.

- If the WLS_BAMx server crashes, Node Manager attempts to restart it locally. It attempts to restart the servers according to the configured restart count threshold.
  - For failures related to the WebLogic Server on which the BAM Server is running, if repeated restarts fail, the WebLogic Server infrastructure attempts to perform a server migration to the other node in the cluster. Ongoing requests from the clients time out during failover. Once the server's restart completes on the other node, the clients should be restarted to continue routing to it. For opened reports running in the Web browser, if a request is delivered though the BAM WebApps running in the same managed server as BAM Server, a **Reconnecting** message appears in the BAM WebApps report. If the request is being delivered through the BAM WebApps running in the other node, no message appears. In this situation, and during failover, you may be viewing stale data (while a failover takes place or while connection from webapps to servers is reestablished). The BAMWebApps Servlet remains available, and reconnects as soon as BAM Server comes up. For new reports or requests, BAM Web Applications in APPHOST2 reconnect once the VIP is migrated and the managed server is restarted. BAM Web Applications listening on VIP1 become functional once the server migration completes. At this point, the HTTP Server restarts routing HTTP requests to the Managed Server.
  - Failures affecting the WebLogic Server where only the BAM Web Applications are running, do not affect the BAM system. Other BAM Web Applications (running in APPOHOST1) remain available and maintain session information by using the WebLogic Server Session replication cluster. Failover should be transparent.
- The `oracle-bam` application where BAM Server and BAM Web Applications run may be down due to failure in accessing resources, errors in reading the database, or other issues unrelated to the status of the managed server where it is located. Therefore, you should monitor Oracle SOA Service Infrastructure application and watch for errors caused by the application in the managed server logs, for log file locations, see Section 5.2.1.6, "Oracle SOA Service Infrastructure Log File Locations".
- Failover may not succeed for an open report if when the report is opened, only one BAM managed server in the cluster was up, and after starting an additional managed server in the cluster, no other operations were performed. One trigger for session state replication is opening a report. Session state replication is not triggered by active data updates.

Node Failure

For node failures in APPHOST2, the behavior in case of a node failure is equivalent to the process failure scenario: Oracle BAM Web Applications in the other node remain available and can serve requests. Sessions are preserved by the session replication framework provided by WebLogic Server and failover to the other node should be transparent.

For node failures in APPHOST1, server migration is triggered after the available server verifies the time stamp in the database leasing system. While failover occurs, clients are unable to feed data into the system and retry appropriately. Oracle BAM Web Applications connecting to the server attempt to reconnect until the VIP is migrated and the server is restarted.

Database Failure

For information about Oracle BAM database failure, see Section 5.2.2.1, "Oracle SOA Service Infrastructure Protection from Failures and Expected Behavior"

5.10.2.2 Oracle Business Activity Monitoring Cluster-Wide Configuration Changes

The standard Java EE artifacts that Oracle BAM Server and Oracle BAM Web Application use are configured as part of Oracle WebLogic Domain in which Oracle BAM is installed. Oracle WebLogic Clusters provide automatic configuration synchronization for artifacts such as data-sources, persistent stores, and JMS modules, across the WebLogic Server domain. At the same time, the WebLogic Server cluster is in charge of synchronizing the deployments, and libraries used by Oracle BAM Web Applications and Oracle BAM Server.

As explained in the single instance section Oracle BAM Server's and Oracle BAM Web Applications configuration are maintained in the `DOMAIN_HOME/servers/BAM_Server_Name/tmp/_WL_user/oracle-bam_11.1.1/yhryfp/APP-INF/classes/config` directory (notice that `yhryfp` is the random directory generated at installation time for deploying BAM applications). The properties in these files can be modified by using the Mbeans exposed in Oracle Enterprise Manager Fusion Middleware Control. The properties exposed through MBeans are specific to each server. The properties exposed through Enterprise Manager-specific screens are cluster-wide and are only modified on one server. All properties, whether applied in Enterprise Manager or in an MBean browser require a restart of Oracle WebLogic Servers where Oracle BAM runs. For details on the configuration options for BAM Server and BAM Web Applications see the *Oracle Fusion Middleware Administrator's Guide for Oracle SOA Suite* and *Oracle Fusion Middleware User's Guide for Oracle Business Activity Management*.

One of the configuration options related to high availability environments is the Application URL which is used to determine the front end host used by the system in a cluster configuration. This option is used to produce the copy shortcut URL for reports and alerts. The other relevant parameter for Oracle BAM high availability configuration is the Server Name in OracleBAM Web configuration screen. This parameter is used by Oracle Web Application to determine Oracle BAM server to which it connects for accessing the Active Data Cache.

For SOA high availability installations frontended by Oracle HTTP Server, monitoring should be done on the on the oRacle HTTP Server ports of the real backend servers. This is the case when a deployment is using all the components deployed to the SOA Managed Server. A simple HTTP monitor that pings the HTTP/HTTPS port and expects a pre-determined response in turn should suffice. If only a specific SOA component is being used (such as B2B), then a monitor that does a deeper level check all the way to the Managed server can be considered to validate the health of the component in use. Please check with your load balancer vendor on setting up the HTTP monitors with your load balancer.

5.11 Configuring High Availability for Oracle SOA Service Infrastructure and Component Service Engines

The procedures described in this section include setting up the Service engines contained in Oracle SOA Service Infrastructure system, such as Oracle BPEL PM, Oracle Mediator, Oracle Human Workflow and Oracle Decision Services, as well as Oracle B2B and Oracle User Messaging Service.

Note: Oracle strongly recommends reading the release notes for any additional installation and deployment considerations prior to starting the setup process.

Figure 5–28 represents the example architecture that the configuration steps in this section create.

Figure 5–28 Oracle SOA Service Infrastructure High Availability Architecture

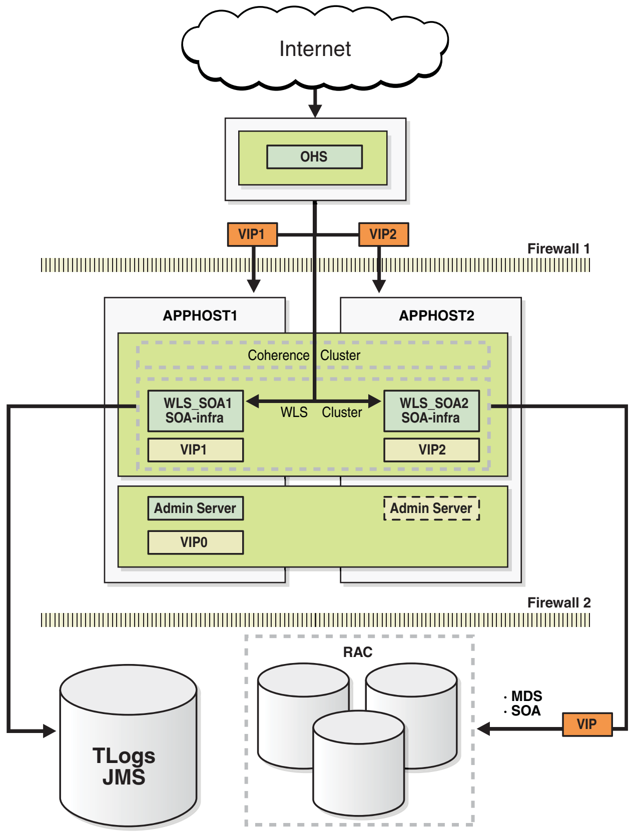


Figure 5–28 describes a two-node SOA cluster running on two Oracle WebLogic Servers. Oracle Servers are front ended by an Oracle HTTP Server, which load balances incoming requests. A separate Oracle WebLogic Server is used for custom logic and application deployment. This configuration uses an Oracle RAC database for storing metadata and SOA schemas, and shared storage for transaction and JMS stores. Virtual IP addresses (VIPs) provide manual failover for the administration server and for Oracle SOA Servers (for Server Migration). For more details about the components contained in this architecture, see the individual component sections in this chapter.

For information about configuring virtual IPs for the administration server and configuring the administration server for high availability, see Section 10.2.2.3, "Transforming the Administration Server for Cold Failover Clusters."

5.11.1 Preparing the Environment: Prerequisite Steps Before Setting up a SOA High Availability Configuration

The following sections provide prerequisite steps before setting up an Oracle SOA Service Infrastructure high availability configuration.

For information about platform-specific commands, see the *Oracle Fusion Middleware Installation Guide for Oracle SOA Suite*.

- Section 5.11.1.1, "Database Prerequisites"
- Section 5.11.1.2, "VIP and IP Prerequisites"
- Section 5.11.1.3, "Shared Storage Prerequisites"
- Section 5.11.1.4, "Installing and Configuring an LDAP Provider"
- Section 5.11.1.5, "Synchronizing System Clocks"
- Section 5.11.1.6, "Terminology for Directories and Directory Environment Variables"
- Section 5.11.1.7, "Using Oracle Fusion Middleware Repository Creation Utility to Load the Fusion Middleware Schemas in the Database"
- Section 5.11.1.8, "Installing and Configuring the Database Repository"

5.11.1.1 Database Prerequisites

Oracle SOA Service Infrastructure supports the following database versions:

- Oracle Database 10g (10.2.0.4 or later for non-XE database)
- Oracle Database 11g (11.1.0.7 or later for non-XE database)

To determine the database version, execute this query:

```
SQL>select version from sys.product_component_version where
product like 'Oracle%';
```

5.11.1.2 VIP and IP Prerequisites

As shown in Table 5–1, you configure the administration server and the SOA managed servers to listen on different virtual IPs. This requires the provisioning of the corresponding VIP in the node and related host names in the DNS system in your network. Make sure that the different VIPS are available and are reachable before running the installation.

Table 5–1 Virtual Hosts

Virtual IP	VIP Maps to...	Description
VIP0	APPHOST1VHN0	APPHOST1VHN0 is the virtual host name that is the listen address for the Administration Server and fails over with manual failover of the Administration Server. It is enabled on the node where the Admin Server process is running (APPHOST1 by default).
VIP1	APPHOST1VHN1	APPHOST1VHN1 is the virtual host name that maps to the listen address for WLS_SOA1 and fails over with server migration of this managed server. It is enabled on the node where WLS_SOA1 process is running (APPHOST1 by default).
VIP2	APPHOST2VHN2	APPHOST2VHN1 is the virtual host name that maps to the listen address for WLS_SOA2 and fails over with server migration of this managed server. It is enabled on the node where WLS_SOA2 process is running (APPHOST2 by default).

5.11.1.3 Shared Storage Prerequisites

For proper recovery in case of failure, store both JMS and transaction logs in a location that is accessible to all the nodes that can resume the operations after a failure in a managed server. This requires a shared storage location that can be referenced by multiple nodes. Table 5–2 lists the contents of shared storage.

Table 5–2 Contents of Shared Storage

Server	Type of Data	Vol in Shared Storage	Directory	Files
WLS_SOA1	Tx Logs	VOL1	*ORACLE_BASE*/admin/*domain_name*/*soa_cluster_name*/tlogs	Common location (stores decided by WebLogic Server)
WLS_SOA2	Tx Logs	VOL1	*ORACLE_BASE*/admin/*domain_name*/*soa_cluster_name*/tlogs	Common location (stores decided by WebLogic Server)
WLS_SOA1	JMS Stores	VOL1	*ORACLE_BASE*/admin/*domain_name*/*soa_cluster_name*/jms	Common location but Individual store per server (for example: SOAJMSStore1, UMSJMSStore1)
WLS_SOA2	JMS Stores	VOL1	*ORACLE_BASE*/admin/*domain_name*/*soa_cluster_name*/jms	Common location but Individual store per server (for example: SOAJMSStore2, UMSJMSStore2)

The shared storage can be a NAS or SAN device. The following is an example command based on a NAS device. Your options may be different from the ones specified in this section:

```
APPHOST1> mount nasfiler:/vol/volX/FMWshared
ORACLE_BASE/product/fmw -t nfs -o
rw,bg,hard,nointr,tcp,vers=3,timeo=300,rsize=32768,wsize=32768
```

5.11.1.4 Installing and Configuring an LDAP Provider

For production environments, it is a mandatory requirement for Oracle SOA Suite high availability topologies to have an external LDAP policy store. For more information on the supported policy stores as well as instructions on configuring LDAP, see the *Oracle Fusion Middleware Administrator's Guide for Oracle SOA Suite*.

5.11.1.5 Synchronizing System Clocks

Oracle recommends synchronizing system clocks on each of the cluster nodes for high availability SOA deployments.

5.11.1.6 Terminology for Directories and Directory Environment Variables

The follow list describes the directories and variables used in this chapter:

- ORACLE_BASE: This environment variable and related directory path refers to the base directory under which Oracle products are installed.
- MW_HOME: This environment variable and related directory path refers to the location where Fusion Middleware (FMW) resides.

- WL_HOME: This environment variable and related directory path contains installed files necessary to host a WebLogic Server.
- ORACLE_HOME: This environment variable and related directory path refers to the location where Oracle FMW SOA Suite is installed.
- DOMAIN Directory: This directory path refers to the location where the Oracle WebLogic Domain information (configuration artifacts) is stored.
- ORACLE_INSTANCE: An Oracle instance contains one or more system components, such as Oracle Web Cache, Oracle HTTP Server, or Oracle Internet Directory. An Oracle instance directory contains updateable files, such as configuration files, log files, and temporary files.

The values used and recommended for consistency for this directories are:

ORACLE_BASE:

```
/u01/app/oracle
```

MW HOME (Apptier):

```
ORACLE_BASE/product/fmw
```

WLS_HOME:

```
MW_HOME/wlserver_10.3
```

ORACLE_HOME:

```
MW_HOME/soa
```

Location for JMS file based stores and Tlogs:

```
ORACLE_BASE/admin/domain_name/soa_cluster_name/jms
ORACLE_BASE/admin/domain_name/soa_cluster_name/tlogs
```

Mount point is:

```
ORACLE_BASE/admin/domain_name/soa_cluster_name/
```

Shared Storage location:

```
ORACLE_BASE/admin/domain_name/soa_cluster_name/ (VOL1 or VOL2)
```

5.11.1.7 Using Oracle Fusion Middleware Repository Creation Utility to Load the Fusion Middleware Schemas in the Database

Install the 11g (11.1.1) Oracle Fusion Middleware Metadata Store and Oracle SOA schemas into a Real Application Cluster database before you install Oracle Fusion Middleware SOA components. Oracle Fusion Middleware provides a tool, Oracle Fusion Middleware Repository Creation Utility (RCU), to create the component schemas in an existing database. See Oracle Fusion Middleware Installation Concepts and Repository Creation Utility User's Guide for more information about installing RCU

5.11.1.7.1 Running RCU Run RCU to install the required metadata for Oracle Fusion Middleware 11*g*.

1. For linux, go to *RCU_Home*/bin and use the following command:

   ```
   ./rcu
   ```

2. In the Welcome screen, click **Next**.
3. In the Create Repository screen, select **Create** to load component schemas into a database, click Next.
4. In the Database Connection Details screen, enter connection information for your database:
   - Database Type: Select **Oracle Database**.
   - Host Name: Enter the name of the node that is running the database. For an Oracle RAC database, specify the VIP name, or one of the node names as the host name: **SOADBHOST1VIRTUAL**.
   - Port: The port number for the database: **1521**
   - Service Name: Enter the service name of the database: **soaha.mycompany.com**
   - Username: **SYS**
   - Password: Enter the password for the SYS user.
   - Role: **SYSDBA**
5. Click **Next**.
6. If you receive the following warning message:

   The database you are connecting is with non-UTF8 charset, if you are going to use this DB for multilingual support, you may have data loss. If you are not using for multilingual support you can continue, otherwise we strongly recommend using UTF-8 database.
7. Click **Ignore** or **Stop**.
8. In the Select Components screen, select **Create a New Prefix**, and enter a prefix to use for the database schemas, for example, **SOAHA**

   Write down the schema names so they are available in later procedures.

   - Select the following schemas:
     – Under AS Common Schemas, select **Metadata Services**.
     – Under SOA Service Infrastructure, select **SOA Service Infrastructure** and **User Messaging**.
9. Click **Next**.
10. In the Schema Passwords screen, enter passwords for the main and additional (auxiliary) schema users, and click **Next**.
11. In the Map Tablespaces screen, choose the tablespaces for the selected components, and click **Next**.
12. In the Summary screen, click **Create**.
13. In the Completion Summary screen, click **Close**.

See the *Oracle Fusion Middleware Repository Creation Utility User's Guide* for more information about using RCU.

5.11.1.8 Installing and Configuring the Database Repository

This section describes how to install and configure the database repository.

Oracle Clusterware

- For 10*g* Release 2 (10.2), see *Oracle Database Oracle Clusterware and Oracle Real Application Clusters Installation Guide*.
- For 11*g* Release 1 (11.1), see the *Oracle Clusterware Installation Guide*.

Automatic Storage Management

- For 10*g* Release 2 (10.2), see *Oracle Database Oracle Clusterware and Oracle Real Application Clusters Installation Guide*.
- For 11*g* Release 1 (11.1), see Oracle Real Application Clusters Installation Guide*Oracle Real Application Clusters Installation Guide*.
- When you run the installer, select the **Configure Automatic Storage Management** option in the **Select Configuration** page to create a separate Automatic Storage Management home.

Oracle Real Application Clusters

- For 10*g* Release 2 (10.2), see *Oracle Database Oracle Clusterware and Oracle Real Application Clusters Installation Guide*.
- For 11*g* Release 1 (11.1), see *Oracle Real Application Clusters Installation Guide*.

You must install the 11g (11.1.1) Oracle Fusion Middleware Repository into a Real Application Cluster database before you install the Oracle Fusion Middleware components. Oracle Fusion Middleware provides a tool, Oracle Fusion Middleware Repository Creation Utility (RCU), to create the component schemas in an existing database.

You install RCU in its own, separate Middleware home.

See *Oracle Fusion Middleware Repository Creation Utility User's Guide* for more information about installing RCU.

Database Initialization Parameters

Ensure that the following initialization parameter is set to the required value. It is checked by Repository Creation Assistant.

Table 5–3 Required Initialization Parameters

Parameter	Required Value	Parameter Class
`PROCESSES`	300 or greater	Static

To check the value of the initialization parameter using SQL*Plus, you can use the SHOW PARAMETER command.

As the SYS user, issue the SHOW PARAMETER command as follows:

```
SQL> SHOW PARAMETER processes
```

Set the initialization parameter using the following command:

```
SQL> ALTER SYSTEM SET processes=300 SCOPE=SPFILE
```

Restart the database.

Note: The method that you use to change a parameter's value depends on whether the parameter is static or dynamic, and on whether your database uses a parameter file or a server parameter file. See the *Oracle Database Administrator's Guide* for details on parameter files, server parameter files, and how to change parameter values.

5.11.2 Installing Oracle HTTP Server on WebHost1

To install Oracle HTTP Server on WEBHOST1:

1. Verify that the servers meet the following requirements:
   - The system, patch, kernel, and other requirements meet the requirements specified in the *Oracle Fusion Middleware Installation Guide for Oracle SOA Suite*.
   - Port 7777 is not used by any service on the nodes. You can verify this by running the following command:

     Unix:

     ```
     netstat -an | grep 7777
     ```

     Windows:

     ```
     netstat -an | findstr 7777
     ```

   If the ports are in use, make them available.
2. On UNIX platforms, if the `/etc/oraInst.loc` or `/var/opt/oracle/oraInst.loc` file exists, check that its contents are correct. Specifically, check that the inventory directory is correct, and that you have write permissions for that directory.

   If the `/etc/oraInst.loc` file does not exist, skip this step.
3. Start Oracle Universal Installer for Oracle Fusion Middleware 11*g* Web Tier Utilities CD installation as follows:

   For UNIX, run this command: `runInstaller`.

   For Windows, double-click **setup.exe**.
4. In the Welcome screen, click **Next**.
5. In the Select Installation Type screen, select **Install and Configure**, and click **Next**.
6. In the Prerequisite Checks screen, ensure that all the prerequisites are met, and click **Next**.
7. In the Specify Installation Location screen, set the location to:

   ```
   /u01/app/oracle/product/11.1.1/ohs_1
   ```
8. Click **Next**.
9. In the Configure Components screen:
   - Select **Oracle HTTP Server**.
   - Do not select **Associate Selected Components with WebLogic Domain**.
   - Click **Next**.
10. In the Specify Component Details screen, enter the following values:
    - Instance Home Location: Instance Home

- Instance Home Location: **/u01/app/oracle/product/11.1.1/ohs_1/instances/ohs_instance1**
- Instance Name: **ohs_instance1**
- OHS Component Name: **ohs1**

11. Click **Next**.
12. In the Specify Web Tier Port Details screen:
    - Select **Specify Custom Ports**. If you specify a custom port, select **Specify Ports using Configuration File**, and use the **Browse** function to select the file.
    - Enter the **Oracle HTTP Server port,** for example, **7777**.
13. Click **Next**.
14. In the Configuration Summary screen, ensure that the selections are correct, and click **Install**.
15. In the Installation Progress screen:

    For UNIX systems, a dialog box appears prompting you to run the `oracleRoot.sh` script. Open a command window and run the script, following the prompts.

    Click **Next**.
16. In the Configuration screen, several configuration assistants are launched in succession. When the configuration assistants are finished, the Configuration Completed screen appears.
17. In the Configuration Completed screen, click **Finish** to exit.

5.11.2.1 Validating Oracle HTTP Server

To verify that Oracle HTTP Server is set up correctly, access the root URL context of the server by using the following URL a browser:

HTTP://WebHost1:7777/

If Oracle HTTP Server is set up correctly, the **Hello World** page appears in the browser.

5.11.3 Installing Oracle Fusion Middleware Home

This section describes the procedure for installing Oracle Fusion Middleware on all nodes in the application tier that run Oracle WebLogic Server and Oracle Fusion Middleware SOA Suite.

Install the following Oracle Fusion Middleware components:

- Oracle WebLogic Server (see Section 5.11.3.1, "Installing Oracle WebLogic Server")
- Oracle Fusion Middleware SOA Suite (see Section 5.11.3.2, "Installing Oracle Fusion Middleware for Oracle SOA")

5.11.3.1 Installing Oracle WebLogic Server

To install Oracle WebLogic Server on all nodes in the application tier:

1. On UNIX platforms, if the `/etc/oraInst.loc` file exists, check that its contents are correct. Specifically, check that the inventory directory is correct and that you have write permissions for that directory.

   If the `/etc/oraInst.loc` file does not exist, skip this step.

2. Start Oracle WebLogic Server Installer

   On UNIX (Linux in the following example):

   ```
   APPHOST1> server103_linux32.bin
   ```

   On Windows:

   ```
   APPHOST1> server103_win32.exe
   ```

3. In the Specify Inventory Directory screen, do the following:

   a. Enter *HOME*/oraInventory, where HOME is the home directory of the user performing the installation (this is the recommended location).

   b. Enter the OS group for the user performing the installation.

   c. Click **Next**.

   d. Follow the instructions on to execute `createCentralInventory.sh` as root.

   e. Click **OK**.

4. In the Welcome screen, click **Next**.

5. In the Choose Middleware Home Directory screen:

   - Select **Create a New Middleware Home**.
   - For the **Middleware Home Directory** field, enter **ORACLE_ BASE/product/fmw**.
   - Click **Next**.

6. In the Register for Security Updates screen, enter your contact information for security update notifications, and click **Next**.

7. In the Choose Install Type screen, select **Custom**, and click **Next**.

8. In the Choose Products and Components screen, click **Next**.

9. In the JDK Selection screen, select only **JROCKIT SDK1.6.0_05**, and click **Next**.

10. In the Choose Product Installation Directories screen, accept the following directory:

    ```
    ORACLE_BASE/product/fmw/wlserver_10.3
    ```

    Click **Next**.

11. In the Installation Summary screen, click **Next**.

12. In the Installation Complete screen, deselect **Run QuickStart**, and click **Done**.

5.11.3.2 Installing Oracle Fusion Middleware for Oracle SOA

On Linux platforms, if the `/etc/oraInst.loc` file exists, check that its contents are correct. Specifically, check that the inventory directory is correct and that you have write permissions for that directory.

If the `/etc/oraInst.loc` file does not exist, you can skip this step.

1. Start Oracle Fusion Middleware 11*g* Oracle SOA Suite Installer:

   On UNIX:

   ```
   APPHOST1> runInstaller
   ```

On Windows:

```
APPHOST1> setup.exe
```

When Oracle Fusion Middleware 11g Oracle SOA Suite Installer prompts you for a **JRE/JDK location** enter Oracle SDK location created in Oracle WebLogic Server installation, for example, **ORACLE_BASE/product/fmw/jrockit_160_05_ R27.6.2-20**.

2. In the Specify Inventory Directory screen, do the following:
   a. Enter HOME/oraInventory, where HOME is the home directory of the user performing the installation (this is the recommended location).
   b. Enter the OS group for the user performing the installation.
   c. Click **Next**.
   d. Follow the instructions on the screen to execute `/createCentralInventory.sh` as root.
   e. Click **OK**.
3. In the Welcome screen, click **Next**.
4. In the Prerequisite Checks screen, verify that the checks complete successfully, and click **Next**.
5. In the Specify Installation Location screen:
   - For Middleware Home, enter **ORACLE_BASE/product/fmw**.
   - For Oracle Home Directory, enter **soa**.

   Click **Next**.
6. In the Installation Summary screen, click **Install**.
7. In the Installation Complete screen, click **Finish**.

5.11.4 Running Oracle Fusion Middleware Configuration Wizard on APPHOST1 to Create the SOA Domain

Run Oracle Fusion Middleware Configuration Wizard from the SOA home directory to create a domain containing the Administration Server and Oracle SOA components. Ensure that the database where you installed the repository is running. For RAC databases, all the instances must be running.

1. Change the directory to the location of Oracle WebLogic Server Configuration Wizard, located in the Middleware home, SOA directory:

   ```
   APPHOST1> cd ORACLE_HOME/common/bin
   ```

2. Start Oracle WebLogic Server Configuration Wizard:

   For Linux:

   ```
   APPHOST1> ./config.sh
   ```

   For Windows:

   ```
   APPHOST1> config.cmd
   ```

3. In the Welcome screen, select **Create a New WebLogic Domain**, and click **Next**.

4. In the Select Domain Source screen, select **Generate a domain configured automatically to support the following products**, and select the following products:

   - Oracle SOA Suite - 11.1.1.0 [soa]
   - Oracle Enterprise Manager - 11.1.1.0 [soa]
   - Oracle JRF - 11.1.1.0 [soa] (selected by default)
   - Oracle WSM Policy Manager 11.1.1.0 [soa] (selected by default)

   Click **Next**.

5. In the Specify Domain Name and Location screen, make the following entries:

   - Domain Name: **soadomain**
   - Domain Location: accept the default
   - Application Location: accept the default

   Click **Next**.

6. In the Configure Administrator Username and Password screen, enter the username and password to be used for the domain's administrator, and click **Next**.

7. In the Configure Server Start Mode and JDK screen, make the following selections:

   - WebLogic Domain Startup Mode: select **Production Mode**
   - JDK Selection: select **JROCKIT SDK1.6.0_05**.

   Click **Next**.

8. In the Configure JDBC Component Schema screen:

   a. Select all the component schemas that appear in the table at the bottom: **SOA Infrastructure**, **User Messaging Service**, **OWSM MDS Schema** and **SOA MDS Schema**.

   b. Select **Configure selected component schemas as RAC multi data source schemas in the next panel**.

   c. Ensure that the following data source appears on the screen. The user names shown in Table 5–4, assume that **soaha** was used as the prefix for schema creation from RCU.

      For information about multi data source configuration with RAC and the MDS repository, see Section 4.1.2, "Using Multi Data Sources with Oracle RAC."

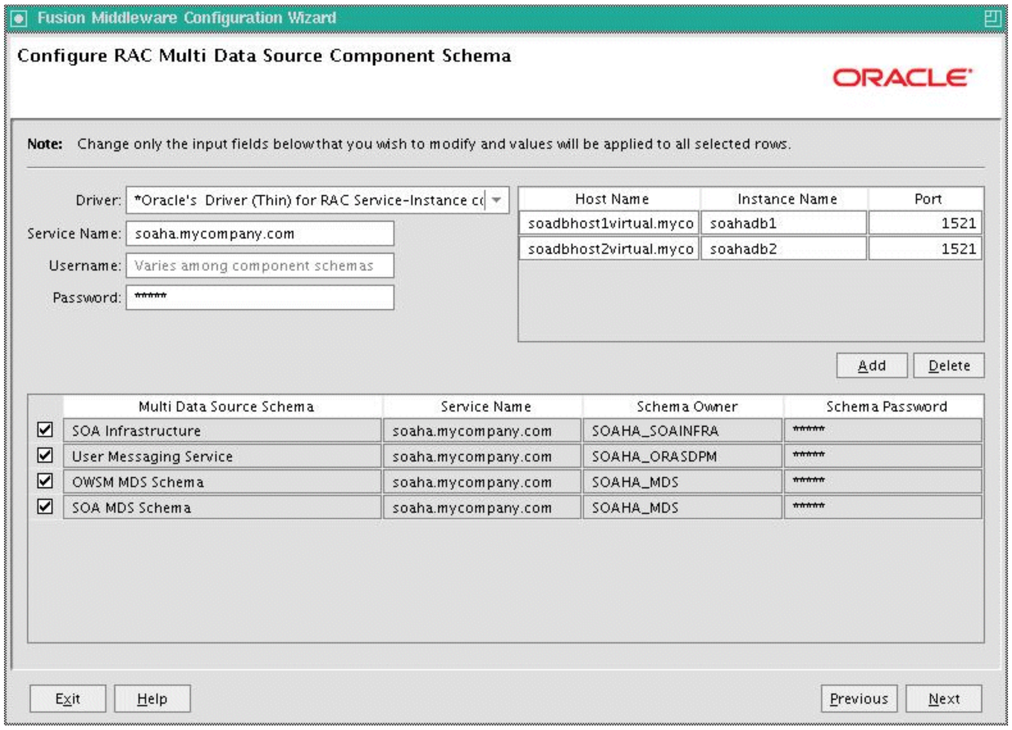
   Table 5–4 Configuring Values for Data Sources

Multi Data Source Schema	Schema Owner
SOA Infrastructure	SOAHA_SOAINFRA
User Messaging Service	SOAHA_ORASDPM
OWSM MDS Schema	SOAHA_MDS
SOA MDS Schema	SOAHA_MDS

   d. Click **Next**.

9. In the Configure RAC Multi Data Source Component Schema screen, enter values for the following fields, specifying the connect information for the RAC database that was seeded with RCU:

Figure 5–29 Configure RAC Multi Data Source Component Schema Screen


Fusion Middleware Configuration Wizard
Configure RAC Multi Data Source Component Schema
ORACLE
Add
Delete
Exit
Help
Previous
Next

Note: Change only the input fields below that you wish to modify and values will be applied to all selected rows.

Driver: *Oracle's Driver (Thin) for RAC Service-Instance c(
Service Name: soaha.mycompany.com
Username: Varies among component schemas
Password: ******

Host Name	Instance Name	Port
soadbhost1virtual.myco	soahadb1	1521
soadbhost2virtual.myco	soahadb2	1521

	Multi Data Source Schema	Service Name	Schema Owner	Schema Password
☑	SOA Infrastructure	soaha.mycompany.com	SOAHA_SOAINFRA	******
☑	User Messaging Service	soaha.mycompany.com	SOAHA_ORASDPM	******
☑	OWSM MDS Schema	soaha.mycompany.com	SOAHA_MDS	******
☑	SOA MDS Schema	soaha.mycompany.com	SOAHA_MDS	******

a. Driver: Select **Oracle driver (Thin) for RAC Service-Instance connections, Versions:10, 11**.

b. Service Name: Enter the service name of the database, for example, **soaha.mycompany.com**.

c. Username prefix: Enter the prefix for the schemas. The user names shown in Table 5–4 assume that **soaha** was used as prefix for schema creation from RCU.

d. Password and Confirm Password: Enter the password for access to the schemas.

e. Click **Add**, and enter the details for the first RAC instance.

f. Update each multi data source schema by selecting one data source at a time, adding the appropriate details.

   Make sure that the information is entered for all the multi data source schemas: **SOA Infrastructure**, **User Messaging Service**, **OWSM MDS Schema**, and **SOA MDS Schema**.

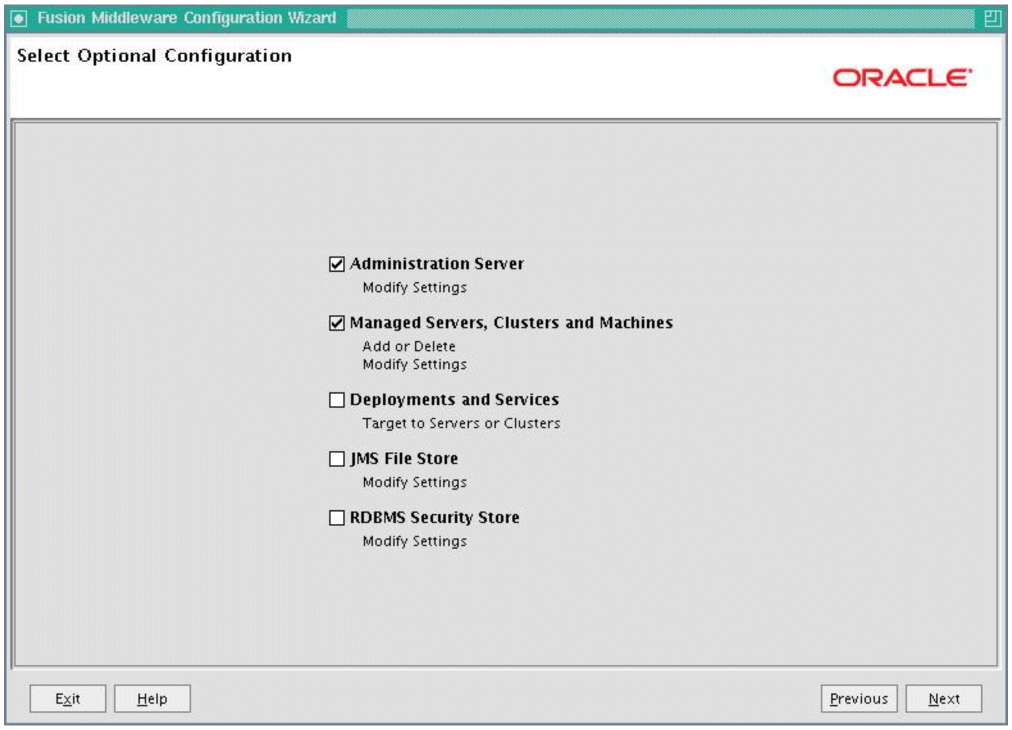
g. Click **Next**.

10. In the Test JDBC Data Sources screen, the connections are tested automatically. The **Status** column displays the results. Ensure that all connections were successful. If not, click **Previous** to return to the previous screen and correct your entries.

    Click **Next** when all the connections are successful.

11. In the Select Optional Configuration screen, select the following:

Figure 5–30 Select Optional Configuration Screen


Fusion Middleware Configuration Wizard
Select Optional Configuration
ORACLE
Exit
Help
Previous
Next

[✓] Administration Server
Modify Settings

[✓] Managed Servers, Clusters and Machines
Add or Delete
Modify Settings

[ ] Deployments and Services
Target to Servers or Clusters

[ ] JMS File Store
Modify Settings

[ ] RDBMS Security Store
Modify Settings

- **Administration Server**
- **Managed Servers, Clusters and Machines**

12. In the Customize Server and Cluster Configuration screen, select **Yes**, and click **Next**.
13. In the Configure the Administration Server screen, enter the following values:
    - Name: **AdminServer**
    - Listen Address: Enter the hostname for the VIP0 virtual IP
    - Listen Port: **7001**
    - SSL listen port: N/A
    - SSL enabled: leave unchecked

    Click **Next**.
14. In the Configure Managed Servers screen, add the following managed servers:

Table 5–5 Configuring Managed Servers

Name	Listen Address	Listen Port	SSL Listen Port	SSL Enabled
WLS_SOA1	VIP1	8001	n/a	No
WLS_SOA2	VIP2	8001	n/a	No

Note: Since WLS_SOA1 and WLS_SOA2 use server migration, it is required to use IPs as listen addresses instead of hostnames.

Do not delete any server that appears. You can modify the servers. If you delete a server and add a new one, targeting fails.

Note: Although the standard recommendation is to run custom applications and other systems in a separate WebLogic Managed Server, the creation of the custom WLS managed servers described in Figure 5–28 is not addressed here.

Click **Next**.

15. In the Configure Clusters screen, add the following cluster:

    - Name: **SOA_Cluster**
    - Cluster Messaging Mode: **unicast**
    - Multicast Address: N/A
    - Multicast Port: N/A
    - Cluster Address: Leave empty

    Click **Next**.

16. In the Assign Servers to Clusters screen, assign the following servers to SOA_ Cluster:

    - WLS_SOA1
    - WLS_SOA2

    Click **Next**.

17. In the Configure Machines screen:

    - Delete the **LocalMachine** that appears by default.
    - Click the **Unix Machine** tab, and add the following machines:

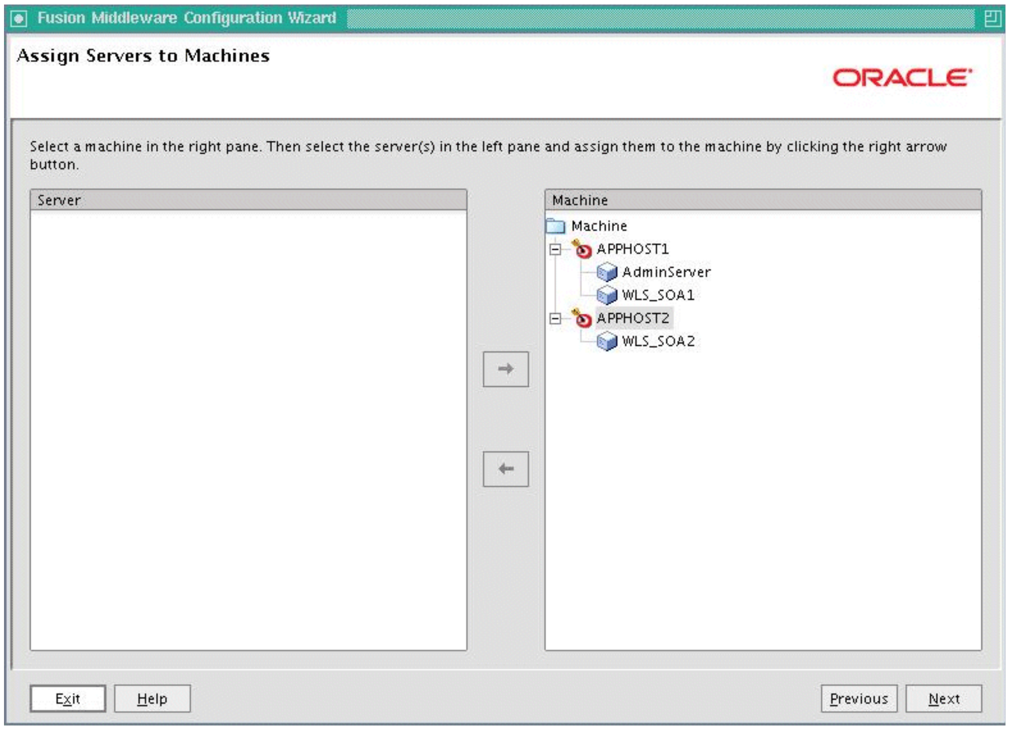
    Table 5–6 Configuring Machines

Name	Node Manager Listen Address
APPHOST1	Hostname of APPHOST1
APPHOST2	Hostname of APPHOST2

    Leave all other fields to their default values, and click **Next**.

18. In the Assign Servers to Machines screen, assign servers to machines as follows:

Figure 5–31 Assign Servers to Machines Screen


Fusion Middleware Configuration Wizard

Assign Servers to Machines

Select a machine in the right pane. Then select the server(s) in the left pane and assign them to the machine by clicking the right arrow button.

Server

Machine
- Machine
  - APPHOST1
    - AdminServer
    - WLS_SOA1
  - APPHOST2
    - WLS_SOA2

- APPHOST1: **AdminServer**, **WLS_SOA1**
- APPHOST2: **WLS_SOA2**

Click **Next**.

19. In the Review WebLogic Domain screen, click **Next**.
20. In the Configuration Summary screen, click **Create**.
21. In the Creating Domain screen, click **Done**.

Note: The multicast and unicast addresses are different from the ones used by the WebLogic Server cluster for cluster communication. SOA guarantees that composites are deployed to members of a single WebLogic Server cluster even though the communication protocol for the two entities (the WebLogic Server cluster and the groups to which composites are deployed) are different.

5.11.5 Creating boot.properties for the Administration Server on APPHOST1

This is an optional step for enabling the administration server to start up without prompting you for the administrator username and password. Create a boot.properties file for the administration server on APPHOST1.

For the Administration Server:

1. Create the following directories:

```
APPHOST1> mkdir ORACLE_BASE/product/fmw/user_
projects/domains/soadomain/servers/

APPHOST1> mkdir ORACLE_BASE/product/fmw/user_
projects/domains/soadomain/servers/AdminServer/
```

```
APPHOST1> mkdir ORACLE_BASE/product/fmw/user
_projects/domains/soadomain/servers/AdminServer/security
```

2. Use a text editor to create a file called `boot.properties` in the security directory created in the previous step, and enter the following lines in the file:

```
username=adminuser
password=password
```

Note: When you start up the administration server, the username and password entries in the file are encrypted.

For security reasons, minimize the time the entries in the file are left unencrypted. After you edit the file, start up the server as soon as possible in order for the entries to be encrypted.

5.11.6 Starting the System in APPHOST1

This section describes procedures for starting the system in APPHOST1

5.11.6.1 Starting the Administration Server on APPHOST1

To Start the Administration Server on APPHOST1 run the following commands:

```
APPHOST1> cd ORACLE_BASE/product/fmw/user_projects/domains/soadomain/bin

APPHOST1> ./startWebLogic.sh
```

5.11.6.2 Validating the Administration Server

To verify that the Administration Server is properly configured:

1. In a browser, go to `http://vip0:7001/console`.
2. Log in as the administrator.
3. Verify that the WLS_SOA1 and WLS_SOA2 managed servers are listed.
4. Verify that the SOA_Cluster cluster is listed.
5. Verify that you can access Enterprise Manager at `http://vip0:7001/em`.

5.11.6.3 Disabling Host Name Verification for the Administration Server and the WLS_SOA1 Managed Server

This step is required if you have not set up the appropriate certificates for hostname verification between the administration server and Node Manager. If SSL is not set up, you receive an error message unless you disable host name verification.

You can re-enable host name verification when you have set up SSL communication between the administration server and the Node Manager.

To disable host name verification:

1. In Oracle WebLogic Server Administration Console, select **Administration Server**, **SSL**, and then **Advanced**.
2. Set **Hostname Verification** to **None**.

3. In Oracle WebLogic Server Administration Console, select **WLS_SOA1**, **SSL**, and then **Advanced**.
4. Set **Hostname Verification** to **None**.
5. Save and activate the changes.
6. Restart the servers.

5.11.6.4 Starting Node Manager on APPHOST1

Perform these steps to start Node Manager on APPHOST1:

1. Run the `setNMProps.sh` script, which is located in the *ORACLE_HOME*/common/bin directory, to set the `StartScriptEnabled` property to `true` before starting Node Manager

   ```
   SOAHOST1> cd ORACLE_HOME/common/bin
   SOAHOST1> ./setNMProps.sh
   ```

   > **Note:** You must use the `StartScriptEnabled` property to avoid class loading failures and other problems.

2. Start Node Manager:

   ```
   ORACLE_BASE/product/fmw/wlserver_10.3/server/bin

   APPHOST1> ./startNodeManager.sh
   ```

5.11.6.5 Running the soa-createUDD.py Script

Before you can start the WLS_SOA1 managed server on APPHOST1, run the `soa-createUDD.py` script. This scripts transforms the regular JMS destination in highly available and load balanced Uniform Distribute Destinations making the system more scalable and robust.

You run the script using WLST offline:

1. Set up the environment.

   ```
   APPHOST1>. ORACLE_BASE/product/fmw/
   user_projects/domains/soadomain/bin/setDomainEnv.sh
   ```

2. Run the `soa-createUDD.py` script. This script is located at `ORACLE_HOME/bin/soa-createUDD.py`.

   ```
   APPHOST1> WL_HOME/jrockit_160_05_R27.6.2-20/bin/java weblogic.WLST
            ORACLE_HOME/bin/soa-createUDD.py
            --domain_home ORACLE_BASE/product/fmw/user_projects/domains/soadomain/
            --soacluster SOA_Cluster
   ```

3. Restart the Administration Server.
4. Verify that the following modules are listed in the Oracle WebLogic Server Administration Console:

   - **SOAJMSModuleUDDs**

     Verify this module by first expanding the **Services** node in the Domain Structure window and then expanding the **Messaging** node in the Oracle WebLogic Server Administration Console. Select **JMS Modules**. Select **SOAJMSMOduleUDDS** (represented as a hyperlink) in the Names column of the table. Click the **Subdeployment** tab. Click **SOAJMSSubDM** and then select **Targets**.

Verify that the module is targeted to SOAJMSServer_auto_1 and SOAJMSServer_auto_2.

- **UMSJMSSystemResource**

  Verify this module by first expanding the **Services** node in the Domain Structure window and then expanding the **Messaging** node in the Oracle WebLogic Server Administration Console. Select **JMS Modules**. Select **UMSJMSSystemResource** (represented as a hyperlink) in the Names column of the table. Click the **Subdeployment** tab. Click **SOAJMSSubDM** and then select **Targets**.

  Verify that the module is targeted to UMSJMSServer_auto_1 and UMSJMSServer_auto_2.

Note: Whenever the script is run after domain extension use the `--extend true` option, for example if the domain is extended to include Oracle BAM.

For example:

```
MW_HOME/jrockit_160_05__R27.6.2-20/bin/java weblogic.WLST
MW_HOME/soa/bin/soa-createUDD.py --domain_home MW_HOME/
user_projects/domains/soadomain --bamcluster BAM_Cluster --extend
true
```

5.11.6.6 Starting and Validating the WLS_SOA1 Managed Server

To start up the WLS_SOA1 managed server and check that it is configured correctly:

1. Start the WLS_SOA1 managed server using Oracle WebLogic Server Administration Console.
2. When WLS_SOA1 is started, the following URLs become available:
   - http://APPHOST1VHN1:8001/b2b
     – Verify login to B2B console
   - http://APPHOST1VHN1:8001/integration/worklistapp
     – Verify login to worklist console
   - http://APPHOST1VHN1:8001/wsm-pm
     – Verify the policy validator link

5.11.7 Install WebLogic Server and Oracle SOA on APPHOST2

Repeat the procedures for installing WebLogic Server and Oracle SOA for APPHOST2. The directory paths for binary files and domains used when installing new nodes must be exactly the same as those used for first node. If these paths and domains are not exactly the same as those used for the first node, failover does not occur.

5.11.8 Propagating the Domain Configuration to APPHOST2 with pack/unpack Utilities

Follow these steps to propagate the domain configuration to APPHOST2 using Pack/Unpack utilities:

1. Run the following pack command on APPHOST1 to create a template pack:

   ```
   APPHOST1> cd ORACLE_HOME/common/bin
   APPHOST1> ./pack.sh -managed=true
   -domain=ORACLE_BASE/product/fmw/user_projects/domains/soadomain/
   -template=soadomaintemplate.jar
   -template_name=soa_domain_template
   ```

2. Run the following command on APPHOST1 to copy the template file created in the previous step to APPHOST2:

   ```
   APPHOST1> scp soadomaintemplate.jar
    oracle@node2:ORACLE_HOME/common/bin
   ```

3. Run the unpack command on APPHOST2 to unpack the propagated template:

   ```
   APPHOST2> cd ORACLE_HOME/common/bin
   APPHOST2> ./unpack.sh
    -domain=ORACLE_BASE/product/fmw/user_projects/domains/soadomain/
    -template=soadomaintemplate.jar
   ```

Note: When you run the Fusion Middleware Configuration Wizard after a Pack/unpack procedure, products that already exist in the original domain, and appear in the Select Extension Source screen, are not selected or greyed out.

Templates created with the pack command are considered user-created templates. They are treated differently from default templates. User-created templates are self-contained and do not contain any information related to the creation of the original domain.

5.11.9 Extracting XEngine Files in the Second Node

To enable B2B's XEngine in the second node, it is required to extract the content of the ZEngine tar manually:

```
SOAHOST2>cd ORACLE_HOME/soa/thirdparty/edifecs
SOAHOST2>tar xzvf XEngine.tar.gz
```

5.11.10 Starting the System in APPHOST2

This section describes procedures for starting the system in APPHOST2.

5.11.10.1 Disabling Host Name Verification for the WLS_SOA2 Managed Server

Before you can start and verify the WLS_SOA2 managed server, you need to disable host name verification. You can re-enable it when you have set up SSL communication between the Administration Server and the Node Manager.

To disable host name verification:

1. In Oracle WebLogic Server Administration Console, select **WLS_SOA2**, **SSL** , and then **Advanced**.
2. Set **Hostname Verification** to **None**.

5.11.10.2 Starting Node Manager on APPHOST2

To start the Node Manager on APPHOST2, repeat the steps from Section 5.11.6.4, "Starting Node Manager on APPHOST1" on APPHOST2.

5.11.10.3 Starting and Validating the WLS_SOA2 Managed Server

To start up the WLS_SOA2 managed server and verify that it is configured correctly:

1. Start the WLS_SOA2 managed server using Oracle WebLogic Server Administration Console

   When WLS_SOA2 is started, the following URLs become available:

   http://VIP2:8001/b2b

   Verify login to B2B console.

   http://VIP2:8001/integration/worklistapp

   Verify login to worklist console.

   http:/VIP2:8001/wsm-pm

   Verify the policy validator link.

5.11.11 Configuring Oracle HTTP Server for the Administration Server and the WLS_SOAn Managed Servers

Enable Oracle HTTP Server to route to the Administration Server, that contains the WLS_SOAn managed servers, by setting the WebLogicCluster parameter to the list of nodes in the cluster.

1. Add the following lines to the `OHS_HOME/instances/ohs_instance1/config/OHS/ohs1/mod_wl_ohs.conf` file:

```
# SOA soa-infra app
<Location /soa-infra>
    SetHandler weblogic-handler
    WebLogicCluster apphost1vhn1:8001,apphost2vhn1:8001
    WLLogFile /tmp/web_log.log
</Location>

# Worklist
<Location /integration/>
    SetHandler weblogic-handler
    WebLogicCluster apphost1vhn1:8001,apphost2vhn1:8001
    WLLogFile /tmp/web_log.log
</Location>

<Location /DefaultToDoTaskFlow/>
    SetHandler weblogic-handler
    WebLogicCluster SOAHOST1VHN2:8001,SOAHOST2VHN1:8001
</Location>

# B2B
<Location /b2b>
    SetHandler weblogic-handler
    WebLogicCluster apphost1vhn1:8001,apphost2vhn1:8001
    WLLogFile /tmp/web_log.log
</Location>

# UMS prefs
```

```
<Location /sdpmessaging/userprefs-ui >
    SetHandler weblogic-handler
    WebLogicCluster apphost1vhn1:8001,apphost2vhn1:8001
    WLLogFile /tmp/web_log.log
</Location>
```

2. Restart Oracle HTTP Server on WEBHOST1:

```
WEBHOST1> OHS_HOME/instances/ohs_instance1/bin/opmnctl restartproc
ias-component=ohs1
```

5.11.11.1 Validating Access through Oracle HTTP Server

Verify the following URLS to ensure that appropriate routing and failover is working from the HTTP Serer to the SOA_Cluster.

1. While WLS_SOA2 is running, stop WLS_SOA1 from Oracle WebLogic Server Administration Console.
2. Access the following URLs and verify the appropriate functionality:
   - WebHost1:7777/soa-infra
   - WebHost1:7777/b2b
   - WebHost1:7777/integration/worklistapp
   - WebHost1:7777/sdpmessaging/userprefs-ui
   - WebHost1:7777/wsm-pm
3. Start WLS_SOA1 from Oracle WebLogic Server Administration Console.
4. Stop WLS_SOA2.
5. Access the following URLs and verify the appropriate functionality:
   - WebHost1:7777/soa-infra
   - WebHost1:7777/b2b
   - WebHost1:7777/integration/worklistapp
   - WebHost1:7777/sdpmessaging/userprefs-ui

5.11.11.2 Configuring JMS Persistence Store as Shared Across the Servers

Configure the location for all persistence stores to a directory visible from both nodes. Change all persistent stores to use this shared base directory.

From the Fusion Middleware Administration Console, select **Services**, **Persistence Store**, **Persistence_Store_Name**, and then **Directory**.

To enable resume of pending JMS messages, you must specify a location on a persistent storage solution (NAS, SAN) that is available to other servers in the cluster. Therefore, the directory that you enter must be accessible by both WLS_SOA1 and WLS_SOA2. This directory must exist before the server is restarted.

5.11.11.3 Configuring a Default Persistent Store for Transaction Recovery

Each server has a transaction log, which stores information about committed transactions coordinated by the server that may not have been completed. WebLogic Server uses the transaction log when recovering from system crashes or network failures. To take advantage of the migration capability of the Transaction Recovery Service for servers in a cluster, you must store the transaction log in a location that is available to a server and its backup servers, preferably on a dual-ported SCSI disk, or a Storage Area Network (SAN). To configure the default persistent store:

1. In the **Domain Structure** tree, expand **Environment** and select **Servers**.
2. Select the server you want to modify.
3. Select the **Configuration**, **Services** tab.
4. Under **Default Store**, in the **Directory** field, enter the path to the folder where you want the default persistent store to store its data files.

   To enable migration of the Transaction Recovery Service, you must specify a location on a persistent storage solution that is available to other servers in the cluster. Therefore, the directory that you enter must be accessible by both WLS_SOA1 and WLS_SOA2. This directory must exist before the server is restarted.

5.11.12 Setting the Front End HTTP Host and Port

You need to set the front end HTTP host and port for Oracle WebLogic Server cluster:

1. In the WebLogic Server Administration Console, in the Change Center section, click **Lock & Edit**.
2. In the left pane, select **Environment** > **Clusters**.
3. Select the WLS_SOA cluster.
4. Select **HTTP**.
5. Set the values for the following:
   - **Frontend Host**: webhost1

     Frontend HTTP Port: 7777
   - **Frontend HTTPS Port**: leave it blank

   Note: Make sure this address is correct and available (the hTTP server is up). An incorrect value, for example, **http://** in the address, or trailing **/** in the hostname, may prevent the SOA system from being accessible even when using the virtual IPs to access it.

6. Click **Save**.
7. To activate the changes, click **Activate Changes** in the Change Center section of the Administration Console.
8. If you have started the server before, notice this change requires a restart of the participants in the cluster.

Note: The SOA system calculates the callback URL as follows:

- If a request to SOA originates from an external or internal service is originating a request to SOA, then SOA uses the callback URL specified by the client.
- If a request to an external or internal service originates from SOA, the callbackURL cannot be populated in the SOA request dynamically because SOA is the originator. Instead, callbackServerURL is used if it is specified as a binding property for the specific reference. (You can set this when modeling the composite or at runtime using the MBeans accessed through Oracle Enterprise Manager console.) This allows different service calls to have different callback URLs. That is, a callback URL from an external service is different than one to an internal service.

  However, if the callbackServer URL is not specified as a binding property for that reference, then the system uses the callback URL as specified in `soa-infra-config`. If the callback URL is not specified in `soa-infra-config`, then the system uses the callback URL as the front end host specified in WLS. If the front end host is not specified in WLS, the system uses the callback URL as the local host name as provided by WLS MBean APIs.

For SOA high availability installations frontended by Oracle HTTP Server, monitoring should be done on the on the Oracle HTTP Server ports of the real backend servers. This is the case when a deployment is using all the components deployed to the SOA Managed Server. A simple HTTP monitor that pings the HTTP/HTTPS port and expects a pre-determined response in turn should suffice. If only a specific SOA component is being used (such as B2B), then a monitor that does a deeper level check all the way to the Managed server can be considered to validate the health of the component in use. Please check with your load balancer vendor on setting up the HTTP monitors with your load balancer.

If you do not set the front end HTTP host and port, you get the following message when trying to retrieve a document definition XSD from Oracle B2B:

```
An error occured while loading the document definitions.
java.lang.IllegalArgumentException: Cluster address must be set when
clustering is enabled.
```

5.11.13 Using Oracle Coherence for Deploying Composites

Although deploying composites uses multicast communication by default, Oracle recommends using unicast communication for SOA high availability. Use unicast if you disable multicast communication for security reasons.

Note: An incorrect configuration of the Oracle Coherence framework that is used for deployment may prevent the SOA system from starting. The deployment framework must be properly customized for the network environment on which the SOA system runs. Oracle recommends the following configuration described in this section.

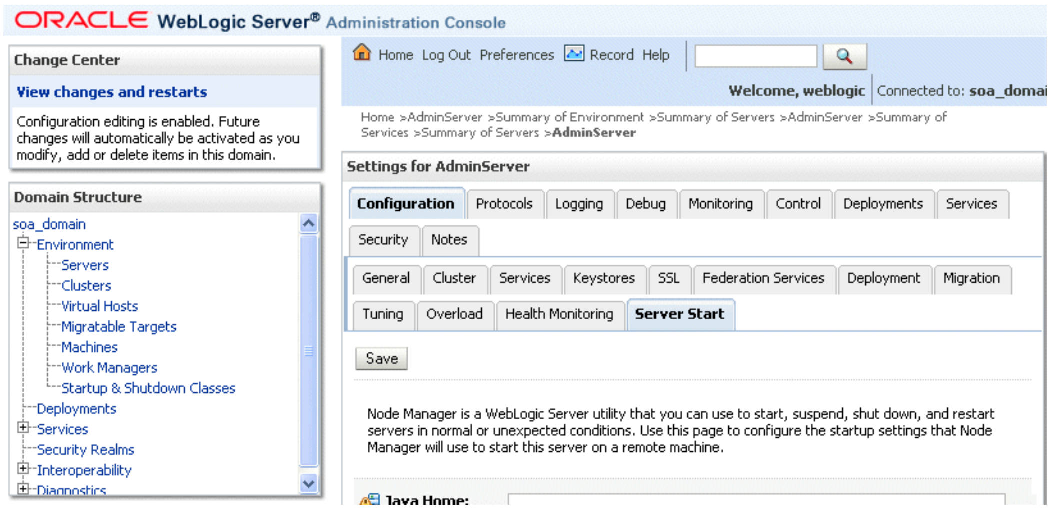
Enabling Communication within Clusters Using Unicast Communication

Multicast communication enables Oracle Fusion Middleware SOA to discover all of the members of a cluster to which to it deploys composites dynamically. However, unicast communication does not enable nodes to discover other cluster members in this way. Consequently, you must specify the nodes that belong to the cluster. You do not need to specify all of the nodes of a cluster, however. You need only specify enough nodes so that a new node added to the cluster can discover one of the existing nodes. As a result, when a new node has joined the cluster, it is able to discover all of the other nodes in the cluster. Additionally, in a configuration where multiple IPs are available in the same node, you must configure Oracle Coherence to use a specific hostname to create Oracle Coherence cluster.

> **Tip:** To guarantee high availability during deployments of SOA composites, specify enough nodes so that at least one of them is running at any given time.

Specify the nodes using the `tangosol.coherence.wka`*<n>* system property, where *<n>* is a number between 1 and 9. You can specify up to 9 nodes. Start the numbering at 1. This numbering must be sequential and must not contain gaps. In addition, specify the hostname used by Oracle Coherence to create a cluster through the `tangosol.coherence.localhost` system property. This hostname should be the virtual hostname used by the SOA servers that maps the corresponding listener addresses (VIP1 and VIP2). Set this property by adding the `-Dtangosol.coherence.localhost` parameters to the Arguments field of Oracle WebLogic Server Administration Console's Server Start tab (Figure 5–32).

Figure 5–32 Setting the Hostname Using the Start Server Tab of Oracle WebLogic Server Administration Console


ORACLE WebLogic Server® Administration Console
Home Log Out Preferences Record Help
Welcome, weblogic Connected to: soa_domai
Change Center
View changes and restarts
Domain Structure
soa_domain
Environment
Servers
Clusters
Virtual Hosts
Migratable Targets
Machines
Work Managers
Startup & Shutdown Classes
Deployments
Services
Security Realms
Interoperability
Diagnostics
Home >AdminServer >Summary of Environment >Summary of Servers >AdminServer >Summary of Services >Summary of Servers >AdminServer
Settings for AdminServer
Configuration Protocols Logging Debug Monitoring Control Deployments Services
Security Notes
General Cluster Services Keystores SSL Federation Services Deployment Migration
Tuning Overload Health Monitoring Server Start
Save
Java Home:

Configuration editing is enabled. Future changes will automatically be activated as you modify, add or delete items in this domain.

Node Manager is a WebLogic Server utility that you can use to start, suspend, shut down, and restart servers in normal or unexpected conditions. Use this page to configure the startup settings that Node Manager will use to start this server on a remote machine.

Specifying the hostname

To add the hostname used by Oracle Coherence:

1. Log into Oracle WebLogic Server Administration Console.
2. In the Domain Structure window, expand the **Environment** node.
3. Click **Servers**. The Summary of Servers page appears.
4. Click the name of the server (represented as a hyperlink) in Name column of the table. The settings page for the selected server appears.
5. Click the **Server Start** tab (illustrated in Figure 5–32).
6. Enter the following for WLS_SOA1 and WLS_SOA2 into the Arguments field.

For WLS_SOA1, enter the following:

```
-Dtangosol.coherence.wka1=apphost1vhn1
-Dtangosol.coherence.wka2=apphost2vhn1
-Dtangosol.coherence.localhost=apphost1vhn1
```

For WLS_SOA2, enter the following:

```
-Dtangosol.coherence.wka1=apphost1vhn1
-Dtangosol.coherence.wka2=apphost2vhn1
-Dtangosol.coherence.localhost=apphost2vhn1
```

7. Click **Save** and activate the changes.
8. This change requires the SOA servers to be restarted.

Note: The multicast and unicast addresses are different from the ones used by the WebLogic Server cluster for cluster communication. SOA guarantees that composites are deployed to members of a single WebLogic Server cluster even though the communication protocol for the two entities (the WebLogic Server cluster and the groups to which composites are deployed) are different.

5.11.14 Deploying Applications

You can deploy SOA composite applications from Oracle Enterprise Manager Fusion Middleware Control Console with the Deploy SOA Composite wizard. Use the Deploy SOA Composite wizard to deploy any of the following:

- A new SOA composite application for the first time
- A new revision (for example, 2.0) alongside an older revision (for example, 1.0) without impacting the latter. The revision deployed last becomes the new default revision of that composite (unless you specify otherwise at a later step during deployment).
- A bundle (ZIP file) containing multiple SOA composite application revisions (for example, revisions 2.0, 3.0, and 4.0) of a SOA composite application that already has a different revision that is currently deployed (for example, 1.0). This option enables you to deploy revisions 1.0, 2.0, 3.0, and 4.0 at the same time. The bundle can also contain revisions of different composites. There is no restriction that all revisions must be of the same composite application.

Deployment extracts and activates the composite application in SOA Service Infrastructure. Once an application is deployed, you can perform administration tasks, such as creating instances, configuring properties, monitoring performance, managing instance life cycles, and managing policies and faults.

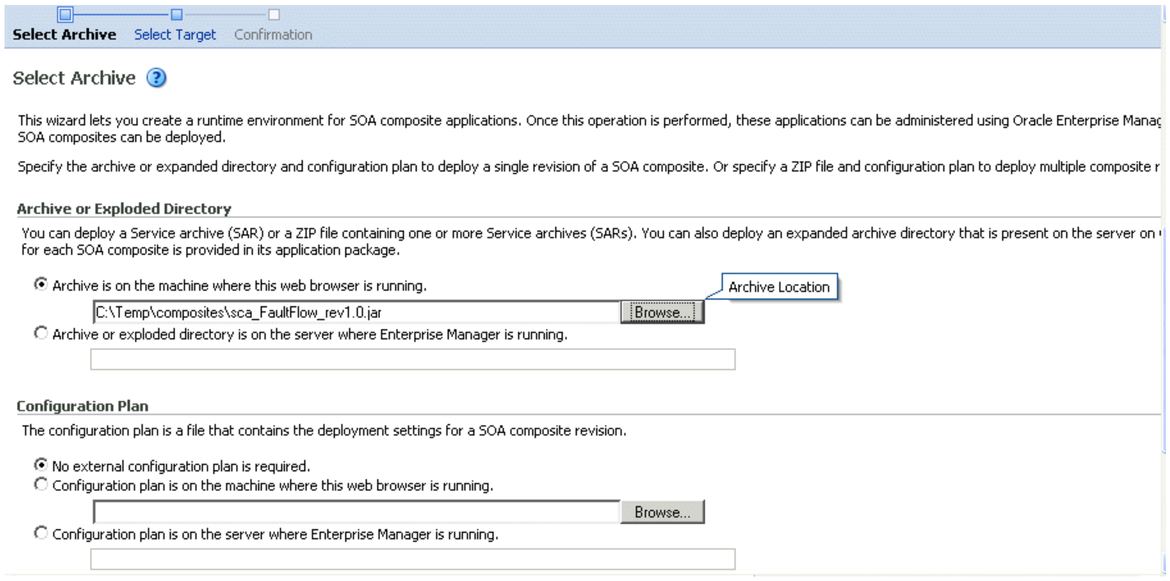
Note: If you want to redeploy an *existing* revision of an application, do *not* use this wizard. Instead, use the Redeploy SOA Composite wizard.

To deploy applications:

1. Access the Deploy SOA Composite wizard through one of the following options:

From SOA Infrastructure Menu...	From the SOA Folder in the Navigator...	From the SOA Infrastructure Home Page...	From the SOA Composite Menu...
1. Select **SOA Deployment** > **Deploy**.	1. Right-click **soa-infra**. 2. Select **SOA Deployment** > **Deploy**.	1. Click the **Deployed Composites** tab. 2. Above the **Composite** table, click **Deploy**.	1. Select **SOA Deployment** > **Deploy Another Composite.**

The Select Archive page appears.



Select Archive

This wizard lets you create a runtime environment for SOA composite applications. Once this operation is performed, these applications can be administered using Oracle Enterprise Manag
SOA composites can be deployed.

Specify the archive or expanded directory and configuration plan to deploy a single revision of a SOA composite. Or specify a ZIP file and configuration plan to deploy multiple composite r

Archive or Exploded Directory

You can deploy a Service archive (SAR) or a ZIP file containing one or more Service archives (SARs). You can also deploy an expanded archive directory that is present on the server on
for each SOA composite is provided in its application package.

[✓ Archive is on the machine where this web browser is running.]
C:\Temp\composites\sca_FaultFlow_rev1.0.jar
Archive or exploded directory is on the server where Enterprise Manager is running.

Configuration Plan

The configuration plan is a file that contains the deployment settings for a SOA composite revision.

[✓ No external configuration plan is required.]
Configuration plan is on the machine where this web browser is running.
Configuration plan is on the server where Enterprise Manager is running.

2. In the **Archive or Exploded Directory** section, specify the archive of the SOA composite application to deploy. The archive contains the project files of the composite to be deployed (for example, **HelloWorld_rev1.0.jar** for a single archive or **OrderBooking_rev1.0.zip** for multiple archives).
3. In the **Configuration Plan** section, optionally specify the configuration plan to include with the archive. The configuration plan enables you to define the URL and property values to use in different environments. During process deployment, the configuration plan is used to search the SOA project for values that must be replaced to adapt the project to the next target environment.
4. Click **Next**.

   The Select Target page appears.
5. Select the WebLogic Server or cluster to which to deploy the SOA composite application archive. You can deploy to multiple servers and clusters.
6. Click **Next**.

   The Confirmation page appears.
7. Review your selections.
8. Select whether or not to deploy the SOA composite application as the default revision. The default revision is instantiated when a new request comes in.
9. Click **Deploy**.

   Processing messages are displayed.
10. When deployment has completed, click **Close**.

See Also: *Oracle Fusion Middleware Developer's Guide for Oracle SOA Suite* for instructions on creating configuration plans and deploying applications from Oracle JDeveloper

5.11.15 Configuring Server Migration for the WLS_SOA Servers

The high availability architecture for a SOA system uses server migration to protect some singleton services against failures. For more information on Whole Server Migration see *Oracle Fusion Middleware Using Clusters for Oracle WebLogic Server*.

The WLS_SOA1 managed server is configured to be restarted on APPHOST2 in case of failure, and the WLS_SOA2 managed server is configured to be restarted on APPHOST1 in case of failure. For this configuration the WLS_SOA1 and WLS_SOA2 servers listen on specific floating IPs that are failed over by WLS Server Migration. To configure server migration for the WLS_SOAn managed servers, follow these steps:

Step 1 Set up user and tablespace for the Server Migration leasing table

1. Create a tablespace called `leasing`.

   Example: Log on to SQL*Plus as the sysdba user and run the following command:

   ```
   SQL> create tablespace leasing
          logging datafile 'DB_HOMEs/oradata/orcl/leasing.dbf'
          size 32m autoextend on next 32m maxsize 2048m extent management local;
   ```

2. Create a user named `leasing` and assign to it the leasing tablespace.

   ```
   SQL> create user leasing identified by welcome1;

   SQL> grant create table to leasing;

   SQL> grant create session to leasing;

   SQL> alter user leasing default tablespace leasing;

   SQL> alter user leasing quota unlimited on LEASING;
   ```

3. Create the leasing table using the `leasing.ddl` script.

   a. Copy the `leasing.ddl` file, located in the *ORACLE_BASE*/product/fmw/wlserver_10.3/server/db/oracle/817 or *ORACLE_BASE*/product/fmw/wlserver_10.3/server/db/oracle/920 directories, to your database node.

   b. Connect to the database as the leasing user.

   c. Run the `leasing.ddl` script in SQL*Plus.

      ```
      SQL> @copy_location/leasing.ddl;
      ```

Step 2 Create a Multi Data Source from Oracle WebLogic Server Administration Console

The second step is to create a multi data source for the leasing table from the Oracle WebLogic Server Administration Console:

You create a data source to each of the Oracle RAC database instances during the process of setting up the multi data source, both for these data sources and the global leasing multi data source. When you create a data source:

- Make sure that this is a non-xa data source.

- The names of the multi data sources are in the following format: *<MultiDS>-rac0*, *<MultiDS>-rac1*
- Use Oracle's Driver (Thin) Version 9.0.1, 9.2.0, 10, 11.
- Use Supports Global Transactions, One-Phase Commit, and specify a service name for your database.
- Target these data sources to the SOA cluster.

To create a multi data source:

1. From Domain Structure window in the Oracle WebLogic Server Administration Console, expand the **Services** node, then expand the **JDBC** node.
2. Click Multi Data Sources. The Summary of JDBC Multi Data Source page appears.
3. Click **New**. The Create a New JDBC Multi Data Source page appears.
4. Click **Lock and Edit**.
5. Click **New**.
6. Enter leasing as the Name
7. Enter jdbc/leasing as the JNDI name.
8. Select **Failover as algorithm (default)**.
9. Click **Next.**
10. Select **non-XA driver (the default)**.
11. Click **Next**.
12. Click **Create New Data Source**.
13. Enter *leasing-rac0* as name. Enter *jdbc/leasing-rac0* as JNDI name. Enter *oracle* as the database type. For the driver type, enter *Oracle Driver (Thin) for RAC server-Instance connection Version 10,11.*
14. Click **Next**.
15. Deselect **Supports Global Transactions**.
16. Click **Next**.
17. Enter the service name, database name, host port, and password for your leasing schema.
18. Click **Next**.
19. Click **Test Configuration** and verify the connection works.
20. Target the data source to the SOA cluster.
21. Select the data source and add it to the right screen.
22. Click **Create a New Data Source** and repeat the steps for the second instance of your RAC database.
23. Add the second data source to your multi data source.
24. Save and Activate the changes.

Step 3 Edit the Node Manager's properties file

For information on Node manager and Whole Server Migration, see Section 3.9, "Whole Server Migration."

The `nodemanager.properties` file is located in the `ORACLE_BASE/product/fmw/wlserver_10.3/common/nodemanager` directory. For server migration to work properly, you need to add the properties listed in this section:

- `Interface=eth0`

  This property specifies the interface name for the floating IP (eth0, for example).

  Be sure that the interface provided is the public interface for this node. On multi home nodes, this interface should be the one on which the floating IP can be enabled.

  > **Note:** For Windows, the Interface should be set to the Network Interface Name. For example: `Interface="Local Area Connection"`.

- `NetMask=255.255.255.0`

  This property specifies the net mask for the interface for the floating IP. The netmask provided (255.255.255.0) is just an example. The actual value depends on your network.

- `UseMACBroadcast=true`

  This property specifies whether or not to use a node's MAC address when sending ARP packets, that is, whether or not to use the -b flag in the arping command.

After starting Node Manger, verify in Node Manager's output (shell where Node Manager is started) that these properties are being used, or problems may arise during migration. You should see something like this in the Node Manager's output:

```
StateCheckInterval=500

Interface=eth0

NetMask=255.255.255.0

UseMACBroadcast=true
```

> **Note:** The steps in this section are not required if the server properties (start properties) have been properly set and the Node Manager can start the servers remotely.

1. Set the following property in the nodemanager.properties file.
   - StartScriptEnabled

     Set this property to true.
2. Start the Node Manager on Node 1 and Node 2 by running the startNodeManager.sh script located in the *ORACLE_BASE*/product/fmw/wlserver_10.3/server/bin directory.
3. Validate the changes to the nodemanager.properties file by checking the nodemanager.log file.

Step 4 Set environment and superuser privileges for the wlsifconfig.sh script

1. Grant sudo privilege to the WebLogic user ('oracle') with No Password restriction, and grant execute privilege on `/sbin/ifconfig and /sbin/arping` binaries.

Make sure the script is executable by the WebLogic user ('oracle'). The following is an example of an entry inside `/etc/sudoers` granting sudo execution privilege for `oracle` and also over the `ifconfig` and `arping`:

```
oracle ALL=NOPASSWD: /sbin/ifconfig,/sbin/arping
```

Step 5 Configure Server Migration Targets

Configuring Cluster Migration sets the `DataSourceForAutomaticMigration` property to true. Follow the steps in this section to configure cluster migration in a migration in a cluster:

1. Log into Oracle WebLogic Server Administration Console
2. In the left pane, expand **Environment** and select **Clusters**.
3. Select the cluster for which you want to configure migration (**SOA_Cluster**).
4. Click **Migration**.
5. In the **Available** field, select the machine to which to allow migration and click the right arrow. In this case, select **APPHOST1** and **APPHOST2**.
6. Select the data source to be used for automatic migration. In this case select the leasing data source.
7. Click **Save**.
8. Set the Candidate Machines for Server Migration. This needs to be done for all the managed servers.

   To do this, follow these steps:

   a. In the left pane of Oracle WebLogic Server Administration Console, expand **Environment** and select **Servers**.
   b. Select the server for which you want to configure migration.
   c. Click the **Migration** tab.
   d. In the **Available** field, located in the **Migration Configuration** section, select the machines to which to allow migration and click the right arrow. For **WLS_SOA1**, select **APPHOST2**. For **WLS_SOA2**, select **APPHOST1**.
   e. Select the checkbox for **Automatic Server Migration Enabled**. This enables the Node Manager to start a failed server on the target node automatically.
   f. Click **Save** and activate the changes.

Step 6 Test Server Migration

To verify that Server Migration is working properly, follow these steps:

From Node 1:

1. Force stop the WLS_SOA1 managed server.

   To do this, run this command:

   ```
   APPHOST1> kill -9 <pid>
   ```

   pid specifies the process ID of the managed server. You can identify the pid in the node by running this command:

   ```
   APPHOST1> ps -ef | grep WLS_SOA1
   ```

Note: For Windows, the Managed Server can be terminated by using the `taskkill` command. For example:

```
taskkill /f /pid <pid>
```

Where <*pid*> is the process Id of the Managed Server.

To determine the process Id of Managed server run the following command and identify the pid of the WLS_SOA Managed Server.

```
MW_HOME\jrockit_160_05_R27.6.2-20\bin\jps -l -v
```

---

2. Watch the Node Manager console: you should see a message indicating that WLS_SOA1's floating IP has been disabled.
3. Wait for the Node Manager to try a second restart of WLS_SOA1. Node Manager waits for a fence period of 30 seconds before trying this restart.
4. Once Node Manager restarts the server, stop it again. Now Node Manager should log a message indicating that the server will not be restarted again locally.

From Node2:

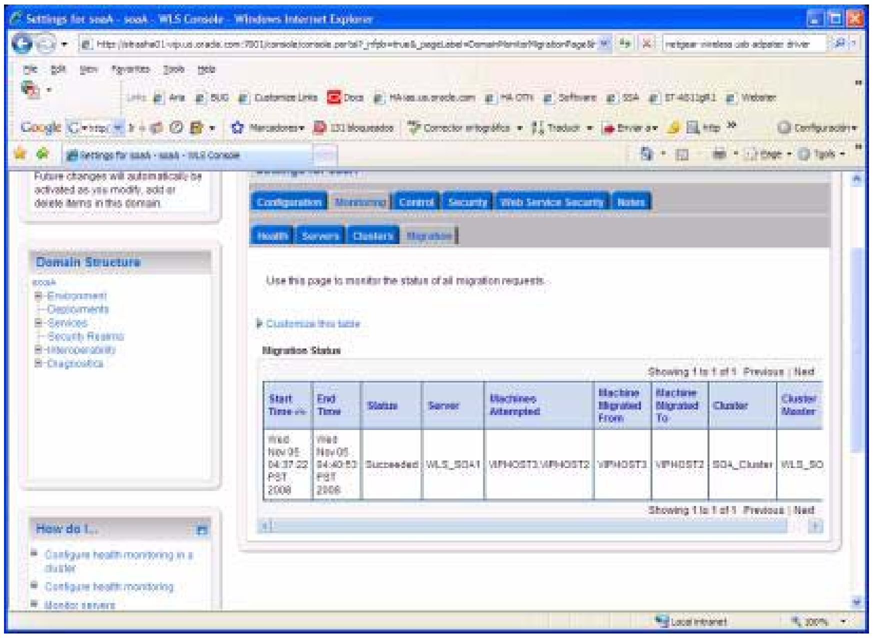
1. Watch the local Node Manager console. After 30 seconds since the last try to restart WLS_SOA1on Node 1, Node Manager on Node 2 should prompt that the floating IP for WLS_SOA1 is being brought up and that the server is being restarted in this node.
2. Access the soa-Service Infrastructure console in the same IP.

Migration can also be verified in Oracle WebLogic Server Administration Console:

1. Log into Oracle WebLogic Server Administration Console.
2. Click on **Domain** on the left pane.
3. Click on the **Monitoring** tab and then on the **Migration** sub-tab.

   The Migration Status table provides information on the status of the migration.

Figure 5–33 Migration Status Screen in Oracle WebLogic Server Administration Console



Use this page to monitor the status of all migration requests.

Migration Status

Showing 1 to 1 of 1 Previous | Next

Start Time	End Time	Status	Server	Machines Attempted	Machine Migrated From	Machine Migrated To	Cluster	Cluster Master
Wed Nov 05 04:37:22 PST 2008	Wed Nov 05 04:40:53 PST 2008	Succeeded	WLS_SOA1	VIPHOST3,VIPHOST2	VIPHOST3	VIPHOST2	SOA_Cluster	WLS_SO

Showing 1 to 1 of 1 Previous | Next

Note: On Windows, when you manually shut down multiple servers at the same time, on the same machine, and then, on another machine, attempt to start one of the servers that was shut down, the IP bind may not work. This happens because the original machine still has claim to the IP address, even though `netsh` has reported that the IP address has been removed.

To resolve this, you must check the network configuration either by using the ipconfig utility or Windows Network Configuration. Either of these may show that one of the virtual/floating IP addresses is still configured even though the servers have been shut down. You can then use Windows Network Configuration to remove the IP address using the following procedure:

1. From Windows Control Panel, select **Network Connections**.
2. Select the appropriate network interface, right-click, and select **Properties**.
3. Select **Internet Protocol (TCP/IP)** and click the **Properties** button.
4. Select **Advanced**.
5. Select the appropriate IP address and click the **Remove** button.

5.11.16 Scaling the Topology

You can scale out or scale up the enterprise topology. When you "scale up" the topology, you add new managed servers to nodes that are already running one or more managed servers. When you "scale out" the topology, you add new managed servers to new nodes.

5.11.16.1 Scaling Up the Topology (Adding Managed Servers to Existing Nodes)

In this case, you already have a node that runs a managed server configured with SOA components. The node contains a Middleware home, an Oracle HOME (SOA) and a domain directory for existing managed servers.

You can use the existing installations (the Middleware home, and domain directories) for creating new WLS_SOA servers. There is no need to install SOA binaries in a new location, or to run pack and unpack.

Follow these steps for scaling up the topology:

1. Using the Administration Console, clone WLS_SOA1 into a new managed server. The source managed server to clone should be one that already exists on the node where you want to run the new managed server.

   To clone a managed server:

   a. Select **Environment** -> **Servers** from the Administration Console.

   b. Select the managed server that you want to clone (for example, WLS_SOA1).

   c. Select **Clone**.

   Name the new managed server WLS_SOA*n*, where *n* is a number to identify the new managed server.

   The rest of the steps assume that you are adding a new server to APPHOST1, which is already running WLS_SOA1.

2. For the listen address, assign the host name or IP to use for this new managed server. If you are planning to use server migration as recommended for this server, this should be the VIP (also called a floating IP) to enable it to move to another node. The VIP should be different from the one used by the managed server that is already running.

3. Create JMS Servers for SOA and UMS on the new managed server.

   a. Use the Oracle WebLogic Server Administration Console to create a new persistent store for the new SOAJMSServer and name it, for example, SOAJMSFileStore_N. Specify the path for the store. This should be a directory on shared storage, as recommended in section Section 5.11.1.3, "Shared Storage Prerequisites":

      Note: This directory must exist before the managed server is started or the start operation fails.

      ```
      ORACLE_BASE/admin/DOMAIN_NAME/cluster_name/jms/SOAJMSFileStore_N
      ```

   b. Create a new JMS Server for SOA, for example, SOAJMSServer_N. Use the SOAJMSFileStore_N for this JMSServer. Target the SOAJMSServer_N Server to the recently created Managed Server (WLS_SOAn).

   c. Create a new persistence store for the new UMSJMSServer, for example, UMSJMSFileStore_N Specify the path for the store. This should be a directory on shared storage as recommended in sectionSection 5.11.1.3, "Shared Storage Prerequisites":

      ```
      ORACLE_BASE/admin/domain_name/cluster_name/jms/UMSJMSFileStore_N.
      ```

Note: This directory must exist before the managed server is started or the start operation fails.

You can also assign SOAJMSFileStore_N as store for the new UMS JMS Servers. For the purpose of clarity and isolation, individual persistent stores are used in the following steps.

---

d. Create a new JMS Server for UMS, for example, UMSJMSServer_N. Use the UMSJMSFileStore_N for this JMSServer. Target the UMSJMSServer_N Server to the recently created Managed Server (WLS_SOAn).

e. Update the SubDeployment targets for the SOA JMS Module to include the recently created SOA JMS Server. To do this, expand the **Services** node and then expand the **Messaging** node. Choose **JMS Modules** from the Domain Structure window of the Oracle WebLogic Server Administration Console. The JMS Modules page appears. Click **SOAJMSModuleUDDs** (represented as a hyperlink in the **Names** column of the table). The Settings page for SOAJMSModuleUDDs appears. Click the **SubDeployments** tab. The SOAJMSSubDM subdeployment appears.

Note: This subdeployment module results from updating the JMS configuration for the first two servers (WLS_SOA1 and WLS_SOA2) with the Uniform Distributed Destination Script (soa-createUDD.py) which is required for the initial HA topology set up.

Click the **SOAJMSSubDM** subdeployment. Add the new JMS Server for SOA called SOAJMSServer_N to this subdeployment. Click **Save**.

f. Target the UMSJMSSystemResource to the SOA_Cluster as it may have changed during extend operations. To do this, expand the **Services** node and then expand the **Messaging** node. Choose JMS Modules from the Domain Structure window of the Oracle WebLogic Server Administration Console. The JMS Modules page appears. Click **UMSJMSSytemResource** and click the **Targets** Tab. Make sure all of the servers in the SOA_Cluster appear selected (including the recently cloned WLS_SOAn).

g. Update the SubDeployment Targets for UMSJMSSystemResource to include the recently created UMS JMS Server. To do this, expand the **Services** node and then expand the **Messaging** node. Choose **JMS Module**s from the Domain Structure window of the Oracle WebLogic Server Administration Console. The JMS Modules page appears. Click **UMSJMSSystemResource** (represented as a hyperlink in the Names column of the table). The Settings page for UMSJMSSystemResource appears. Click the **SubDeployments** tab. The UMSJMSSubDMSOA subdeployment appears.

Note: This subdeployment module results from updating the JMS configuration for the first two servers (WLS_SOA1 and WLS_SOA2) with the Uniform Distributed Destination Script (soa-createUDD.py) which is required for the initial HA topology set up.

Click the **SOAJMSSubDM** subdeployment. Add the new JMS Server for UMS called UMSJMSServer_N to this subdeployment. Click **Save**.

h.

4. Configure TX persistent store for the new server. This should be a location visible from other nodes as indicated in the recommendations about shared storage for the EDG.

   From the Administration Console, select Server_name > Services tab. Under Default Store, in Directory, enter the path to the folder where you want the default persistent store to store its data files.

5. Start and test the new managed server from the Administration Console.

   a. Shut down the existing managed servers in the cluster.

   b. Ensure that the newly created managed server, WLS_SOAn, is up.

   c. Access the application on the newly created managed server (http://vip:port/soa-infra). The application should be functional.

6. Configure Server Migration for the new managed server.

   Note: Since this is a scale-up operation, the node should already contain a Node Manager and environment configured for server migration. The floating IP for the new SOA managed server should also be already present.

   Configure server migration following these steps:

   a. Log into the Administration Console.

   b. In the left pane, expand Environment and select Servers.

   c. Select the name of the new managed server for which you want to configure migration.

   d. Click the Migration tab.

   e. In the Available field, in the "Migration Configuration" section, select the machines to which to allow migration and click the right arrow. Select the same migration targets as for the servers that already exist on the node.

      For example, for new managed servers on APPHOST1, which is already running WLS_SOA1, select APPHOST2. For new managed servers on APPHOST2, which is already running WLS_SOA2, select APPHOST1.

      Make sure the appropriate resources are available to run the managed servers concurrently during migration.

   f. Select the "Automatic Server Migration Enabled" option. This enables the Node Manager to start a failed server on the target node automatically.

   g. Click Save.

   h. Restart the Administration Server, managed servers, and Node Manager.

7. Test server migration for this new server. Follow these steps from the node where you added the new server:

   a. Stop the WLS_SOAn managed server.

      To do this, run "kill -9 <pid>" on the PID of the managed server. You can identify the PID of the node using "ps -ef | grep WLS_SOAn".

   b. Watch the Node Manager Console: you should see a message indicating that WLS_SOA1's floating IP has been disabled.

c. Wait for the Node Manager to try a second restart of WLS_SOAn. Node Manager waits for a fence period of 30 seconds before trying this restart.

d. Once Node Manager restarts the server, stop it again. Now Node Manager should log a message indicating that the server will not be restarted again locally.

5.11.16.2 Scaling Out the Topology (Adding Managed Servers to New Nodes)

When you scale out the topology, you add new managed servers configured with SOA to new nodes.

Before performing the steps in this section, check that you meet these requirements:

- There must be existing nodes running managed servers configured with SOA within the topology.
- The new node can access the existing home directories for WebLogic Server and SOA. (Use the existing installations in shared storage for creating a new WLS_SOA or WLS_WSM managed server. You do not need to install WebLogic Server or SOA binaries in a new location but you do need to run pack and unpack to bootstrap the domain configuration in the new node.)

Note: If there is no existing installation in shared storage, installing WebLogic Server and SOA in the new nodes is required as described in Section 5.11, "Configuring High Availability for Oracle SOA Service Infrastructure and Component Service Engines."

Note: When an ORACLE_HOME or WL_HOME is shared by multiple servers in different nodes, Oracle recommends keeping the Oracle Inventory and Middleware home list in those nodes updated for consistency in the installations and application of patches. To update the oraInventory in a node and "attach" an installation in a shared storage to it, use *ORACLE_HOME*/oui/bin/attachHome.sh. To update the Middleware home list to add or remove a WL_HOME, edit the *user_home*/bea/beahomelist file. See the following steps.

Follow these steps for scaling out the topology:

1. On the new node, mount the existing Middleware home, which should include the SOA installation and the domain directory, and ensure that the new node has access to this directory, just like the rest of the nodes in the domain.
2. To attach ORACLE_HOME in shared storage to the local Oracle Inventory, execute the following command:

   ```
   APPHOSTn>cd ORACLE_BASE/product/fmw/soa/
   APPHOSTn>./attachHome.sh -jreLoc ORACLE_BASE/fmw/jrockit_160_05_R27.6.2-20
   ```

   To update the Middleware home list, create (or edit, if another WebLogic installation exists in the node) the *MW_HOME*/bea/beahomelist file and add *ORACLE_BASE*/product/fmw to it.
3. Log in to the Oracle WebLogic Administration Console.
4. Create a new machine for the new node that will be used, and add the machine to the domain.

5. Update the machine's Node Manager's address to map the IP of the node that is being used for scale out.

6. Use the Oracle WebLogic Server Administration Console to clone WLS_SOA1 into a new managed server. Name it WLS_SOAn, where n is a number.

   Note: These steps assume that you are adding a new server to node *n*, where no managed server was running previously.

7. Assign the host name or IP to use for the new managed server for the listen address of the managed server.

   If you are planning to use server migration for this server (which Oracle recommends) this should be the VIP (also called a floating IP) for the server. This VIP should be different from the one used for the existing managed server.

8. Create JMS servers for SOA and UMS on the new managed server.

   Note: These steps are not required for scaling out the WSM_PM managed server, only for WLS_SOA managed servers. They are not required either to scale up the BAM Web Applications system.

   To create JMS servers for SOA and UMS

   a. Use the Oracle WebLogic Server Administration Console to create a new persistent store for the new SOAJMSServer (which is created in a later step) and name it, for example, SOAJMSFileStore_N. Specify the path for the store. This should be a directory on shared storage as recommended in Section 5.11.1.3, "Shared Storage Prerequisites."

      Note: This directory must exist before the managed server is started or the start operation fails.

   b. Create a new JMS Server for SOA, for example, SOAJMSServer_N. Use the SOAJMSFileStore_N for this JMSServer. Target the SOAJMSServer_N Server to the recently created managed server (WLS_SOAn).

   c. Create a new persistence store for the new UMSJMSServer, and name it, for example, UMSJMSFileStore_N. Specify the path for the store . This should be a directory on shared storage as recommended in Section 5.11.1.3, "Shared Storage Prerequisites."

      ```
      ORACLE_BASE/admin/domain_name/cluster_name/jms/UMSJMSFileStore _N
      ```

      Note: This directory must exist before the managed server is started or the start operation fails.

      Note: It is also possible to assign SOAJMSFileStore_N as the store for the new UMS JMS Servers. For the purpose of clarity and isolation, individual persistent stores are used in the following steps.

d. Create a new JMS Server for UMS: for example, UMSJMSServer_N. Use the UMSJMSFileStore_N for this JMS Server. Target the UMSJMSServer_N Server to the recently created managed server (WLS_SOAn).

e. Update the SubDeployment targets for the SOA JMS Module to include the recently created SOA JMS Server. To do this, expand the **Services** node and then expand the **Messaging** node. Choose JMS Modules from the Domain Structure window of the Oracle WebLogic Server Administration Console. The JMS Modules page appears. Click **SOAJMSModuleUDDs** (represented as a hyperlink in the Names column of the table). The Settings page for SOAJMSModuleUDDs appears. Open the **SubDeployments** tab. The SOAJMSSubDM subdeployment appears.

   Note: This subdeployment module results from updating the JMS configuration for the first two servers (WLS_SOA1 and WLS_SOA2) with the Uniform Distributed Destination Script (soa-createUDD.py), which is required for the initial high availability topology setup.

   Click the **SOAJMSSubDM** subdeployment. Add the new JMS Server for SOA called SOAJMSServer_N to this subdeployment. Click **Save**.

f. Target the UMSJMSSystemResource to the SOA_Cluster as it may have changed during extend operations. To do this, expand the **Services** node and then expand the **Messaging** node. Choose **JMS Modules** from the Domain Structure window of the Oracle WebLogic Server Administration Console. The JMS Modules page appears. Click **UMSJMSSytemResource** and open the **Targets** tab. Make sure all of the servers in the SOA_Cluster appear selected (including the recently cloned WLS_SOAn).

g. Update the SubDeployment targets for UMSJMSSystemResource to include the recently created UMS JMS Server. To do this, expand the **Services** node and then expand the **Messaging** node. Choose JMS Modules from the Domain Structure window of the Oracle WebLogic Server Administration Console. The JMS Modules page appears. Click **UMSJMSSystemResource** (represented as a hyperlink in the Names column of the table). The Settings page for UMSJMSSystemResource appears. Open the **SubDeployments** tab. The UMSJMSSubDMSOA subdeployment appears

   Note: This subdeployment module results from updating the JMS configuration for the first two servers (WLS_SOA1 and WLS_SOA2) with the Uniform Distributed Destination Script (soa-createUDD.py), which is required for the initial HA topology setup.

   Click the **UMSJMSSubDMSOA** subdeployment. Add the new JMS Server for UMS called UMSJMSServer_N to this subdeployment. Click **Save**.

9. Run the pack command on APPHOST1 to create a template pack as follows:

```
APPHOST1> cd ORACLE_BASE/product/fmw/soa/common/bin

APPHOST1> ./pack.sh -managed=true -domain= ORACLE_BASE/product/fmw/user_
projects/domains/soadomain/
-template=soadomaintemplateScale.jar -template_name=soa_domain_templateScale
```

   Run the following command on APPHOST1 to copy the template file created to APPHOSTN:

```
APPHOST1> scp soadomaintemplateScale.jar oracle@APPHOSTN:/ ORACLE_
BASE/product/fmw/soa/common/bin
```

Run the unpack command on APPHOSTN to unpack the template in the managed server domain directory as follows:

```
APPHOSTN> cd ORACLE_BASE/product/fmw/soa/common/bin

APPHOSTN> ./unpack.sh -domain= ORACLE_BASE/product/fmw/user_
projects/domains/soadomain/ -template=soadomaintemplateScale.jar
```

10. Start the Node Manager on the new node. To start the Node Manager, use the installation in shared storage from the existing nodes, and start Node Manager by passing the host name of the new node as a parameter as follows:

    ```
    APPHOSTN> ORACLE_BASE/product/fmw/wlserver_10.3/server/bin/startNodeManager
    <new_node_ip>
    ```

11. Disable host name verification for the new managed server. Before starting and verifying the WLS_SOAN managed server, you must disable host name verification. You can re-enable it after you have configured server certificates for the communication between the Oracle WebLogic Administration Server and the Node Manager in APPHOSTn.

    To disable host name verification:

    a. In the Oracle Enterprise Manager Console, select **Oracle WebLogic Server Administration Console**.
    b. Expand the **Environment** node in the Domain Structure window.
    c. Click **Servers**. The Summary of Servers page appears.
    d. Select **WLS_SOAn** in the Names column of the table.

       The Settings page for server appears.
    e. Click the **SSL** tab.
    f. Click **Advanced**.
    g. Set Hostname Verification to **None**.
    h. Click **Save**.

12. Start and test the new managed server from the Oracle WebLogic Server Administration Console:

    a. Shut down all the existing managed servers in the cluster.
    b. Ensure that the newly created managed server, WLS_SOAn, is running.
    c. Access the application on the newly created managed server (http://vip:port/soa-infra). The application should be functional.

13. Configure the appropriate coherence "well known addresses list" as start parameters for the server as described in Section 5.11, "Configuring High Availability for Oracle SOA Service Infrastructure and Component Service Engines."

14. Configure server migration for the new managed server.

Note: Since this new node is using an existing shared storage installation the node is already using a Node Manager and an environment configured for server migration that includes netmask, interface, wlsifconfig script superuser privileges. The floating IP for the new SOA Managed Server is already present in the new node.

To log into the Administration Console and configure server migration:

a. In the left pane, expand **Environment** and select **Servers**.

b. Select the server (represented as hyperlink) for which you want to configure migration from the Names column of the table. The Setting page for that server appears.

c. Click the **Migration** tab.

d. In the Available field, in the Migration Configuration section, select the machines to which to allow migration and click the right arrow.

Note: Specify the least-loaded machine as the migration target for the new server. The required capacity planning must be completed so that this node has enough available resources to sustain an additional managed server.

---

e. Select the **Automatic Server Migration Enabled** option. This enables the Node Manager to start a failed server on the target node automatically.

f. Click **Save**.

g. Restart the Administration Server, managed servers, and Node Manager.

5.12 Configuring High Availability for Oracle BAM

This section describes how to set up a 2 node Highly Available configuration for BAM. This includes the BAM Server and the BAM Web Applications. The architecture targeted for the configuration steps is as follows:

Note: Oracle strongly recommends reading the release notes for any additional installation and deployment considerations prior to starting the setup process.

Figure 5–34 represents the example architecture that configuration steps in this section create.

Figure 5–34 Oracle SOA BAM High Availability Architecture

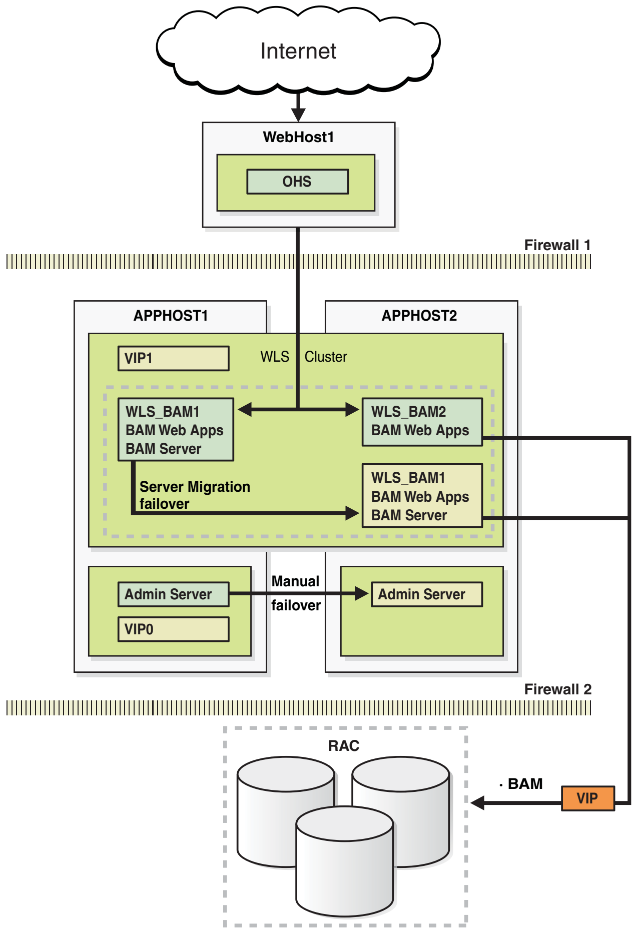


Figure 5–34 illustrates a two-node cluster running Oracle BAM Web Applications. One of them is also running Oracle BAM Server. The WebLogic Servers are front ended by Oracle HTTP Server which loads balances incoming requests to BAM Web Applications. An Oracle RAC database is used for storing metadata and BAM schemas.

5.12.1 Preparing the Environment: Prerequisite Steps Before Setting up a High Availability Configuration for Oracle BAM

The following sections provide prerequisite steps before setting up an Oracle SOA Service Infrastructure high availability configuration with Oracle BAM:

5.12.1.1 Database Prerequisites

Oracle SOA Service Infrastructure supports the following database versions:

- Oracle Database 10g Release 2 (10.2.x)
- Oracle Database 11g Release 1 (11.1.x)

To determine the database version, execute this query:

```
SQL>select version from sys.product_component_version where
product like 'Oracle%';
```

5.12.1.2 VIP and IPs Prerequisites

As shown in Table 5–7, you configure the Administration and the BAM Server in APPHOST1 to listen on two different virtual IPs, as shown in the architecture diagram. This requires the provisioning of the corresponding VIP in the box and related host names in the DNS system in your network. Make sure that the different VIPS are available and are reachable before running the installation.

Table 5–7 Virtual Hosts

Virtual IP	VIP Maps to...	Description
VIP0	APPHOST1VHN0	APPHOST1VHN0 is the virtual host name that is the listen address for the Administration Server and fails over with manual failover of the Administration Server. It is enabled on the node where the Admin Server process is running (APPHOST1 by default).
VIP1	APPHOST1VHN1	APPHOST1VHN1 is the virtual host name that maps to the listen address for WLS_BAM1 and fails over with server migration of this managed server. It is enabled on the node where WLS_BAM1 process is running (APPHOST1 by default).

5.12.1.3 Installing and Configuring the Database Repository

This section describes how to install and configure the database repository.

Oracle Clusterware

- For 10*g* Release 2 (10.2), see the *Oracle Database Oracle Clusterware and Oracle Real Application Clusters Installation Guide*.
- For 11*g* Release 1 (11.1), see *Oracle Clusterware Installation Guide*.

Automatic Storage Management

- For 10*g* Release 2 (10.2), see *Oracle Database Oracle Clusterware and Oracle Real Application Clusters Installation Guide*.
- For 11*g* Release 1 (11.1), see *Oracle Clusterware Installation Guide*.
- When you run the installer, select the **Configure Automatic Storage Management** option in the **Select Configuration** page to create a separate Automatic Storage Management home.

Oracle Real Application Clusters

- For 10*g* Release 2 (10.2), see *Oracle Database Oracle Clusterware and Oracle Real Application Clusters Installation Guide*.
- For 11*g* Release 1 (11.1), see *Oracle Real Application Clusters Installation Guide*.

Database Initialization Parameters

Ensure that the following initialization parameter is set to the required value. It is checked by Repository Creation Assistant.

Table 5–8 Required Initialization Parameters

Parameter	Required Value	Parameter Class
`PROCESSES`	300 or greater	Static

To check the value of the initialization parameter using SQL*Plus, you can use the SHOW PARAMETER command.

As the SYS user, issue the SHOW PARAMETER command as follows:

```
SQL> SHOW PARAMETER processes
```

Set the initialization parameter using the following command:

```
SQL> ALTER SYSTEM SET processes=300 SCOPE=SPFILE
```

Restart the database.

Note: The method that you use to change a parameter's value depends on whether the parameter is static or dynamic, and on whether your database uses a parameter file or a server parameter file. See the *Oracle Database Administrator's Guide* for details on parameter files, server parameter files, and how to change parameter values.

5.12.1.4 Using Oracle Fusion Middleware Repository Creation Utility to Load Oracle Fusion Middleware schemas

Install the 11g (11.1.1) Oracle Fusion Middleware Metadata Store and SOA schemas into a Real Application Cluster database before you install Oracle Fusion Middleware SOA components. Oracle Fusion Middleware provides a tool, Oracle Fusion Middleware Repository Creation Utility (RCU), to create the component schemas in an existing database. See Oracle Fusion Middleware Installation Concepts and Repository Creation Utility User's Guide for more information about installing RCU.

5.12.1.4.1 Running RCU Run RCU to install the required metadata for Oracle Fusion Middleware 11*g*.

1. Start up RCU using the following command:

   ```
   RCU_HOME/bin/rcu &
   ```

2. In the Welcome screen, click **Next**.
3. In the Create Repository screen, select **Create** to load component schemas into a database, click Next.
4. In the Database Connection Details screen, enter connection information for your database:
   - Database Type: Select **Oracle Database**.
   - Host Name: Enter the name of the node that is running the database. For an Oracle RAC database, specify the VIP name, or one of the node names as the host name: **SOADBHOST1VIRTUAL**.
   - Port: The port number for the database: **1521**
   - Service Name: Enter the service name of the database: **soaha.mycompany.com**
   - Username: **SYS**
   - Password: Enter the password for the SYS user.
   - Role: **SYSDBA**
5. Click **Next**.
6. If you receive the following warning message:

The database you are connecting is with non-UTF8 charset, if you are going to use this DB for multilingual support, you may have data loss. If you are not using for multilingual support you can continue, otherwise we strongly recommend using UTF-8 database.

7. Click **Ignore** or **Stop**.
8. In the Select Components screen, select **Create a New Prefix**, and enter a prefix to use for the database schemas, for example, **SOAHA**

   Write down the schema names so they are available in later procedures.

   - Select the following schemas:
     - AS Common Schemas:

       Metadata Services
     - SOA Infrastructure:

       Business Activity Monitoring

       User Messaging
9. Click **Next**.
10. In the Schema Passwords screen, enter passwords for the main and additional (auxiliary) schema users, and click **Next**.
11. In the Map Tablespaces screen, choose the tablespaces for the selected components, and click **Next**.
12. In the Summary screen, click **Create**.
13. In the Completion Summary screen, click **Close**.

See *Oracle Fusion Middleware Installation Concepts and Repository Creation Utility User's Guide* for more information about using RCU.

5.12.2 Installing Oracle HTTP Server on WebHost1

To install Oracle HTTP Server on WEBHOST1:

1. Verify that the servers meet the following requirements:
   - The system, patch, kernel, and other requirements meet the requirements specified in the *Oracle Fusion Middleware Installation Guide for Oracle SOA Suite*.
   - Port 7777 is not used by any service on the nodes. You can verify this by running the following command:

     Unix:

     ```
     netstat -an | grep 7777
     ```

     Windows:

     ```
     netstat -an | findstr 7777
     ```

   If the ports are in use, make them available.
2. On UNIX platforms, if the `/etc/oraInst.loc` or `/var/opt/oracle/oraInst.loc` file exists, check that its contents are correct. Specifically, check that the inventory directory is correct, and that you have write permissions for that directory.

   If the `/etc/oraInst.loc` file does not exist, skip this step.

3. Start Oracle Universal Installer for Oracle Fusion Middleware 11*g* Web Tier Utilities CD installation as follows:

   For UNIX, run this command: `runInstaller`.

   For Windows, double-click **setup.exe**.

4. In the Welcome screen, click **Next**.
5. In the Select Installation Type screen, select **Install and Configure**, and click **Next**.
6. In the Prerequisite Checks screen, ensure that all the prerequisites are met, and click **Next**.
7. In the Specify Installation Location screen, set the location to:

   `/u01/app/oracle/product/11.1.1/ohs_1`

8. Click **Next**.
9. In the Configure Components screen:
   - Select **Oracle HTTP Server**.
   - Do not select **Associate Selected Components with WebLogic Domain**.
   - Click **Next**.
10. In the Specify Component Details screen, enter the following values:
    - Instance Home Location: **corrected value. See bug 8199458**
    - Instance Home Location: **/u01/app/oracle/product/11.1.1/ohs_1/instances/ohs_instance1**
    - Instance Name: **ohs_instance1**
    - OHS Component Name: **ohs1**
11. Click **Next**.
12. In the Configure Ports screen, do the following:
    - Select **Specify Ports** using Configuration File and copy the `staticports.ini` template file from your installation disk (the file is located in the /Disk1/stage/Response directory) to your user's home. Then use the **Browse** button to select this file.
    - Click **View/Edit File** to open the `staticports.ini` file in an editor.
    - Change the Oracle HTTP Server port in that file to **7777**.
    - Save the file.
13. Click **Next**.
14. In the Configuration Summary screen, ensure that the selections are correct, and click **Install**.
15. In the Installation Progress screen:

    For UNIX systems, a dialog box appears prompting you to run the `oracleRoot.sh` script. Open a command window and run the script, following the prompts.

    Click **Next**.

16. In the Configuration screen, several configuration assistants are launched in succession. When the configuration assistants are finished, the Configuration Completed screen appears.

17. In the Configuration Completed screen, click **Finish** to exit.

5.12.2.1 Validating Oracle HTTP Server

To verify that Oracle HTTP Server is set up correctly, access the root URL context of the server by using the following URL a browser:

HTTP://WebHost1:7777/

If Oracle HTTP Server is set up correctly, the **Hello World** page appears in the browser.

5.12.3 Installing Oracle Fusion Middleware Home

This section describes how to install Oracle Fusion Middleware on all nodes in the application tier that runs Oracle WebLogic Server and Oracle Fusion Middleware SOA Suite.

Install the following Oracle Fusion Middleware components:

- Oracle WebLogic Server (see Section 5.12.3.1, "Installing Oracle WebLogic Server.")
- Oracle Fusion Middleware SOA Suite (see Section 5.12.3.2, "Installing Oracle Fusion Middleware for Oracle SOA")

5.12.3.1 Installing Oracle WebLogic Server

To install Oracle WebLogic Server on all nodes in the application tier:

1. On UNIX platforms, if the `/etc/oraInst.loc` or `/var/opt/oracle/oraInst.loc` file exists, check that its contents are correct. Specifically, check that the inventory directory is correct and that you have write permissions for that directory.

   If the `/etc/oraInst.loc` file does not exist, skip this step.

2. Start Oracle WebLogic Server Installer

   On UNIX (Linux in the following example):

   ```
   APPHOST1> server103_linux32.bin
   ```

   On Windows:

   ```
   APPHOST1> server103_win32.exe
   ```

3. In the Welcome screen, click **Next**.
4. In the Choose Middleware Home Directory screen:
   - Select **Create a New Middleware Home**
   - For the **Middleware Home Directory** field, enter **ORACLE_BASE/product/fmw**
   - Click **Next**.
5. In the Register for Security Updates screen, enter your contact information for security update notifications, and click **Next**.
6. In the Choose Install Type screen, select **Custom**, and click **Next**.
7. In the Choose Products and Components screen, click **Next**.
8. In the JDK Selection screen, select only **jrockit_160_05_R27.6.2-20**, and click **Next**.

9. In the Choose Product Installation Directories screen, accept the following directory:

   ```
   ORACLE_BASE/product/fmw/wlserver_10.3
   ```

   Click **Next**.

10. In the Installation Summary screen, click **Next**.

11. In the Installation Complete screen, deselect **Run QuickStart**, and click **Done**.

5.12.3.2 Installing Oracle Fusion Middleware for Oracle SOA

To install Oracle Fusion Middleware for Oracle SOA on all the nodes in the application tier:

1. Start Oracle Fusion Middleware 11*g* Oracle SOA Suite Installer:

   On UNIX:

   ```
   APPHOST1> runInstaller
   ```

   On Windows:

   ```
   APPHOST1> setup.exe
   ```

   When Oracle Fusion Middleware 11g Oracle SOA Suite Installer prompts you for a **JRE/JDK location** enter Oracle SDK location created in Oracle WebLogic Server installation, for example, **ORACLE_BASE/product/fmw/jrockit_160_05**.

2. In the Welcome screen, click **Next**.

3. In the Prerequisite Checks screen, verify that the checks complete successfully, and click **Next**.

4. In the Specify Installation Location screen:

   - For Middleware Home, enter **ORACLE_BASE/product/fmw**
   - For Oracle Home Directory, enter **soa**

   Click **Next**.

5. In the Installation Summary screen, click **Install**.

6. In the Installation Complete screen, click **Finish**.

5.12.4 Enabling VIP0 and VIP1 on APPHOST1

For information about configuring virtual IPs for the administration server and configuring the administration server for high availability, see Section 10.2.2.3, "Transforming the Administration Server for Cold Failover Clusters."

5.12.5 Running Oracle Fusion Middleware Configuration Wizard on APPHOST1 to Create the WebLogic Server Oracle BAM Domain

Run Oracle Fusion Middleware Configuration Wizard from the SOA home directory to create a domain containing the Administration Server and Oracle SOA components. Ensure that the database where you installed the repository is running. For RAC databases, all the instances must be running.

1. Change the directory to the location of Oracle WebLogic Server Configuration Wizard, located in the SOA home directory:

   ```
   APPHOST1> cd ORACLE_HOME/common/bin
   ```

2. Start Oracle WebLogic Server Configuration Wizard

   For UNIX:

   ```
   APPHOST1> ./config.sh
   ```

   For Windows:

   ```
   APPHOST1> config.cmd
   ```

3. In the Welcome screen, select **Create a New WebLogic Domain**, and click **Next**.
4. In the Select Domain Source screen, select **Generate a domain configured automatically to support the following products**, and select the following products:
   - Oracle Business Activity Monitoring
   - Oracle WSM-PM (selected by default)
   - Enterprise Manager
   - JRF (selected by default)

   Click **Next**.
5. In the Specify Domain Name and Location screen, make the following entries:
   - Domain Name: **bamdomain**
   - Domain Location: accept the default
   - Application Location: accept the default

   Click **Next**.
6. In the Configure Administrator Username and Password screen, enter the username and password to be used for the domain's administrator, and click **Next**.
7. In the Configure Server Start Mode and JDK screen, make the following selections:
   - WebLogic Domain Startup Mode: select **Production Mode**
   - JDK Selection: select **Oracle SDK1.6.0_05**.

   Click **Next**.
8. In the Configure JDBC Component Schema screen:
   a. Ensure that the following data source appears on the screen. The user names shown in Table 5–9, assume that **soaha** was used as the prefix for schema creation from RCU.

      Table 5–9 Configuring Values for Data Sources

Data Source	User Name
BAMDataSource	soaha_orabam
OraSDPMDataSource	soaha_orasdpm
mds-owsm	soaha_mds

   b. Select **Configure selected component schemas as RAC multi data source schemas in the next panel**.

   c. Click **Next**.

9. In the Configure RAC Multi Data Sources screen enter the following:

   a. With all schemas selected enter values for the following fields, specifying the connect information for the Oracle RAC database that was seeded with RCU.

      Driver: **Oracle driver (Thin) for RAC Service-Instance connections, Versions:10, 11**.

      Service Name: Enter the service name of the database, for example, **soaha.mycompany.com**.

      Username: Enter the complete user name (including the prefix) for the schemas.

      Password: Enter the password to use to access the schemas.

   b. Enter the host name, instance name, and port.

   c. Click **Add**.

   d. Repeat for each Oracle RAC instance.

   e. Select the **BAM Schema Only** and enter the User Name and Password (soaha_orabam/passwd).

   f. Select the **User Messaging Service Schema Only** and enter the User Name and Password (soaha_orasdpm/passwd)

   g. Select the **OWSM MDS Schema Only** and enter the User Name and Password (soaha_mds/passwd)

   h. Click **Next**.

10. In the Test JDBC Data Sources screen, the connections are tested automatically. The Status column displays the results. Ensure that all connections were successful. If not, click **Previous** to return to the previous screen and correct your entries.

    Click **Next** when all the connections are successful.

11. In the Select Optional Configuration screen, select **Administration Server**, **Managed Servers**, **Cluster and Machines** and **Deployment and services**, and click **Next**.

12. In the Configure the Administration Server screen, enter the following values:

    - Name: **AdminServer**
    - Listen Address: **VIP0**
    - Listen Port: **7001**
    - SSL listen port: N/A
    - SSL enabled: leave unchecked

    Click **Next**.

13. In the Configure Managed Servers screen, add the following managed servers:

Table 5–10 Configuring Managed Servers

Name	Listen Address	Listen Port	SSL Listen Port	SSL Enabled
WLS_BAM1	VIP1 (Virtual IP that is migrated by server migration)	8001	n/a	No
WLS_BAM2	APPHOST2 (BAM Server WLS_BAM2 does not use serer migration)	8001	n/a	No

Note: Although the standard recommendation is to run custom applications and other systems in a separate WebLogic Managed Server, the creation of custom WLS managed servers described in Figure 5–34 is not addressed here.

Click **Next**.

14. In the Configure Clusters screen, add the following cluster:
    - Name: **BAM_Cluster**
    - Cluster Messaging Mode: **unicast**
    - Multicast Address: N/A
    - Multicast Port: N/A
    - Cluster Address: Leave empty

    Click **Next**.

15. In the Assign Servers to Clusters screen, assign the following servers to BAM_Cluster:
    - WLS_BAM1
    - WLS_BAM2

    Click **Next**.

16. In the Configure Machines screen:
    - Delete the **LocalMachine** that appears by default.
    - Click the **Unix Machine** tab, and add the following machines:

    Table 5–11 Configuring Machines

Name	Node Manager Listen Address
APPHOST1	APPHOST1
APPHOST2	APPHOST2

    Leave all other fields to their default values, and click **Next**.

17. In the Assign Servers to Machines screen, assign servers to machines as follows:
    - APPHOST1: **AdminServer**, **WLS_BAM1**
    - APPHOST2: **WLS_BAM2**

Click **Next**.

18. In the Target Deployments to Cluster or Services screen, ensure the following targets:
    - User Messaging Deployments should be targeted only to BAM_Cluster. (The usermessaging-xmpp, usermessaging-smpp, and usermessaging-voicexml applications are optional.)
    - wsm-pm should be targeted only to BAM_Cluster.
    - The DMS Application should be targeted to BAM_Cluster and Admin Server.
    - The oracle.rules.*, oracle.sdp.* and oracle.bam.* deployments should be targeted only to BAM_Cluster.
    - The oracle.wsm.seedpolicies library should be targeted only to the BAM_Cluster.
19. In the Target Services to Cluster or Servers screen, ensure the following targets:
    - WSM Startup Class should be targeted only to BAM_Cluster.
    - mds-wsm, mds-wsm-rac0, and mds-wsm-rac1 should be targeted to both BAM_Cluster and AdminServer.
    - OrasDPMDatasource, OrasDPMDatasource-rac0, and OrasDPMDatasource-rac1 should be targeted to the BAM_Cluster.
    - OWSM Startup Class is only targeted to BAM_Cluster *DMS Startup is targeted both to BAM_Cluster and AdminServer*mds-wsm, mds-wsm-rac0, and mds-wsm-rac1 should be targeted to both BAM_Cluster and AdminServer
    - mds-soa, mds-soa-rac0, and mds-soa-rac1 should be targeted to both BAM_Cluster and AdminServer.
20. In the Configuration Summary screen, click **Create**.
21. In the Creating Domain screen, click **Done**.

5.12.6 Configuring the Cache Size for BAM JDBC Data Sources

The statement cache size parameter must be set to 0 for a BAM JDBC data source. While this is the default setting when the Configuration Wizard is run against a single instance of a database, the statement cache size is not set for the BAM JDBC data sources when the Configuration Wizard is run against a RAC database. In this case, you must configure the statement cache size using the Oracle WebLogic Server Administration Console.

Note: To make BAM data sources available in the Oracle WebLogic Server Administration Consoles, you must restart the Administration Server to commit changes made using the Configuration Wizard.

To configure the statement cache size:

1. Expand the **Services** node in the Domain Structure window.
2. Expand the **JDBC** node and then select **Data Sources**. The Summary of JDBC Data Sources page appears.

3. Click **BAMDataSource-rac0** in the Names column of the table. The Settings page appears.
4. Click the **Connection Pool** tab.
5. Enter 0 in the Statement Cache Size field.
6. Click **Save**.
7. Repeat these steps for **BAMDataSource-rac1**.
8. Save and Activate the changes.

5.12.7 Creating boot.properties for the Administration Server and for WLS_BAM1 on APPHOST1

This is an optional step for enabling the Administration Server to start up without prompting you for the administrator username and password. Create a boot.properties file for the administration server on APPHOST1.

For the administration server:

1. Create the following directories:

```
mkdir ORACLE_BASE/product/fmw/user_projects/domains/bamdomain/servers/

mkdir ORACLE_BASE/product/fmw/user_
projects/domains/bamdomain/servers/AdminServer

mkdir ORACLE_BASE/product/fmw/user_
projects/domains/bamdomain/servers/AdminServer/security
```

2. Use a text editor to create a file called `boot.properties` in the directory created in the previous step, and enter the following lines in the file:

```
username=adminuser
password=password
```

Note: When you start up the Administration Server, the username and password entries in the file are encrypted.

For security reasons, minimize the time the entries in the file are left unencrypted. After you edit the file, start up the server as soon as possible in order for the entries to be encrypted.

5.12.8 Starting the Administration Server on APPHOST1

To start the Administration Server on APPHOST1 run the following commands:

For Linux:

```
APPHOST1> cd ORACLE_BASE/product/fmw/user_projects/domains/bamdomain/bin

APPHOST1> ./startWebLogic.sh
```

For Windows:

```
startWebLogic.cmd
```

To verify that the administration server is properly configured:

1. In a browser, go to `http://VIP0:7001/console`.

2. Log in as the administrator.
3. Verify that the WLS_BAM1 and WLS_BAM2 managed servers are listed.
4. Verify that the BAM_Cluster cluster is listed.
5. Verify that you can access Enterprise Manager at http://VIP0:7001/em.

5.12.9 Disabling Host Name Verification for the Servers

This step is required if you have not set up the appropriate certificates to authenticate the different nodes with the administration server. If you have not configured the server certificates, you receive errors when managing the different WebLogic Servers. To avoid these errors, disable host name verification while setting up and validating the topology, and enable them again once the high availability topology configuration is complete.

To disable host name verification:

1. In Oracle WebLogic Server Administration Console, select **Administration Server**, **SSL**, and then **Advanced**.
2. Set **Hostname Verification** to **None**.
3. In Oracle WebLogic Server Administration Console, select **WLS_BAM1**, **SSL**, and then **Advanced**.
4. Set **Hostname Verification** to **None**.
5. In Oracle WebLogic Server Administration Console, select **WLS_BAM2**, **SSL**, and then **Advanced**.
6. Set **Hostname Verification** to **None**.
7. Restart the administration server for the changes to take affect.
8. Save and activate the changes.

5.12.10 Configuring the Cache Size for BAM JDBC Data Sources

The statement cache size parameter must be set to 0 for a BAM JDBC data source. While this is the default setting when the Configuration Wizard is run against a single instance of a database, the statement cache size is not set for the BAM JDBC data sources when the Configuration Wizard is run against a RAC database. In this case, you must configure the statement cache size using the Oracle WebLogic Server Administration Console.

Note: To make Oracle BAM data sources available in the Oracle WebLogic Server Administration Consoles, you must restart the administration server to commit changes made using the Configuration Wizard.

To configure the statement cache size:

1. Expand the **Services** node in the Domain Structure window.
2. Expand the JDBC node and then select **Data Sources**.

   The Summary of JDBC Data Sources page appears.
3. Click **BAMDataSource-rac0** in the Names column of the table.

   The Settings page appears.

4. Click the **Connection Pool** tab.
5. Enter **0** in the Statement Cache Size field.
6. Save and Activate the changes.
7. Repeat these steps for BAMDataSource-rac1.

5.12.11 Configuring a JMS Persistence Store for BAM UMS

Configure the location for all of the persistence stores as a directory that is visible from both APPHOST1 and APPHOST2. This is required in order to enable the resume of transactions when a server is migrated to a different node. By using a shared location for the persistent stores from both nodes, it is guaranteed that no messages are lost should a failover take place.

1. Log into the Oracle WebLogic Server Administration Console.
2. In the Domain Structure window, expand the **Services** node and then click the **Persistence Stores** node.

   The Summary of Persistence Stores page appears.
3. Select the a persistent store (represented as a hyperlink) from the Name column of the table.

   The Settings page for the persistence store appears.
4. In the Configuration tab, enter the location on a persistent storage solution (such as NAS or SAN) that is available to other servers in the cluster in the Directory field. Specifying this location enables pending JMS messages to be sent.
5. Click **Save**.
6. Repeat these steps for all persistent stores.
7. Save and Activate the changes.

5.12.12 Configuring a Default Persistence Store for Transaction Recovery

Each server has a transaction log that stores information about committed transactions that are coordinated by the server that may not have been completed. The WebLogic Server uses this transaction log for recovery from system crashes or network failures. To leverage the migration capability of the Transaction Recovery Service for the servers within a cluster, store the transaction log in a location accessible to a server and its backup servers.

Note: Preferably, this location should be a dual-ported SCSI disk or on a Storage Area Network (SAN).

To set the location for the default persistence store:

1. Log into the Oracle WebLogic Server Administration Console.
2. In the Domain Structure window, expand the **Environment** node and then click the **Servers** node.

   The Summary of Servers page appears.
3. Click the name of the server (represented as a hyperlink) in Name column of the table.

The settings page for the selected server appears and defaults to the Configuration tab.

4. Click the **Services** tab.
5. In the Default Store section of the page, enter the path to the folder where the default persistent stores will store its data files.
6. Save and Activate the changes.

Note: To enable migration of the Transaction Recovery Service, specify a location on a persistent storage solution that is available to other servers in the cluster. This directory must also exist before you restart the server.

5.12.13 Untargeting the BAM Server System from APPHOST2

Because the BAM server component in BAM is a singleton, you must untarget it from one of the WLS_BAM servers before you configure it for server migration. Otherwise the system would use two active BAM Servers which could cause different data inconsistencies. This ways BAM Web applications run in both APPHOST1 and APPHOST2 but BAM Server is initially active only in APPHOST1.

In this step, you remove the BAM server runtime from WLS_BAM2. To untarget the BAM server artifacts from WLS_BAM2:

1. Log into the Oracle WebLogic Administration Console at `http://APPHOST1VHN0:7001/console`.
2. In the Domain Structure window, choose **Environment** and then **Servers**.

   The Summary of Servers page appears.
3. Select **WLS_BAM2** in Name column of the table.

   The Settings page for WLS_BAM2 appears.
4. Click the **Deployments** tab.
5. Select the **oracle-bam** application from the Name column of the table.

   The Settings page for the oracle-bam application appears.
6. Click the **Targets** tab.
7. Click **Lock and Edit**.
8. Change the targets for the modules as described in Table 5–12.

   Save and Activate the changes

Note: You must target all of these components as described in Table 5–12, as incorrect targeting can prevent the BAM system from starting.

Table 5–12 oracle-bam Component Target Types

Component	Type	Target
oracle-bam(11.1.1)	Enterprise Application	BAM_Cluster
/oracle/bam	WEBAPP	WLS_BAM1
oracle-bam-adc-ejb.jar	EJB	WLS_BAM1
oracle-bam-ems-ejb.jar	EJB	WLS_BAM1
oracle-bam-eventengine-ejb.jar	EJB	WLS_BAM1
oracle-bam-reportcache-ejb.jar	EJB	WLS_BAM1
oracle-bam-statuslistener-ejb.jar	EJB	WLS_BAM1
OracleBAM	WEBAPP	BAM_Cluster
OracleBAMWS	WEBAPP	BAM_Cluster
sdpmessagingclient-ejb.jar	EJB	WLS_BAM1

5.12.14 Propagating the Domain Configuration from APPHOST1 with pack/unpack Utilities

Follow these steps to propagate the domain configuration to APPHOST1 using Pack/Unpack utilities:

1. Run the following pack command on APPHOST1 to create a template pack:

```
APPHOST1> cd ORACLE_HOME/common/bin
APPHOST1> ./pack.sh -managed=true
domain=ORACLE_BASE/product/fmw/user_projects/domains/bamdomain/
-template=bamdomaintemplate.jar
-template_name=bam_domain_template
```

2. Run the unpack command on APPHOST2 to unpack the propagated template:

```
APPHOST2> cd
 ORACLE_HOME/common/bin
APPHOST2> ./unpack.sh
 -domain=ORACLE_BASE/product/fmw/user_projects/domains/bamdomain/
 -template=bamdomaintemplate.jar
```

5.12.15 Starting Node Manager on APPHOST1 and APPHOST2

To start Node Manager on APPHOST1 and APPHOST2:

1. Run the `setNMProps.sh` script, located in the *ORACLE_HOME*/common/bin directory.

   This script sets the `StartScriptEnabled` property to **true** before starting Node Manager, and allows staring servers from the Administration Console (the environment variables required for the BAM Servers are set in the default start script in the domain directory):

```
APPHOST1> cd ORACLE_HOME/common/bin
APPHOST1> ./setNMProps.sh
```

Note: You must use the `StartScriptEnabled` property to avoid class loading failures and other problems.

2. Start Node Manager using the following command:

```
APPHOST1> cd ORACLE_BASE/product/fmw//wlserver_10.3/server/bin
APPHOST1> ./startNodeManager.sh
```

3. Repeat step1 and 2 for NodeManager in APPHOST2.

5.12.16 Starting the Oracle BAM System

To start the WLS_BAM1 managed server on APPHOST1:

1. Start the WLS_BAM1 managed servers:
   a. Log into the Oracle WebLogic Server Administration Console using the following URL:

      `http://apphost1vhn0:7001/console`

   b. In the Domain Structure window, expand the **Environment** node and then select **Servers**.

      The Summary of Servers page appears.

   c. Click the **Control** tab.
   d. Select **WLS_BAM1** from the Servers column of the table.
   e. Click **Start**.
2. Access `http://apphost1vhn1:9001/OracleBAM` to verify status of WLS_BAM1.
3. Start the WLS_BAM2 managed servers:
   a. Log into the Oracle WebLogic Server Administration Console at http://apphost1vhn0:7001/console.
   b. In the Domain Structure window, expand the **Environment** node and then select **Servers**.

      The Summary of Servers page appears.

   c. Click the **Control** tab.
   d. Select **WLS_BAM2** from the Servers column of the table.
   e. Click **Start**.
4. Access `http://apphost2:9001/OracleBAM` to verify status of WLS_BAM2.

If the managed servers fails to start with the following message:

```
Listener refused the connection with the following error:
ORA-12519, TNS:no appropriate service handler found
The Connection descriptor used by the client was <db_connect_string>
```

Verify that the PROCESSES initialization parameter for the database is set to a high enough value.

5.12.17 Configuring RAC Failover for the WLS_BAM Servers

Oracle BAM allows customizing the behavior of a BAM server when Oracle RAC is used as the repository for the BAM schemas and a failure occurs in the database. The properties that allow this customization can be adjusted depending on the application and based on the desired expected behavior for each BAM system. The properties are configured in the Fusion Middleware Control Console System MBean Browser, or in the corresponding Oracle BAM-specific XML configuration file.

If you want to completely disable Oracle BAM's failover to the database in a RAC configuration set `UseDBFailover` to `false` in the Fusion Middleware Control Console System MBean Browser for your BAM server. The default value of this property is true, therefore, failover is provided. You can also increase or decrease the number of retries in the access to the database when there is a database instance failure (for BAM to retry the in flight transactions). To adjust the number of retries change the `MaxDBNodeFailoverRetries` in the Fusion Middleware Control Console System MBean Browser. The default value for MaxDBNodeFailoverRetries is 5 times. See the *Oracle Fusion Middleware Administrator's Guide for Oracle SOA Suite* for more details on the properties that can be configured for Oracle BAM.

5.12.18 Configuring the BAM Web Applications to Use the BAM Server in APPHOST1

To configure the OracleBamWeb(WLS_BAM1) and OracleBamWeb(WLS_BAM2) applications to use the BAM server in APPHOST1:

1. Access Oracle Enterprise Manager Fusion Middleware Control at the following URL

   ```
   http:// apphost1vhn0:7001/em
   ```

2. Expand **BAM** in the navigation tree.
3. Right-click **OracleBamWeb(WLS_BAM1)**.
   - Select **BAM Web Properties** from the context menu.

     The BAM Web Properties page appears.
   - Define the following properties:
     – Enter **webhost:7777** for the application URL.
     – Enter **APPOST1VHN1** for the server name.
4. Select **OracleBamWeb(WLS_BAM2)** from the navigation tree and repeat these steps.

5.12.19 Configuring the ADCServer to Use the Appropriate BAMServer Address

To configure the ADCServer to use the correct BAMServer address:

1. Access Oracle Enterprise Manager Fusion Middleware Control using the followingURL:

   ```
   http://apphost1vhn0:7001/em
   ```

2. Expand BAM in the navigation tree.
3. Right-click **OracleBamServer(WLS_BAM1)**.
4. Select **System MBean Browser** from the context menu.
5. In the System MBean Browser, expand **oracle.bam.server**, located within Application Defined MBeans.

6. Click the **BamServerConfig** MBean.
7. Update the ADCServerName attribute by entering **APPHOST1VHN1** (server address of WLS_BAM1) as the value.
8. Update the ADCServerPort attribute to the listening port of the WLS server where the BAM server is running (9001).
9. Click **Apply**.

5.12.20 Configuring Oracle HTTP Server for the Administration Server and the WLS_BAMn Managed Servers

Enable Oracle HTTP Server to route to the administration server and the WSM-PM_Cluster, which contain the WLS_BAMn managed servers, by setting the WebLogicCluster parameter to the list of nodes in the cluster.

Add the following lines to the `mod_wl_ohs.conf` file, located in the *OHS_HOME*/instances/ohs_instance1/config/OHS/ohs1 directory:

```
# BAM Web app
<Location /OracleBAM>
    SetHandler weblogic-handler
    WebLogicCluster apphost1vn1:9001,apphost2:9001
</Location>

# WSM-PM
<Location /wsm-pm>
    SetHandler weblogic-handler
    WebLogicCluster apphost1vn1:9001,apphost2:9001
</Location>
```

Restart Oracle HTTP Server on WEBHOST1:

```
WEBHOST1> OHS_HOME/instances/ohs_instance1/bin/opmnctl restartproc
ias-component=ohs1
```

5.12.21 Validating Access through Oracle HTTP Server

Verify the following URLs to ensure that appropriate routing and failover is working from the HTTP Server to the SOA_Cluster.

1. While WLS_BAM2 is running, stop WLS_BAM1 from Oracle WebLogic Server Administration Console.
2. Access the following URLs and verify the appropriate functionality:
   - WebHost1:7777/wsm-pm
   - WebHost1:7777/OracleBAM
3. Start WLS_BAM1 from Oracle WebLogic Server Administration Console.
4. Stop WLS_BAM2.
5. Access the following URLs and verify the appropriate functionality:
   - WebHost1:7777/wsm-pm
   - WebHost1:7777/OracleBAM

5.12.22 Configuring Server Migration for the WLS_BAM Servers

The high availability architecture for Oracle BAM uses server migration to protect singleton services against failures. The WLS_BAM1 managed server is configured so that it can be restarted on APPHOST2 in case of failure. For this configuration, WLS_BAM1 listens on a specific floating IP address that is failed over by Oracle WebLogic Server Migration. To configure server migration for the WLS_BAM1 managed servers, follow these steps:

5.12.22.1 Setting Up the User and Tablespace for the Server Migration Leasing Table

To create the user and tablespace:

1. Create a tablespace called **leasing**. For example, log on to SQL*Plus as the sysdba user and run the following command:

   Example: Log on to SQL*Plus as the sysdba user and run the following command:

   ```
   SQL> create tablespace leasing
           logging datafile 'DB_HOME/oradata/orcl/leasing.dbf'
           size 32m autoextend on next 32m maxsize 2048m extent management local;
   ```

2. Create a user named **leasing** and assign to it the leasing tablespace.

   ```
   SQL> create user leasing identified by welcome1;

   SQL> grant create table to leasing;

   SQL> grant create session to leasing;

   SQL> alter user leasing default tablespace leasing;

   SQL> alter user leasing quota unlimited on LEASING;
   ```

3. Create the leasing table using the `leasing.ddl` script.

   a. Copy the `leasing.ddl` file located in the `ORACLE_BASE/product/fmw/wlserver_10.3/server/db/oracle` directory to your database node.

   b. Connect to the database as the leasing user.

   c. Run the `leasing.ddl` script in SQL*Plus.

   ```
   SQL> @copy_location/leasing.ddl;
   ```

5.12.22.2 Creating a Multi-Data Source from the WebLogic Server Administration Console

Create a multi-data source for the leasing table from the Oracle WebLogic Server Administration Console:

You create a data source to each of the RAC database instances during the process of setting up the multi-data source, both for these data sources and the global leasing multi-data source. When you create a data source:

- Make sure that this is a non-xa data source
- The names of the multi-data sources are in the format of *<MultiDS>-rac0*, *<MultiDS>-rac1*, and so on
- Use Oracle's Driver (Thin) Version 9.0.1, 9.2.0, 10, 11

- Use Supports Global Transactions, One-Phase Commit, and specify a service name for your database
- Target these data sources to the SOA cluster

Creating a Multi-Data Source

To create a multi-data source, complete these steps:

1. From Domain Structure window in the Oracle WebLogic Server Administration Console, expand the **Services** node, then expand the **JDBC** node.
2. Click Multi Data Sources. The Summary of JDBC Multi Data Source page appears.
3. Click **New**. The Create a New JDBC Multi Data Source page appears.
4. Click **Lock and Edit**.
5. Click **New**.
6. Enter leasing as the Name
7. Enter jdbc/leasing as the JNDI name.
8. Select **Failover as algorithm (default)**.
9. Click **Next.**
10. Select **non-XA driver (the default)**.
11. Click **Next**.
12. Click **Create New Data Source**.
13. Enter *leasing-rac0* as name. Enter *jdbc/leasing-rac0* as JNDI name. Enter *oracle* as the database type. For the driver type, enter *Oracle Driver (Thin) for RAC server-Instance connection Version 10,11.*

    Note: When creating the multi-datasources for the leasing table, enter names in the format of *<MultiDS>-rac0*, *<MultiDS>-rac1*, and so on.

14. Click **Next**.
15. Deselect **Supports Global Transactions**.
16. Click **Next**.
17. Enter the service name, database name, host port, and password for your leasing schema.
18. Click **Next**.
19. Click **Test Configuration** and verify the connection works.
20. Target the data source to the SOA cluster.
21. Select the data source and add it to the right screen.
22. Click **Create a New Data Source** and repeat the steps for the second instance of your RAC database.
23. Add the second data source to your multi-data source.
24. Click **Activate Changes**.

5.12.22.3 Edit the Node Manager's Properties File

The `nodemanager.properties` file is located in the *ORACLE_BASE*/product/fmw/wlserver_10.3/server/bin directory.

```
Interface=eth0
NetMask=255.255.255.0
UseMACBroadcast=true
```

- `Interface=eth0`

  This property specifies the interface name for the floating IP (eth0, for example).

- `NetMask`

  This property specifies the net mask for the interface for the floating IP.

- UseMACBroadcast

  This property specifies whether or not to use a node's MAC address when sending ARP packets, that is, whether or not to use the -b flag in the arping command.

Perform this configuration in the two nodes APPHOST1 and APPHOST2. Verify in the output of Node Manager (the shell where the Node Manager is started) that these properties are in use. Otherwise, problems may occur during migration. The output should be similar to the following:

```
...
StateCheckInterval=500
Interface=eth0
NetMask=255.255.255.0
...
```

5.12.22.4 Set Environment and Superuser Privileges for the wlsifconfig.sh Script

To set the environment and superuser privileges for the wlsifconfig.sh script:

1. Ensure that your PATH environment variable includes the files listed in Table 5–13.

Table 5–13 Files Required to be in the PATH Environment Variable

File	Located in this directory
`wlsifconfig.sh`	`ORACLE_BASE/product/fmw/user_projects/domains/bamdomain/bin/server_migration`
`wlscontrol.sh`	`ORACLE_BASE/product/fmw/wlserver_10.3/common/bin`
`nodemanager.domains`	`ORACLE_BASE/product/fmw/wlserver_10.3//common`

2. Grant sudo configuration for the wlsifconfig.sh script.
   - Configure sudo to work without a password prompt.
   - For security reasons, sudo should be restricted to the subset of commands required to run the `wlsifconfig.sh` script. For example, to set the environment and superuser privileges for the `wlsifconfig.sh` script, complete these steps:
     – Grant sudo privilege to the WebLogic user ('oracle') with no password restriction, and grant execute privilege on the `/sbin/ifconfig and /sbin/arping` binaries.

– Make sure the script is executable by the WebLogic user ('oracle'). The following is an example of an entry inside `/etc/sudoers` granting sudo execution privilege for oracle and also over `ifconfig` and `arping`:

```
oracle ALL=NOPASSWD: /sbin/ifconfig,/sbin/arping
```

Note: Ask the system administrator for the sudo and system rights as appropriate to this step.

5.12.22.5 Configure Server Migration Targets

Configuring Cluster Migration sets the `DataSourceForAutomaticMigration` property to **true**. Follow these steps to configure cluster migration in a cluster:

1. Log into the Oracle WebLogic Server Administration Console.
2. In the Domain Structure window, expand **Environment** and select **Clusters**.

   The Summary of Clusters page appears.
3. Click the cluster for which you want to configure migration (BAM_Cluster) in the Name column of the table.
4. Click the **Migration** tab.
5. In the Available field, select the machine to which to allow migration and click the right arrow. In this case, select **APPHOST1** and **APPHOST2**.
6. Select the data source to be used for automatic migration. In this case select the **leasing** data source.
7. Click **Save**.
8. Set the Candidate Machines for **Server Migration**.

   You must perform this task for WLS_BAM1:

   a. In Domain Structure window of the Oracle WebLogic Server Administration Console, expand **Environment** and select **Servers**.

   b. Select the server for which you want to configure migration.

   c. Click the **Migration** tab.

   d. In the Available field, located in the Migration Configuration section, select the machines to which to allow migration and click the right arrow. Select **APPHOST2** for **WLS_BAM1**.

   e. Select **Automatic Server Migration Enabled**.

      This enables the Node Manager to start a failed server on the target node automatically.

   f. Click **Save** and Activate the changes.

   g. Restart the Administration Server and the WLS_BAM1 server.

5.12.22.6 Test Server Migration

To verify that Server Migration is working properly, follow these steps:

From Node 1:

1. Kill the WLS_BAM1 managed server using the following command:

```
APPHOST1> kill -9 <pid>
```

`pid` specifies the process ID of the managed server. You can identify the `pid` in the node by running the following command:

```
APPHOST1> ps -ef | grep BAM_SOA1
```

2. In the Node Manager console, a message appears indicating that WLS_BAM1's floating IP has been disabled.
3. Wait for the Node Manager to try a second restart of WLS_BAM1.

   Node Manager waits for a fence period of thirty seconds before trying this restart.
4. Once Node Manager restarts the server, stop it again.

   Node Manager should now log a message indicating that the server will not be restarted again locally.

From APPHOST2:

1. Watch the local Node Manager console.

   After thirty seconds since the last try to restart WLS_BAM1on Node 1, Node Manager on APPHOST2 should prompt that the floating IP for WLS_BAM1 is being brought up and that the server is being restarted in this node.
2. Access the Oracle BAM console using APPHOST1VHN1/OracleBAM and WebHost1:7777/OracleBAM.

Verification From the Administration Console

You can verify Migration in the Administration Console:

1. Log into the Administration Console.
2. Click on **Domain** on the left console.
3. Click the **Monitoring** tab and then on the **Migration** tab.

The Migration Status table provides information on the status of the migration.

6 Configuring High Availability for Oracle ADF and WebCenter Applications

This chapter describes high availability concepts and configuration procedures for Oracle Application Development Framework (Oracle ADF) and Oracle WebCenter.

This chapter includes the following sections:

- Section 6.1, "Oracle ADF and High Availability Concepts"
- Section 6.2, "Configuring an Oracle ADF High Availability Deployment"
- Section 6.3, "Oracle WebCenter and High Availability Concepts"
- Section 6.4, "Configuring High Availability for Oracle WebCenter"
- Section 6.5, "Configuring High Availability for Custom Oracle ADF and WebCenter Applications"

6.1 Oracle ADF and High Availability Concepts

Oracle ADF is an end-to-end application framework that builds on Java Platform, Enterprise Edition (Java EE) standards and open-source technologies to simplify and accelerate implementing service-oriented applications. Oracle ADF is suitable for enterprise developers who want to create applications that search, display, create, modify, and validate data using web, wireless, desktop, or web services interfaces. Used in tandem, Oracle JDeveloper 11g and Oracle ADF provide an environment that covers the full development lifecycle from design to deployment, with drag-and-drop data binding, visual UI design, and team development features built in.

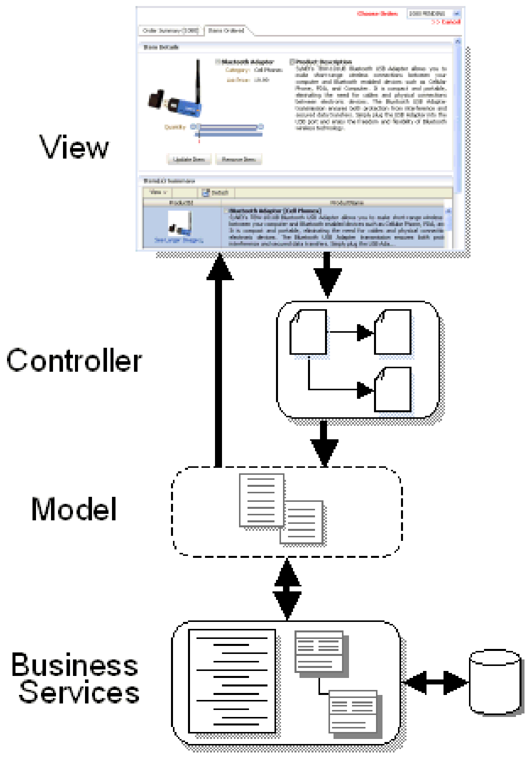
6.1.1 Understanding Oracle ADF

In line with community best practices, applications built using the Fusion web technology stack achieve a clean separation of business logic, page navigation, and user interface by adhering to a model-view-controller (MVC) architecture supported by the following layers.

As shown in Figure 6–1, in an MVC architecture:

- The model layer represents the data values related to the current page
- The view layer contains the UI pages used to view or modify that data
- The controller layer processes user input and determines page navigation
- The business service layer handles data access and encapsulates business logic

Figure 6–1 Overview of Oracle ADF Architecture



6.1.1.1 Oracle ADF Components

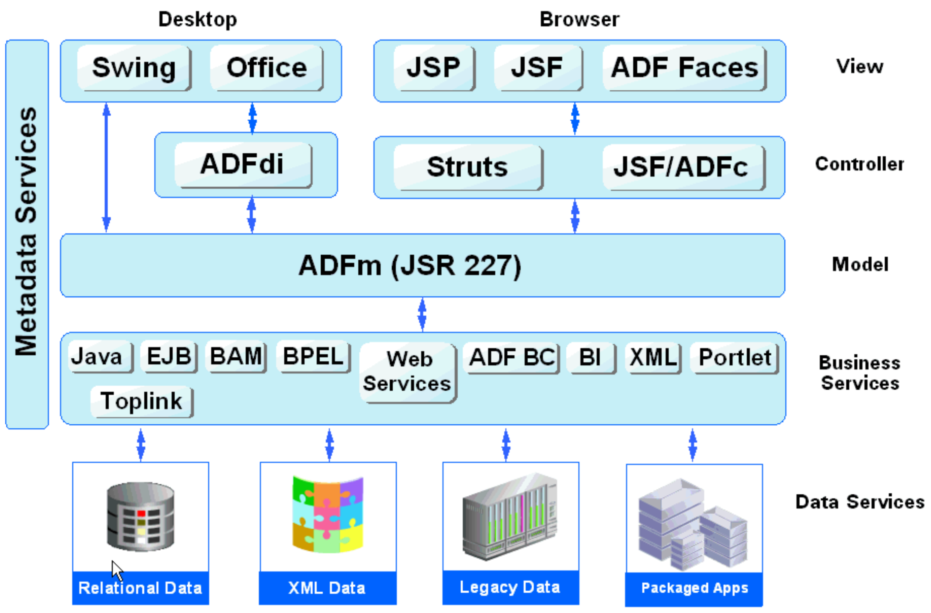
The core module in the framework is Oracle ADF Model, a declarative data binding facility that implements the JSR-227 specification. This specification provides an API for accessing declarative data binding metadata. The Oracle ADF Model layer enables a unified approach to bind any user interface to any business service, without the need to write code.

The other modules that make up a Fusion web application technology stack are:

- Oracle ADF Business Components, which simplifies building business services.
- Oracle ADF Faces, which offers a rich library of AJAX-enabled UI components for Web applications built with JavaServer Faces (JSF).
- Oracle ADF Controller, which integrates JSF with Oracle ADF Model. The ADF Controller extends the standard JSF controller by providing additional functionality, such as reusable task flows that pass control not only between JSF pages, but also between other activities, for instance method calls or other task flows.

Figure 6–2 illustrates where each Oracle ADF module fits in the Fusion web application architecture.

Figure 6–2 Simple Oracle ADF Architecture



6.1.1.1.1 ADF Business Components ADF Business Components are prebuilt application objects that accelerate the job of delivering and maintaining high-performance, richly functional, database-centric services. When building service-oriented Java EE applications, developers implement the core business logic as one or more business services. These backend services provide clients with a way to query, insert, update, and delete business data as required while enforcing appropriate business rules. ADF Business Components provides a ready-to-use implementation of Java EE design patterns and best practices.

Oracle ADF Business Components provides the following key components to simplify building database-centric business services:

- Entity object

  An entity object represents a row in a database table and simplifies modifying its data by handling all data manipulation language (DML) operations. It can encapsulate business logic to ensure that business rules are consistently enforced. Developers can associate an entity object with others to reflect relationships in the underlying database schema to create a layer of business domain objects to reuse in multiple applications.

- View object

  A view object represents a SQL query and simplifies working with its results. Developers use the SQL language to join, project, filter, sort, and aggregate data into the shape required by the end-user task being represented in the user interface. This includes the ability to link a view object with other entity objects to create master-detail hierarchies of any complexity. When end users modify data in the user interface, view objects collaborate with entity objects to consistently validate and save the changes.

- Application module

  An application module is the transactional component that UI clients use to work with application data. It defines an updateable data model and top-level procedures and functions (called service methods) related to a logical unit of work related to an end-user task.

For more information about Oracle ADF Business Components, go to the *Oracle Fusion Middleware Fusion Developer's Guide for Oracle Application Development Framework*.

6.1.1.1.2 ADF Model Layer In the model layer, Oracle ADF Model implements the JSR-227 service abstraction called the data control. Data controls abstract the implementation technology of a business service by using standard metadata interfaces to describe the service's operations and data collections, including information about the properties, methods, and types involved. In Oracle JDeveloper, developers can view that information as icons that they can easily drag and drop onto a page. When the developer drags the representation of the service onto the page, Oracle JDeveloper automatically creates the bindings from the page to the services. At runtime, the ADF Model layer reads the information describing the application's data controls and data bindings from appropriate XML files and implements the two-way connection between the user interface and the application's business service.

Oracle ADF provides out-of-the-box data control implementations for the most common business service technologies. Using Oracle JDeveloper and Oracle ADF together provides a declarative, drag-and-drop data binding experience for building user interfaces. Along with support for ADF Business Components application modules, ADF Model also provides support for the following service technologies:

- Enterprise JavaBeans (EJB) session beans and JPA entities
- JavaBeans
- Web services
- XML
- CSV files

For more information about Oracle ADF Model, go to the *Oracle Fusion Middleware Fusion Developer's Guide for Oracle Application Development Framework*.

6.1.1.1.3 ADF Controller In the controller layer, where handling page flow of the web applications is a key concern, ADF Controller provides an enhanced navigation and state management model on top of JSF. JDeveloper supports declarative creation of task flows that can manage application control between different types of activities, such as pages, methods on managed beans, declarative case statements, or calls to other task flows.

For more information about Oracle ADF Controller, go to the *Oracle Fusion Middleware Fusion Developer's Guide for Oracle Application Development Framework*.

6.1.1.1.4 ADF Faces Rich Client ADF Faces rich client (ADF Faces for short), is a set of standard JSF components that include built-in AJAX functionality. AJAX is a combination of asynchronous JavaScript, dynamic HTML (DHTML), XML, and `XmlHttpRequest` communication channel. This combination allows requests to be made to the server without fully rerendering the page. While AJAX allows rich client-like applications to use standard internet technologies, JSF provides server-side control, which reduces the dependency on an abundance of JavaScript often found in typical AJAX applications.

ADF Faces provides over 100 rich components, including hierarchical data tables, tree menus, in-page dialogs, accordions, dividers, and sortable tables. ADF Faces also provides ADF Data Visualization components, which are Flash- and SVG-enabled components capable of rendering dynamic charts, graphs, gauges, and other graphics that can provide a real-time view of underlying data. Each component also supports customization and skinning, along with internationalization and accessibility.

To achieve these front-end capabilities, ADF Faces components use a rendering kit that handles displaying the component and also provides the JavaScript objects needed for the rich functionality. This built-in support enables developers to build rich applications without needing extensive knowledge of the individual technologies on the front or back end.

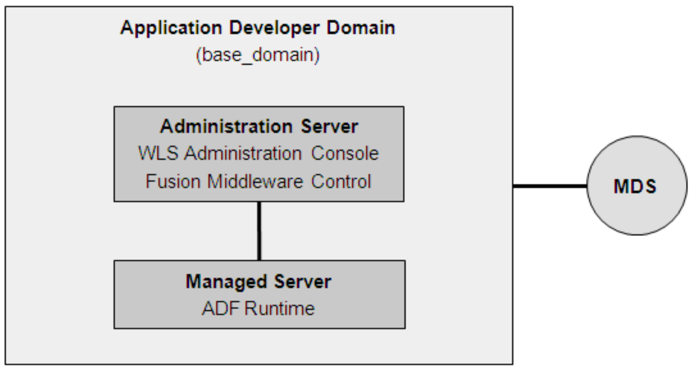
For more information about ADF Faces, including the architecture and detailed information about each of the components, go to the *Oracle Fusion Middleware Web User Interface Developer's Guide for Oracle Application Development Framework*.

6.1.1.2 Oracle ADF Single Node Architecture

You can install the Oracle ADF runtime to the Oracle WebLogic Server using either the Oracle JDeveloper Installer or the Oracle Fusion Middleware Application Developer Installer. The Application Developer Installer also lets you optionally install Fusion Middleware Control to provide web-based administration support for all Managed Servers in the domain. The Oracle JDeveloper installer does not install Fusion Middleware Control. Both of these options are described in the deployment chapter of the *Oracle Fusion Middleware Fusion Developer's Guide for Oracle Application Development Framework*.

When you use the Application Developer Installer to install the Oracle ADF runtime, it results in the creation of an Oracle Application Developer home directory (by default, `Oracle_APPDEV1`) located under the Middleware home. After the administrator uses the domain configuration wizard to create an Application Developer domain (`base_domain`) based on the JRF domain template, the administrator can configure the topology of the server. In a typical set up, the domain contains an Administration server containing the WLS Administration Console and Fusion Middleware Control. Typically, the Oracle ADF runtime libraries (part of the Java Required Files) get deployed to the Managed Servers, in addition to the user-facing custom Fusion web applications. To provide customization and personalization features, an optional MDS repository may also be installed, and needs to be configured separately. Figure 6–3 shows a basic single-node Oracle ADF architecture.

Figure 6–3 Basic Single-Node Oracle ADF Architecture


Application Developer Domain
(base_domain)
Administration Server
WLS Administration Console
Fusion Middleware Control
Managed Server
ADF Runtime
MDS

For more information about domains and servers, see the *Oracle Fusion Middleware Administrator's Guide*.

6.1.1.3 Oracle ADF External Dependencies

If the Fusion web application involves customization using Oracle Metadata Services (MDS), you should register your MDS repository with the Oracle WebLogic Server domain before you deploy the application. For information about registering MDS, see the *Oracle Fusion Middleware Administrator's Guide*.

Then, when you deploy the application, JDeveloper prompts you to choose the target metadata repository or shared metadata repository. You will be able to choose from the list of metadata repositories registered with the Oracle WebLogic Administration Server.

To ensure that you receive the metadata repository prompt, the application's `adf-config.xml` file must define a `cust-config` element in the `mds-config` section. This element specifies an ordered and named list of customization classes. A customization class is the interface that MDS uses to define which customization applies to the base definition metadata. In JDeveloper, you can use the overview editor for the `adf-config.xml` file to define a `cust-config` element.

For information about configuring the `adf-config.xml` file for MDS, see the chapter on customizing with MDS in the *Oracle Fusion Middleware Fusion Developer's Guide for Oracle Application Development Framework*.

For more information about the MDS architecture and metadata repositories (database and file-based) and archives (EAR, MAR), refer to the *Oracle Fusion Middleware Administrator's Guide*.

6.1.1.4 Oracle ADF Log File

The operations performed by the Fusion web application are logged directly to the WebLogic Managed Server where the application is running:

```
DOMAIN_HOME/servers/server_name/logs/server_name-diagnostic.log
```

The log files for the different WebLogic Managed Servers are also available from the Oracle WebLogic Server Administration Console. To verify the logs, access the Oracle WebLogic Server Administration Console `http://<admin_server_host>:<port>/console` and click on **Diagnostics-Log Files**.

This log's granularity and logging properties can be changed using Oracle Enterprise Manager Fusion Middleware Control (Fusion Middleware Control). Fusion Middleware Control is a Web browser-based, graphical user interface that you can use to monitor and administer a farm. To receive high availability warning diagnostic messages for Oracle ADF, the level should be set to `FINE`, as described in Section 6.1.4.3, "Troubleshooting Oracle ADF Replication and Failover Issues."

For more information about the level of diagnostics you can specify for Fusion web applications, see the chapter on testing and debugging in the *Oracle Fusion Middleware Fusion Developer's Guide for Oracle Application Development Framework*.

For details about using Fusion Middleware Control to change the log settings of WebLogic Managed Servers and Oracle ADF, see the *Oracle Fusion Middleware Administrator's Guide*.

6.1.2 Oracle ADF High Availability Considerations

Fusion web applications built on the Oracle ADF technology stack are Java EE applications (and J2EE applications). You should observe the same best practices for Fusion web applications as you would any other Java EE application in a high availability environment.

6.1.2.1 Oracle ADF Scope and Session State

At runtime, ADF objects such as the binding container and managed beans are instantiated. Each of these objects has a defined life span set by its scope attribute.

There are six types of scopes in a Fusion web application:

- Application scope: The object is available for the duration of the application.
- Session scope: The object is available for the duration of the session.
- Page flow scope: The object is available for the duration of a bounded task flow.
- Request scope: The object is available from the time an HTTP request is made until a response is sent back to the client.
- Backing bean scope: Used for managed beans for page fragments and declarative components only, the object is available from the time an HTTP request is made until a response is sent back to the client.
- View scope: The object is available until the view ID for the current view activity changes. This scope can be used to hold values for a given page. However, unlike request scope, which can be used to store a value needed from one page to the next, anything stored in view scope will be lost once the view ID changes.

When the Fusion web application runs in a clustered environment, a portion of the application's state is serialized and copied to another server or a data store at the end of each request so that the state is available to other servers in the cluster.

When you are designing an application to run in a clustered environment, you must:

- Ensure that all managed beans with a life span longer than one request are serializable (that is, they implement the `java.io.Serializable` interface). Specifically, beans stored in session scope, page flow scope, and view scope need to be serializable.
- Ensure that Oracle ADF is aware of changes to managed beans stored in ADF scopes (view scope and page flow scope) and enable the tracking of changes to ADF memory scopes.

When a value within a managed bean in either view scope or page flow scope is modified, the application needs to notify Oracle ADF so that it can ensure the bean's new value is replicated.

In Example 6–1, an attribute of an object in view scope is modified.

Example 6–1 Code that Modifies an Object in viewScope

```
Map<String, Object> viewScope =
    AdfFacesContext.getCurrentInstance().getViewScope();
MyObject obj = (MyObject)viewScope.get("myObjectName");
Obj.setFoo("newValue");
```

Without additional code, Oracle ADF will be unaware of this change and will not know that a new value needs to be replicated within the cluster. To inform Oracle ADF that an object in an ADF scope has been modified and that replication is needed, use the `markScopeDirty()` method, as shown in Example 6–2. The `markScopeDirty()` method accepts only `viewScope` and `pageFlowScope` as parameters.

Example 6–2 Additional Code to Notify Oracle ADF of Changes to an Object

```
controllerContext ctx = ControllerContext.getInstance();
ctx.markScopeDirty(viewScope);
```

This code is needed for any request that modifies an existing object in one of the ADF scopes. If the scope itself is modified by the scope's `put()`, `remove()`, or `clear()` methods, it is not necessary to notify Oracle ADF.

To enable ADF Controller to track changes to ADF memory scopes and replicate the page flow scope and view scope within the server cluster, you can enable the`<adf-scope-ha-support>` parameter in the `adf-config.xml` file, as described in Section 6.1.3.3, "Configuring adf-config.xml."

For more information about ADF object scopes, see the chapter on Fusion page lifecycle in the *Oracle Fusion Middleware Fusion Developer's Guide for Oracle Application Development Framework*.

6.1.2.2 Oracle ADF Failover and Expected Behavior

An Oracle WebLogic cluster provides application high availability. If one member of the cluster is unavailable, any other available member of the cluster is able to handle the request.

Session Failover Requirements

For seamless failover of a Fusion web application, the application must meet the following conditions:

- The application is in a cluster and at least one member of the application cluster is available to serve the request.
- For stateful applications, state replication is configured correctly as described in Section 6.1.3, "Configuring Oracle ADF for High Availability."
- If you are using Oracle HTTP Server, the server is configured with the WebLogicCluster directive to balance among all available application instances.
- If you are using a hardware load balancer, the load balancer is:
  - Routing traffic to all available instances
  - Configured correctly with a health monitor to mark unavailable instances
  - Configured to support persistence of session state

Expected Behavior for Application Failover

If the environment has been configured correctly, application users do not notice when an application instance in a cluster becomes unavailable. The sequence of events in an application failover is, for example, as follows:

1. A user makes a request and is routed by a hardware load balancer to instance A of the application.
2. Instance A of the application becomes unavailable because of node failure, process failure, or network failure.
3. The hardware load balancer marks instance A as unavailable.
4. The user makes a subsequent request. The request is routed to instance B.
5. Instance B is configured as a replication partner of Instance A and has the user's session state.
6. The application resumes using the session state on Instance B and the user continues working without interruption.

6.1.2.3 Oracle ADF Active Data Services

The Fusion technology stack includes the Active Data Service (ADS), which allows you to bind ADF Faces components to an active data source using the ADF Model layer. In JDeveloper, you configure individual components in your JSF pages to display active data. When you configure components to use active data, data is pushed to the client whenever a change event is raised by the data source. The data is pushed from the server to the client and displayed by the browser.

To support failover for pages configured to display active data, it is necessary to enable failover for the ADF Business Components application module and to enable ADF Controller to track changes to the ADF memory scopes. With these ADF settings configured, when failover occurs, the ADF server will detect the failover and request all pages configured to display active data to refresh themselves, after that ADS restarts and data is pushed to the client.

For details about how to enable failover for the Fusion web application, see Section 6.1.3, "Configuring Oracle ADF for High Availability."

For more information about using Oracle ADF Faces components with an active data service, see the chapter on ADS in the *Oracle Fusion Middleware Fusion Developer's Guide for Oracle Application Development Framework*.

6.1.2.4 Configuring the ADF Application Module for Oracle RAC

When configuring the ADF application module to access a highly available database system, such as redundant databases or Oracle Real Application Clusters (Oracle RAC) as the backend, the data source must be container-defined. In this scenario, it is required to use a multi data source; however, from the standpoint of the application module configuration, the naming convention for the multi data source is the same as it is an non-multi data source. This ensures that the correct data source will be used at runtime. For details about configuring multi data sources for high availability applications, see Section 4.1.3, "Configuring Multi Data Sources with Oracle RAC.".

6.1.3 Configuring Oracle ADF for High Availability

To support automatic replication and failover for web applications within a clustered environment, Oracle WebLogic Server supports mechanisms for replicating HTTP session state across clusters. You can configure Oracle ADF to ensure the Fusion web application's state can be restored from any server in the cluster.

6.1.3.1 Configuring Application Modules

An application module is the transactional component that UI clients use to work with application data. It defines an updateable data model and top-level procedures and functions (called service methods) related to a logical unit of work related to an end-user task. An application module supports storing of its transaction state as a snapshot in the database. It also supports the reverse operation of activating the transaction state from one of these saved snapshots.

To enable support for ADF Business Components failover, set the `jbo.dofailover` parameter to `true` so that the application module state is saved on release. This allows Oracle ADF to restore the application module state from a snapshot saved from a previous checkin. By contrast, when the failover feature is disabled, which it is by default, then application module state is saved only when the application is reused by a subsequent user session and only when the application module pool cannot find an unused application module.

You can set this parameter in your application module configuration on the **Pooling and Scalability** tab of the Edit Business Components Configuration dialog.

To configure application modules for high availability:

1. Launch JDeveloper and open the application.
2. In the Application Navigator, expand the project that contains the data model and locate the application module.
3. Right-click the application module and select **Configurations**.
4. Click **Edit**.
5. Click the **Pooling and Scalability** tab.
6. Select the **Failover Transaction State Upon Managed Release** checkbox.
7. Click **OK** to close the **Edit Business Components Configuration** dialog.
8. Click **OK** to close the **Manage Configurations** dialog.

For more information about Oracle ADF state management facilities and how to use them, see the chapter on application state management in the *Oracle Fusion Middleware Fusion Developer's Guide for Oracle Application Development Framework*.

6.1.3.2 Configuring weblogic.xml

To enable support for replicating HTTP session state, you must assign a value to the `persistent-store-type` element in the Oracle WLS `weblogic.xml` file. The value `replicated_if_clustered` ensures that the in-effect persistent store type will be replicated so that sessions on the clustered environment are stored in accordance with the value set for the cluster of servers to which this server belongs.

To configure the `weblogic.xml` file for high availability:

1. Launch JDeveloper and open the application.
2. In the Application Navigator, expand the project that contains the web application and expand the **WEB-INF** folder.
3. Double-click the **weblogic.xml** file, and click the **Source** tab to edit the file.
4. In the file, add the `persistent-store-type` definition to the `session-descriptor` element:

```
<weblogic-web-app>
    <session-descriptor>
    <persistent-store-type>
    replicated_if_clustered
    </persistent-store-type>
    </session-descriptor>
</weblogic-web-app>
```

6.1.3.3 Configuring adf-config.xml

When you are designing an application to run in a clustered environment, you must make sure that Oracle ADF is aware of changes to managed beans stored in ADF scopes (view scope and page flow scope).

When a value within a managed bean in either view scope or page flow scope is modified, the application needs to notify Oracle ADF so that it can ensure the bean's new value is replicated.

To enable ADF Controller to track changes to ADF memory scopes and replicate the page flow scope and view scope within the server cluster, you must set the ADF

Controller parameter `<adf-scope-ha-support>` in the application's `adf-config.xml` file to `true`. For example, when set to `true` for an application and that application adds or removes a bean from a page flow scope during a request, the change will automatically replicated within a cluster.

The `adf-config.xml` file is the central configuration file for all ADF components. The file contains sections to configure the runtime behavior for ADF components, including, ADF Controller.

To configure the `adf-config.xml` file for high availability:

1. Launch JDeveloper and open the application.
2. In the Application Navigator, expand the **Application Resources**.
3. Select **Descriptors**, and then select the **ADF META-INF** node.
4. Double-click the **adf-config.xml** file, and click the **Source** tab to edit the file.
5. Add the following to the file:

```
<adf-controller-config xmlns="http://xmlns.oracle.com/adf/controller/config">
 <adf-scope-ha-support>true</adf-scope-ha-support>
</adf-controller-config>
```

For more information about using the `adf-config.xml` file to configure ADF, see the chapter on creating complex task flows in the *Oracle Fusion Middleware Fusion Developer's Guide for Oracle Application Development Framework*.

6.1.4 Troubleshooting Oracle ADF High Availability

This section describes procedures for troubleshooting possible issues with Oracle ADF.

6.1.4.1 Troubleshooting Oracle ADF Development Issues

When you develop the Fusion web application in Oracle JDeveloper, the integrated development environment provides support for detecting potential High Availability issues. The warnings that JDeveloper provides are generated by the audit framework and will be triggered to display in the JDeveloper source editors. The warnings the editors display are based on the audit rules for High Availability applications.

The High Availability audit rules that JDeveloper enables by default are:

1. **ADF Controller Configuration - High Availability for ADF Scopes is not Enabled** simply warns the developer that the `adf-scope-ha-support` flag in the `adf-config.xml` file is set is not set to `true`. This audit rule fires only when the `<adf-controller-config>` element is present the ADF application-level configuration file (`adf-config.xml`).
2. **ADF Page Flows - EL Bean is Modified** warns the developer when the class calls a setter method on a bean to indicate that the scope must be marked dirty (using the `ControllerContext.markScopeDirty()` method). This audit rule fire only when the `adf-scope-ha-support` flag in the `adf-config.xml` file is set to `true`.
3. **ADF Page Flows - Managed Bean Class Not Serializable** warns the developer that a managed bean has a non-serializable class defined in `viewScope`, `pageFlowScope`, or `sessionScope`. This audit rule fire only when the `adf-scope-ha-support` flag in the `adf-config.xml` file is set to `true`.

You can modify the High Availability audit rule settings using the Preference dialog in JDeveloper. From the JDeveloper toolbar, choose **Tools - Preferences**, under **Audit -**

Profiles expand **ADF Controller Configuration** or **ADF Pages Flows** and make the desired audit rule selections.

You can also trigger the audit by choosing **Build - Audit** ***project.jpr*** from the JDeveloper toolbar.

6.1.4.2 Troubleshooting Oracle ADF Deployment Issues

Fusion web applications are deployed when the managed server is first started. Use the Oracle WebLogic Server Administration Console first to check that all application deployments were successful:

Click **Deployments** in the left hand pane. The right hand pane shows the application deployments and their status. The state of all applications, assuming all the servers are running, should be ACTIVE.

If an application deployment has failed, the server logs may provide some indication of why the application was not deployed successfully. The server logs are located in the `DOMAIN_HOME/servers/`*`server_name`*`/logs` directory. Common issues include:

- Unavailability of external resources, such as database resources. Examine the error, fix it, and attempt to redeploy the application.
- The appropriate applications or libraries are not targeted correctly to the right managed server or Cluster.

6.1.4.3 Troubleshooting Oracle ADF Replication and Failover Issues

State Replication is most prominent in failover scenarios. A user working on one server may discover that, upon failover:

- Windows may be closed or state might be reset.
- Screens may require a reset.
- The application may be redirecting to the logon screen.

The following steps provide guidance in troubleshooting and diagnosing state replication issues.

1. Confirm that this is not a known replication issue.

   See Section 6.1.2.2, "Oracle ADF Failover and Expected Behavior" for possible expected behaviors. Before proceeding to further diagnose the issue, first confirm that the failover behavior is not an expected behavior.

2. Check load balancer settings.

   For replication and failover to function correctly, the load balancer must be configured with the appropriate persistence settings. For more details on configuring Hardware Load Balancers for Oracle WebLogic Server, see *Oracle Fusion Middleware Using Clusters for Oracle WebLogic Server*.

3. Check the cluster status.

   Replication occurs within the context of a cluster. For failover to be successful, there must be at least one other healthy member of the cluster available. You can check cluster status in one of two ways:

   - Check the cluster status using the Oracle WebLogic Server Administration Console - In the Left-hand pane, click on **Servers**. Verify the state of all servers in the cluster.

- Check the cluster status using weblogic.Admin utility The weblogic.Admin command can be used to query the state of all servers in a specific cluster. For example:

  ```
  $ java weblogic.Admin -url Adminhost:7001 -username <username> -password
   <password>  CLUSTERSTATE -clustername Spaces_Cluster
  ```

  This example returns:

  ```
  There are 2 server(s) in cluster: Spaces_Cluster
  The alive servers and their respective states are listed below:
  Application Server---RUNNING
  Managed Server---RUNNING
  ```

4. Check cluster communications.

   Although Cluster members may all be running, there may be communication issues which prevent them from communicating replication information to each other. There are two types of cluster communication configurations. Troubleshooting depends on the cluster type:

   - Checking Unicast cluster communications - For Unicast clusters, managed servers must be able to access each other's hosts and each other's default listening port.

     Ensure that all individual managed servers have their Listen Address set correctly. You can find this setting by selecting **Configuration**, **General** for each managed server.

   - Checking Multicast cluster communications - For multicast clusters, servers must be able to intercept the same multicast traffic. Ensure that multicast is configured correctly by running the WebLogic utility `utils.MulticastTest` on each machine. For example:

     ```
     $ java utils.MulticastTest -H
     ```

5. Confirm Oracle WLS application configuration.

   Oracle WLS is not configured by default for failover by default. In-memory replication takes place only with the proper setting in the `weblogic.xml` file:

   ```
   <session-descriptor>
   <persistent-store-type>replicated_if_clustered</persistent-store-type>
   </session-descriptor>
   ```

   A persistent-store-type of replicated is also acceptable. This setting can be made in JDeveloper, as described in Section 6.1.3.2, "Configuring weblogic.xml."

6. Confirm Oracle ADF Business Components configuration.

   Oracle ADF is not configured by default for failover by default. Failover is supported only with the proper setting in the ADF Business Components configuration file (`bc4j.xml`):

   ```
   <AppModuleConfig ...
    <AM-Pooling jbo.dofailover="true"/>
   </AppModuleConfig>
   ```

   This setting is made in JDeveloper through the Edit Business Components Configuration dialog, as described in Section 6.1.3.1, "Configuring Application Modules."

7. Confirm Oracle ADF Controller configuration.

Oracle ADF is not configured by default to replicate changes to ADF objects in ADF memory scopes. ADF object replication is supported only with the proper setting in the ADF application-level configuration file (`adf-config.xml`):

```
<adfc:adf-controller-config>
 <adfc:adf-scope-ha-support>true</adfc:adf-scope-ha-support>
</adfc:adf-controller-config>
```

This setting is made in JDeveloper through the source editor, as described in Section 6.1.3.3, "Configuring adf-config.xml."

8. Check default logger messages.

   By default the ADF log display high-level messages (`INFO` level). The default logging often reports problems with serialization and replication without the need to enable more detailed log messages. For more information about the log, see Section 6.1.1.4, "Oracle ADF Log File."

9. Enable log messages for ADF high availability applications.

   Configure the ADF logger to output runtime messages for high availability. By default the ADF log display high-level messages (`INFO` level). You enable high availability diagnostics for ADF Controller by setting the logging level in Fusion Middleware Control to `FINE`.

   When enabled, the logger outputs a warning if the `adfc:adf-scope-ha-support` setting in the `adf-config.xml` file is not set. For more information about the ADF logger, see Section 6.1.1.4, "Oracle ADF Log File."

10. Enable debug.

    Check the server logs for any unusual messages on managed server startup. In particular, if the managed server is unable to locate other members of the cluster. The server logs are located in the `DOMAIN_HOME/servers/{SERVER_NAME}/logs` directory.

    For further debugging, enable the flags `DebugCluster`, `DebugClusterAnnouncements`, `DebugFailOver`, `DebugReplication`, and `DebugReplicationDetails`. Each flag can be enabled with the `weblogic.Admin` utility:

```
$ java weblogic.Admin -url Adminhost:7001 -username <username> -password
 <password> SET -type ServerDebug -property DebugCluster true
```

11. Enable component state serialization checking.

    Enable server checking to ensure no unserializable state content on session attributes is detected. This check is disabled by default to reduce runtime overhead. Serialization checking is supported by the Java server system property `org.apache.myfaces.trinidad.CHECK_STATE_SERIALIZATION`.

    For high availability testing, start off by validating that the Session and JSF state is serializable by launching the application server with the system property:

```
-Dorg.apache.myfaces.trinidad.CHECK_STATE_SERIALIZATION=session,tree
```

    If a JSF state serialization failure is detected, relaunch the application server with the system property to enable component and property flags and rerun the test:

```
-Dorg.apache.myfaces.trinidad.CHECK_STATE_SERIALIZATION=all
```

6.2 Configuring an Oracle ADF High Availability Deployment

This section describes how to configure an example Oracle ADF high availability deployment.

Note: Oracle strongly recommends reading the release notes for any additional installation and deployment considerations prior to starting the setup process.

6.2.1 Terminology for Directories and Directory Environment Variables

This list describes the directories and variables used in this section:

- ORACLE_BASE: This environment variable and related directory path refers to the base directory under which Oracle products are installed.
- MW_HOME: This environment variable and related directory path refers to the location where Oracle Fusion Middleware resides.
- WL_HOME: This environment variable and related directory path contains installed files necessary to host a WebLogic Server.
- ORACLE_HOME: This environment variable and related directory path refers to the location where Oracle Fusion Middleware SOA Suite is installed.
- DOMAIN directory: This directory path refers to the location where the Oracle WebLogic domain information (configuration artifacts) is stored.
- ORACLE_INSTANCE: An Oracle instance contains one or more system components, such as Oracle Web Cache, Oracle HTTP Server, or Oracle Internet Directory. An Oracle instance directory contains updateable files, such as configuration files, log files, and temporary files.

The values used and recommended for consistency for this directories are:

- ORACLE_BASE: `/u01/app/oracle`
- MW_HOME (application tier): `ORACLE_BASE/product/fmw`
- WLS_HOME: `MW_HOME/wlserver_10.3`
- ORACLE_HOME: `MW_HOME/adf`

6.2.2 Using RCU to Load Fusion Middleware Schemas in the Database

This step is required only if your ADF application needs to use any of schemas that are part of Oracle Fusion Middleware. Typically, this is done if the ADF application uses MDS repository, in which case you must install the 11g (11.1.1) Oracle Fusion Middleware Metadata Store into a Real Application Clusters (RAC) database before you install Oracle Fusion Middleware. Oracle Fusion Middleware provides a tool, the Oracle Fusion Middleware Repository Creation Utility (RCU), to create the component schemas in an existing database. See the *Oracle Fusion Middleware Repository Creation Utility User's Guide* for more information about installing RCU.

6.2.2.1 Running RCU

Run RCU to install the required metadata for Oracle Fusion Middleware 11*g*:

1. Start up RCU using the following command:

```
RCU_HOME/bin/rcu &
```

2. In the Welcome screen, click **Next**.
3. In the Create Repository screen, select **Create** to load component schemas into a database, and click **Next**.
4. In the Database Connection Details screen, enter connection information for your database:
   - **Database Type**: Select `Oracle Database`
   - **Host Name**: Enter the name of the node that is running the database. For an Oracle RAC database, specify the VIP name, or one of the node names as the host name: `ADFDBHOST1VIRTUAL`.
   - **Port**: The port number for the database: `1521`
   - **Service Name**: Enter the service name of the database: `adfha.mycompany.com`
   - **Username**: `SYS`
   - **Password**: Enter the password of the SYS user.
   - **Role**: `SYSDBA`

   Click **Next**.
5. If you receive the following message, click **Ignore** or **Stop**:

   The database you are connecting is with non-UTF8 charset, if you are going to use this database for multilingual support, you may have data loss. If you are not using for multilingual support you can continue, otherwise, Oracle strongly recommends using a UTF-8 database.
6. In the Select Components screen, select **Create a New Prefix**, and enter a prefix to use for the database schemas, for example, `ADFHA`.

   Write down the schema names so they are available in later procedures.

   Select the following schemas:
   - **AS Common Schemas**
   - **Metadata Services**

   Click **Next**.
7. In the Schema Passwords screen, enter passwords for the main and additional (auxiliary) schema users, and click **Next**.
8. In the Map Tablespaces screen, choose the tablespaces for the selected components, and click **Next**.
9. In the Summary screen, click **Create**.
10. In the Completion Summary screen, click **Close**.

See the *Oracle Fusion Middleware Repository Creation Utility User's Guide* for more information about installing RCU.

6.2.3 Installing Oracle HTTP Server on WEBHOST1

To install Oracle HTTP Server on WEBHOST1:

1. Verify that the servers meet the following requirements:

- The system, patch, kernel, and other requirements meet the requirements specified in the *Oracle Fusion Middleware Repository Creation Utility User's Guide*.
- This example uses port 7777. If you choose port 7777, ensure that it is not used by any service on the nodes. You can verify this by running the following command:

  ```
  UNIX:
  netstat -an | grep 7777

  Windows:
  netstat -an | findstr 7777
  ```

2. If port 7777 is in use, choose another port, or make it available.
3. On UNIX platforms, if the `/etc/oraInst.loc` or `/var/opt/oracle/oraInst.loc` file exists, check that its contents are correct. Specifically, check that the inventory directory is correct, and that you have write permissions for that directory.

   If the `/etc/oraInst.loc` file does not exist, skip this step.
4. Start Oracle Universal Installer for Oracle Fusion Middleware 11*g* Webtier Utilities CD installation as follows:

   For UNIX, run this command: `./runInstaller`

   For Windows, double-click `setup.exe`.
5. In the Specify Inventory Directory screen, enter the location for the inventory and the user group, and click **OK**.
6. Execute the `root` privileged actions as indicated in the dialog, and click **OK**.
7. In the Welcome screen, click **Next**.
8. In the Select Installation Type screen, select **Install and Configure**, and click **Next**.
9. In the Prerequisite Checks screen, ensure that all the prerequisites are met, and click **Next**.
10. In the Specify Installation Location screen, set the location to:

    ```
    /u01/app/oracle/product/11.1.1/ohs_1
    ```

    Click **Next**.
11. In the Configure Components screen:
    - Select `Oracle HTTP Server`.
    - Do not select **Associate Selected Components with WebLogic Domain**.

    Click **Next**.
12. In the Specify Component Details screen, enter the following values:
    - **Instance Home Location**:

      ```
      /u01/app/oracle/product/11.1.1/ohs_1/instances/ohs_instance1
      ```
    - **Instance Name**: `ohs_instance1`
    - **OHS Component Name**: `ohs1`

    Click **Next**.

13. In the Specify Webtier Port Details screen:
    - Select **Specify Custom Ports**. If you specify a custom port, select **Specify Ports using Configuration File** and then use the Browse function to select the file.
    - Enter Oracle HTTP Server port. For example, enter `7777`.

    Click **Next**.

    > **Note:** For more information about setting ports, refer to the *Oracle Fusion Middleware Installation Guide for Oracle SOA Suite*.

14. In the Configuration Summary screen, ensure that the selections are correct, and click **Install**.
15. In the Installation Progress screen:

    For UNIX systems, a dialog box appears prompting you to run the `oracleRoot.sh` script. Open a command window and run the script, following the prompts.

    Click **Next**.
16. In the Configuration screen, several configuration assistants are started in succession. When the configuration assistants are finished, the Configuration Completed screen appears.
17. In the Configuration Completed screen, click **Finish** to exit.

6.2.3.1 Validating Oracle HTTP Server

To verify that Oracle HTTP Server is set up correctly, access the root URL context of the server by entering the following URL in a web browser:

```
WebHost1:7777/
```

If Oracle HTTP Server is set up correctly, the Hello World page appears in the browser.

6.2.4 Installing the Oracle Fusion Middleware Home

Use the information in these sections to install Oracle Fusion Middleware components:

- Section 6.2.4.1, "Installing Oracle WebLogic Server"
- Section 6.2.4.2, "Installing Oracle Fusion Middleware for Oracle ADF Applications"

6.2.4.1 Installing Oracle WebLogic Server

To install Oracle WebLogic Server on all nodes in the application tier:

1. On UNIX platforms, if the `/etc/oraInst.loc` or `/var/opt/oracle/oraInst.loc` file exists, check that its contents are correct. Specifically, check that the inventory directory is correct and that you have write permissions for that directory.

   If the `/etc/oraInst.loc` file does not exist, skip this step.
2. Start the Oracle WebLogic Server installer.

   ```
   On UNIX (Linux used in this example):
   APPHOST1> server103_linux32.bin
   ```

```
On Windows:
APPHOST1> server103_win32.exe
```

3. In the Welcome screen, click **Next**.
4. In the Choose Middleware Home Directory screen:
   - Select **Create a New Middleware Home**.
   - For the **Middleware Home Directory** field, enter:

     ORACLE_BASE/product/fmw

   Click **Next**.
5. In the Register for Security Updates screen, enter your contact information for security update notifications, and click **Next**.
6. In the Choose Install Type screen, select **Custom**, and click **Next**.
7. In the JDK Selection screen, select only **JROCKIT SDK1.6.0_05**, and click **Next**.
8. In the Choose Product Installation Directories screen, accept the following directory:

   ORACLE_BASE/product/fmw/wlserver_10.3

   Click **Next**.
9. In the Installation Summary screen, click **Next**.
10. In the Installation Complete screen, deselect **Run QuickStart**, and click **Done**.

6.2.4.2 Installing Oracle Fusion Middleware for Oracle ADF Applications

To install Oracle Fusion Middleware for Oracle ADF, use the Application Developer Install and perform the following on all the nodes in the application tier:

1. Start the Oracle Fusion Middleware for Oracle Fusion Middleware 11*g* Application Developer installer:

   ```
   On UNIX (Linux used in this example):
   APPHOST1> runInstaller

   On Windows:
   APPHOST1> setup.exe
   ```

   When Oracle Fusion Middleware 11*g* Application Developer installer prompts you for a JRE/JDK location enter the Oracle SDK location created in the Oracle WebLogic Server installation in Section 6.2.4.1, "Installing Oracle WebLogic Server," for example:

   ORACLE_BASE/product/fmw/jrockit_160_05_R27.6.2-20

2. In the Welcome screen, click **Next**.
3. In the Prerequisite Check screen, verify that the checks complete successfully, and click **Next**.
4. In the Specify Installation Location screen:
   - For **Middleware Home**, enter: *ORACLE_BASE*/product/fmw
   - For **Oracle Home Directory**, enter the directory you want to use, for example: `adf`

Click **Next**.

5. In the Installation Summary screen, click **Install**.
6. In the Installation Complete screen, click **Finish**.

6.2.5 Administration Server High Availability

For information about configuring Oracle WebLogic Server Administration Server, see Chapter 10, "Active-Passive Topologies for Oracle Fusion Middleware High Availability."

6.2.6 Running the Configuration Wizard on APPHOST1 to Create the WebLogic Server ADF Domain

Run the Oracle Fusion Middleware Configuration Wizard from the `adf` directory in the Middleware home to create a domain containing the administration server and Oracle components.

1. Start Oracle WebLogic Server Configuration Wizard from the *`MW_HOME`*`/common/bin` directory using the following command:

   ```
   APPHOST1> ./config.sh
   ```

2. In the Welcome screen, select **Create a New WebLogic Domain** and click **Next**.
3. In the Select Domain Source screen, select **Generate a domain configured automatically to support the following products**, and select the following products:
   - **Oracle Enterprise Manager - 11.1.1.0**
   - **Oracle JRF - 11.1.1.0**

   Click **Next**.
4. Enter the **Domain Name**, **Domain Location**, and **Application Location** and click **Next**.
5. In the Configure Administrator Username and Password screen, enter the username and password to be used for the domain's administrator, and click **Next**.
6. In the Configure Server Start Mode and JDK screen, make the following selections:
   - **WebLogic Domain Startup Mode**: select **Production Mode**
   - **JDK Selection**: select **JROCKIT SDK1.6.0_05**

   Click **Next**.
7. In the Select Optional Configuration screen, select the following:
   - **Administration Server**
   - **Managed Servers, Clusters and Machines**

   Click **Next**.
8. In the Customize Server and Cluster Configuration screen, select **Yes**, and click **Next**.
9. In the Configure the Administration Server screen, enter the following values:
   - **Name**: `AdminServer`
   - **Listen Address**: `APPHOST1`

- **Listen Port**: `7001`
- **SSL listen port**: `NA`
- **SSL enabled**: Leave unchecked

Click **Next**.

10. In the Configure Managed Servers screen, add the following managed servers:

Managed Server Name	Listen Address	Listen Port	SSL Listen Port	SSL Enabled
WLS_ADF1	Hostname of APPHOST1	8889	NA	unchecked
WLS_ADF2	Hostname of APPHOST2	8889	NA	unchecked

Click **Next**.

11. In the Configure Clusters screen, add the following cluster:
    - **Name**: `ADF_CLUSTER`
    - **Cluster Messaging Mode**: `unicast`
    - **Cluster Address Enabled**: Leave blank

    Click **Next**.

12. In the Assign Servers to Clusters screen, assign the following servers to the Cluster:
    - ADF_CLUSTER:
      – WLS_ADF1
      – WLS_ADF2

    Click **Next**.

13. In the Configure Machines screen:
    - Delete the LocalMachine that appears by default.
    - Click the **Unix Machine** tab, and add the following machines:

Name	Node Manager Listen Address
APPHOST1	Hostname of APPHOST1
APPHOST2	Hostname of APPHOST2

Click **Next**.

14. In the Assign Servers to Machines screen, assign servers to machines as follows:
    - **APPHOST1**: `AdminServer`, `WLS_ADF1`
    - **APPHOST2**: `WLS_ADF2`
15. In the Configuration Summary screen, click **Create**.
16. In the Creating Domain screen, click **Done**.

6.2.6.1 Creating boot.properties for the Administration Server and Managed Servers on APPHOST1

This is an optional step for enabling the Administration Server to start up without prompting you for the administrator username and password. Create a `boot.properties` file for the Administration Server and for the managed servers on APPHOST1.

For the Administration Server, follow these steps:

1. Create the following directory:

   ```
   APPHOST1> mkdir -p MW_HOME/wls/user_
   projects/domains/adfdomain/servers/AdminServer/security
   ```

2. Use a text editor to create a file named `boot.properties` in the directory created in the previous step, and enter the following lines in the file:

   ```
   username=adminUser
   password=adminUserPassword
   ```

   For example:

   ```
   username=weblogic
   password=weblogic
   ```

> **Note:** When you start up the Administration Server or Managed Server, the username and password entries in the file are encrypted.
>
> For security reasons, minimize the time the entries in the file are left unencrypted. After you edit the file, start up the server as soon as possible in order for the entries to be encrypted.

For the WLS_ADF Managed Servers, follow these stesp:

1. Copy the file you created for the Administration Server to all servers:

   ```
   APPHOST1> mkdir -p servers/WLS_ADF1
   APPHOST1> mkdir -p servers/WLS_ADF1/security
   ```

2. Use a text editor to create a file named `boot.properties` in the directory created in the previous step, and enter the following lines in the file:

   ```
   username=adminUser
   password=adminUserPassword
   ```

   For example:

   ```
   username=weblogic
   password=weblogic
   ```

6.2.7 Starting the System in APPHOST1

This section describes procedures for starting the system in APPHOST1.

6.2.7.1 Starting the Administration Server on APPHOST1

To start the Administration Server on APPHOST1 run the following commands:

```
APPHOST1> cd ORACLE_BASE/product/fmw/user_projects/domains/adfdomain/bin
```

```
APPHOST1> ./startWebLogic.sh
```

6.2.7.2 Validating the Administration Server

To verify that the administration server is properly configured, follow these steps:

1. In a Web browser, go to `http://VIP1:7001/console`.
2. Log in as the administrator.
3. Verify that the WLS_ADF1 and WLS_ADF2 managed servers are listed.
4. Verify that the ADF_Cluster cluster is listed.
5. Verify that you can access Enterprise Manager at `http://VIP1:7001/em`.

6.2.7.3 Disabling Host Name Verification for the Administration Server and Managed Servers for APPHOST1 and APPHOST2

This step is required if you have not set up SSL communication between the Administration Server and the Node Manager. If SSL is not set up, you receive an error message unless you disable host name verification.

You can re-enable host name verification when you have set up SSL communication between the Administration Server and the Node Manager.

To disable host name verification on APPHOST1:

1. In Oracle WebLogic Server Administration Console, select **Administration Server**, **SSL**, and then **Advanced**.
2. Click **Lock and Edit**.
3. When prompted, save the changes and activate them.
4. Set **Hostname Verification** to **None**.
5. Select **WLS_ADF1**, **SSL**, and then **Advanced**.
6. Set **Hostname Verification** to **None**.

To disable host name verification on APPHOST2:

1. In Oracle WebLogic Server Administration Console, select **WLS_ADF2**, **SSL**, and then **Advanced**.
2. Set **Hostname Verification** to **None**.

6.2.7.4 Starting Node Manager on APPHOST1

Perform these steps to start Node Manager on APPHOST1:

1. Run the `setNMProps.sh` script, which is located in the *ORACLE_HOME*/common/bin directory, to set the `StartScriptEnabled` property to `true` before starting Node Manager:

   ```
   OAHOST1> cd ORACLE_HOME/common/bin
   SOAHOST1> ./setNMProps.sh
   ```

   Note: You must use the `StartScriptEnabled` property to avoid class loading failures and other problems.

2. Start Node Manager:

```
APPHOST1> cd ORACLE_BASE/product/fmw/wlserver_10.3/server/bin
APPHOST1> ./startNodeManager.sh
```

6.2.8 Installing Oracle WebLogic Server and Oracle ADF on APPHOST2

Repeat the procedures for installing WebLogic Server and Oracle ADF for APPHOST2, start with Section 6.2.3, "Installing Oracle HTTP Server on WEBHOST1." The directory paths for binary files and domains used when installing new nodes must be exactly the same as those used for the first node. If these paths and domains are not exactly the same as those used for the first node, failover does not occur.

6.2.9 Propagating the Domain Configuration to APPHOST2 with pack/unpack Utilities

Follow these steps to propagate the domain configuration to APPHOST2 using pack/unpack utilities:

1. Run the following `pack` command on APPHOST1 to create a template pack:

   ```
   APPHOST1> cd ORACLE_BASE/product/fmw/wlserver_10.3/common/bin
   APPHOST1> ./pack.sh -managed=true -domain=ORACLE_BASE/product/fmw/user_
   projects/domains/adfdomain/
   -template=adfdomaintemplate.jar
   -template_name=adf_domain_template
   ```

2. Run the following `scp` command command on APPHOST1 to copy the template file created in the previous step to APPHOST2:

   ```
   APPHOST1> scp adfdomaintemplate.jar
    user@node2:ORACLE_BASE/product/fmw/wlserver_10.3/common/bin
   ```

3. Run the `unpack` command on APPHOST2 to unpack the propagated template:

   ```
   APPHOST2> cd ORACLE_BASE/product/fmw/wlserver_10.3/common/bin
   APPHOST2> ./unpack.sh
    -domain=ORACLE_BASE/product/fmw/user_projects/domains/adfdomain/
    -template=adfdomaintemplate.jar
   ```

6.2.9.1 Creating boot.properties for the Administration Server and Managed Servers on APPHOST2

Create a `boot.properties` file for the Administration Server and for the Managed Servers on APPHOST2 by following these steps:

1. Create the following directories:

   ```
   APPHOST1> mkdir -p MW_HOME/wls/user_projects/domains/adfdomain/servers/WLS_ADF2
   APPHOST2> mkdir -p MW_HOME/wls/user_projects/domains/adfdomain/servers/WLS_
   ADF2/security
   ```

2. Use a text editor to create a file named `boot.properties` in the directory created in the previous step, and enter the following lines in the file:

   ```
   username=adminUser
   password=adminUserPassword
   ```

   For example:

   ```
   username=weblogic
   password=weblogic
   ```

Note: When you start up the Administration Server or Managed Server, the username and password entries in the file are encrypted.

For security reasons, minimize the time the entries in the file are left unencrypted. After you edit the file, start up the server as soon as possible in order for the entries to be encrypted.

6.2.9.2 Starting Node Manager on APPHOST2

To start the Node Manager on APPHOST2, repeat the steps from Section 6.2.7.4, "Starting Node Manager on APPHOST1" on APPHOST2.

6.2.9.3 Configuring the ADF Application for Replication

Use the procedures in this section to configure your application for replication.

Clustering Requirement

The application must be deployed to an Oracle WebLogic Cluster. This automatically establishes a replication channel for the multiple instances of the application.

Note: In a Unicast cluster, the default replication channel is configured using the Listen address of each managed server. Therefore, the Listen address should be configured to be a specific IP address or host name, instead of being configured to listen on `Any`.

Oracle ADF Replication

It is essential that Oracle ADF is configured properly. The following tag should be present in the `adf-config.xml` file, one of the Application Resources, for a stateful application:

```
<adfc:adf-controller-config><adfc:adf-scope-ha-support>true</adfc:adf-scope-ha-sup
port></adfc:adf-controller-config>
```

Applications must also have replication enabled. Oracle WebLogic Server allows several types of persistent stores for replication. For more information on persistent stores, see the *Oracle Fusion Middleware Using Clusters for Oracle WebLogic Server* manual.

The ADF application can be enabled by for this by default with the following setting in the `weblogic.xml` file:

```
<session-descriptor>
<persistent-store-type>replicated_if_clustered</persistent-store-type>
</session-descriptor>
```

The `replicated_if_clustered` setting disables replication for standalone application environments, and uses in-memory replication within a cluster environment.

Ensure that any custom application is configured for in-memory replication.

6.2.9.4 Deploying the ADF Application

After the cluster has been set up, the ADF application can be deployed. Be aware of the following:

- Deploy the ADF application to an EAR file and deploy using the Administration Server Console.
- Ensure that the deployment is to a cluster.
- If your application uses MDS, register the MDS with the domain using the instructions in the "Registering a Database-Based MDS Repository" section in the *Oracle Fusion Middleware Administrator's Guide* manual.

6.2.9.5 Configuring Oracle HTTP Server for the Administration Server and Oracle WebCenter Managed Servers

Enable Oracle HTTP Server to route to the administration server that contains Oracle WebCenter Managed Servers by setting the `WebLogicCluster` parameter to the list of nodes in the cluster. Follow these steps:

1. Add the following lines to the `OHS_HOME/instances/ohs_instance1/config/OHS/ohs1/mod_wl_ohs.conf` file:

```
# Spaces
<Location /applicationMountpoint>
    WebLogicCluster apphost1.com:8888,apphost2.com:8889
    SetHandler weblogic-handler
</Location>
```

2. Restart Oracle HTTP Server on WEBHOST1:

```
WEBHOST1> OHS_HOME/instances/ohs_instance1/bin/opmnctl restartproc
ias-component=OHS_COMPONENT1
```

6.2.9.6 Validating Access through Oracle HTTP Server

Verify the URLs to ensure that appropriate routing and failover is working from the HTTP Server to Oracle WebCenter cluster. Follow these steps:

1. Start `WLS_Spaces1`, `WLS_Spaces2`, `WLS_Portlet1` and `WLS_Portlet2` from the WebLogic Server Administration Console as follows:
   a. Access the Administration Console at the following URL:

   ```
   http://APPHOST1/console
   ```

   b. Click on one of the Managed Servers, for example, `WLS_Spaces1`.
   c. Select the **Control** tab.
   d. Select **Start** to start the Managed Server.
   e. Repeat the previous steps for each Managed Server.
   f. Verify direct access to the Managed Servers using the following URLs: (JOE: Ask Pradeep if applicationMountpoint in this example and in steps 3 and 6 should be a variable (in CodeInlineItalic font, and if so, whether he has an example of what to specify).

   ```
   apphost1:8888/applicationMountpoint

   apphost2:8888/applicationMountpoint

   apphost1:8889/applicationMountpoint

   apphost2:8889/applicationMountpoint
   ```

2. While `WLS_ADF2` is running, stop `WLS_ADF1` from Oracle WebLogic Server Administration Console.
3. Access the following URL and verify the appropriate functionality:

   ```
   WebHost1:7777/applicationMountpoint
   ```

4. Start `WLS_ADF1` from the WebLogic Server Administration Console.
5. Stop `WLS_ADF2`.
6. Access the following URL and verify the appropriate functionality:

   ```
   WebHost2:7777/applicationMountpoint
   ```

6.3 Oracle WebCenter and High Availability Concepts

The information in this section guides you through the issues and considerations necessary for designing an Oracle WebCenter high availability cluster.

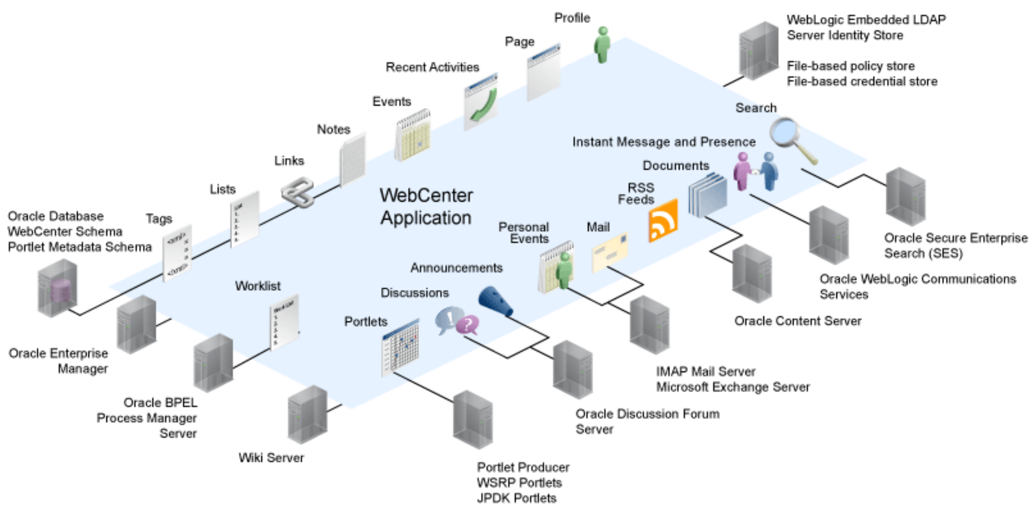
6.3.1 Understanding Oracle WebCenter

Oracle WebCenter combines the standards-based, declarative development of Java Server Faces (JSF), the flexibility and power of portals, and a set of integrated Web 2.0 services.

6.3.1.1 Oracle WebCenter Components

Figure 6–4 illustrates an Oracle WebCenter application and its associated components, portlets, and services.

Figure 6–4 Oracle WebCenter Application Components


Profile
Page
Recent Activities
Events
Notes
Links
Lists
Tags
WebLogic Embedded LDAP Server Identity Store
File-based policy store
File-based credential store
Search
Instant Message and Presence
Documents
RSS Feeds
WebCenter Application
Personal Events
Mail
Oracle Database WebCenter Schema Portlet Metadata Schema
Worklist
Announcements
Discussions
Portlets
Oracle Secure Enterprise Search (SES)
Oracle WebLogic Communications Services
Oracle Content Server
Oracle Enterprise Manager
Oracle BPEL Process Manager Server
Wiki Server
IMAP Mail Server
Microsoft Exchange Server
Oracle Discussion Forum Server
Portlet Producer
WSRP Portlets
JPDK Portlets

Oracle WebCenter includes the following components.

- **Oracle WebCenter Spaces** offers a single, integrated, Web-based environment for social networking, communication, collaboration, and personal productivity through a robust set of services and applications.

  WebCenter Spaces is built using JSF, Oracle ADF, WebCenter Framework, WebCenter Web 2.0 services, and Composer. Oracle WebCenter Spaces provides:

- A browser-based, community-focused application targeting the business user
- Personal Space is an environment that provides personal productivity tools, such as emails and notes.
- Group Space is a rich team collaboration platform.
- Threaded discussions, blogs, wikis, worklists, announcements, RSS, recent activities, and search

- **Oracle WebCenter Portlets Framework** supports deployment and execution of both standards-based portlets (JSR 168, WSRP 1.0 and 2.0), and traditional Oracle PDK-Java based portlets. Oracle WebCenter provides several out-of-the-box producers, such as OmniPortlet, Web Clipping, Rich Text Portlet, and WSRP Tools.
- **Oracle WebCenter Framework** provides the following capabilities:
  - Runtime customization, which lets you make in-place changes to the application without redeploying it.
  - Support for JSR-168 standards-based WSRP portlets, and PDK-Java portlets
  - Content integration through JCR (JSR170) including Oracle Content Server, file system, Oracle Portal, Documentum, Sharepoint, and Lotus
  - JSF-Portlet Bridge, which lets you expose JSF pages and Oracle ADF task flows as standards-based portlets
- **Oracle WebCenter Discussion Server** provides the ability to integrate discussion forums and announcements into your applications.
- **Oracle WebCenter Wiki and Blog server** provides the ability to integrate wikis and blogs into your applications. It also supports features that enable application users to create their own wikis and blogs.

Oracle WebCenter Spaces and WebCenter custom applications can also integrate with the following WebCenter Web 2.0 social networking and personal productivity services:

- **Instant Messaging and Presence (IMP)** - Provides the ability to observe the status of other authenticated users (whether online, offline, busy, or idle) and to contact them instantly.
- **Announcements** - Provides the ability to post announcements about important activities and events to all authenticated users.
- **Wiki** - Provides the ability for geographically diverse teams to originate and collaborate on Web documents.
- **Blog** - Provides the ability to post announcements about important activities and events to all authenticated users.
- **Discussions** - Provides the ability to create threaded discussions, posing and responding to questions, and searching for answers. Also provides an effective group communication mechanism for important activities and events.
- **Mail** - Provides easy integration with IMAP and SMTP mail servers to enable users to perform simple mail functions such as viewing, reading, creating, and deleting messages, creating messages with attachments, and replying to or forwarding existing messages.
- **Worklists** - Provides a personal, at-a-glance view of business processes that require attention. These can include a request for document review, and other types of business process that come directly from enterprise applications.

- **Recent Activities** - Provides a summary view of recent changes to documents, discussions, and announcements.
- **Notes** - Available in Oracle WebCenter Spaces, the Notes service provides the ability to "jot down" and retain quick bits of personally relevant information.
- **Search** - Provides the ability to search tags, services, the application, or an entire site. This includes integrating Oracle Secure Enterprise Search for WebCenter searches.
- **RSS** - Provides the ability to access the content of many different Web sites from a single location—a news reader.
- **Tags** - Provides the ability to assign one or more personally relevant keywords to a given page or document.
- **Documents** - Provides content management and storage capabilities, including content upload, file and folder creation and management, file check out, versioning, and so on.
- **Lists** - Available in Oracle WebCenter Spaces, the Lists service provides the ability to create, publish, and manage lists and tables. Users can create lists from prebuilt structures or create their own custom lists.
- **Links** - Provides the ability to view, access, and associate related information; for example you can link to a solution document from a discussion thread.
- **Events** - Available in WebCenter Spaces, the Events service provides the ability to create and maintain a schedule of events relevant to a wider group of users. Events are published to all authenticated users.

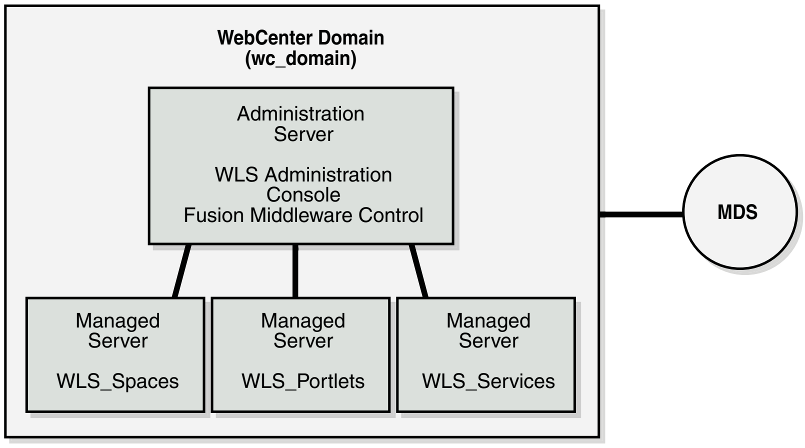
For information about how to configure these WebCenter Web 2.0 social networking and personal productivity services, see *Oracle Fusion Middleware Administrator's Guide for Oracle WebCenter*.

6.3.1.2 Oracle WebCenter Single-node Architecture

Oracle WebCenter installation creates a WebCenter directory under the Middleware home directory. The installation creates a WebCenter domain (`wc_domain`), which contains the Administration Server and three WebLogic Managed Servers: `WLS_Spaces` (which hosts Oracle WebCenter Spaces), `WLS_Portlets` (which hosts Oracle WebCenter Portlets), and `WLS_Services` (which hosts Oracle WebCenter Discussion server, the Oracle WebCenter Wiki and Blog server, and any additional WebCenter Web 2.0 services that you choose to integrate).

An optional fourth managed server (an application server) can be used to run custom WebCenter applications. Typically, there is one custom managed server per custom Webcenter application. When you create additional managed servers, they are provisioned with the appropriate libraries to enable them to draw upon the same external resources as Oracle WebCenter Spaces. For more information about managed servers, see "Understanding Oracle Fusion Middleware Concepts" in the *Oracle Fusion Middleware Administrator's Guide*. Figure 6–5 shows a basic single-node Oracle WebCenter architecture.

Figure 6–5 Basic Single-Node Oracle WebCenter Architecture



6.3.1.3 Oracle WebCenter State and Configuration Persistence

Oracle WebCenter Spaces runs as a Java EE application with application state and configuration persisted to the MDS repository. User session information within the application is held locally in memory. In a cluster environment, this state is replicated to other members of the cluster.

Customizations within a portlet or service environment is persisted by that service. Out of the box, Oracle portlets and any custom portlets you build, the Discussion Server, and Wiki Server all have their own database persistence mechanisms.

6.3.1.4 Oracle WebCenter External Dependencies

Table 6–1 shows the access type for each of the WebCenter components and services. The Configuration column lists the type of information provided to Oracle WebCenter to configure or initialize the connection. The Access column lists the protocol used in runtime access of the service.

The Discussion Server and the wiki/blog server are also service providers to Oracle WebCenter. Oracle Portlets also exposes Portlet Producers as services.

Unavailability of these services does not prevent Oracle WebCenter Spaces application from starting, although application errors may be seen when running. The exception is the MDS repository: WebCenter Spaces does not work without the MDS repository. WebCenter Spaces partially works without the WebCenter database, if it is a different physical database from the MDS repository.

Table 6–1 Oracle WebCenter Access Types

Service	Configuration	Access
Discussion server	HTTP access to Discussion server administration	SOAP/HTTP
Wiki/blog server	HTTP access to Wiki server administration	SOAP/HTTP
Presence server	HTTP access to Presence server	SOAP/HTTP
Oracle Content Server	Socket connection to the Administration Server. HTTP access is required only if the content server needs to be accessed outside WebCenter.	JCR 1.0 over socket or HTTP
MDS repository and schemas	JDBC	JDBC

Table 6–1 (Cont.) Oracle WebCenter Access Types

Service	Configuration	Access
Search	HTTP access to Search server	HTTP
Mail server	IMAP/SMTP server	IMAP/SMTP
Worklist	HTTP access to BPEL server	SOAP/HTTP
Portlets	HTTP location of provider WSDLs	SOAP/HTTP

Configure each of the external services independently for high availability. Oracle WebCenter's framework provides only one point of access for external services.

For HTTP Services, for example, the access URL should be directed to a load balancer, which provides access to multiple service providers on the back end.

Instructions for configuring Oracle WebCenter Discussion and Wiki Servers for high availability are provided in Section 6.4.5, "Running Oracle Fusion Middleware Configuration Wizard on APPHOST1 to Create the WebLogic Server WebCenter Domain."

For the MDS repository and schemas, Oracle recommends an Oracle Real Application Clusters (RAC) database as the back-end database. Instructions for configuring Oracle RAC as a database provider are in Section 6.4.5, "Running Oracle Fusion Middleware Configuration Wizard on APPHOST1 to Create the WebLogic Server WebCenter Domain."

For information about multi data source configuration with RAC and the MDS repository, see Section 4.1.2, "Using Multi Data Sources with Oracle RAC."

6.3.1.5 Oracle WebCenter Configuration Considerations

The main configuration files for WebCenter applications are listed and described in Table 6–2. Both these files are supplied within the WebCenter application deployment .EAR file.

Table 6–2 Oracle WebCenter Configuration Files

Artifact	Purpose
`adf-config.xml`	Stores basic configuration for Application Development Framework (ADF) and WebCenter application settings, such as which discussions server or mail server the WebCenter application is currently using.
`connections.xml`	Stores basic configuration for connections to external services.

WebCenter applications and portlet producers both use the Oracle Metadata Services (MDS) repository to store their configuration data and they access the MDS repository as a JDBC data source within the Oracle WebLogic framework.

The MDS repository stores post deployment configuration changes for WebCenter applications and portlet producers as customizations. MDS uses the original deployed versions of `adf-config.xml` and `connection.xml` as base documents and stores all subsequent customizations separately into MDS using a single customization layer.

When a WebCenter application starts up, customizations stored in MDS are applied to the appropriate base documents and the WebCenter application uses the merged documents (base documents with customizations) as the final set of configuration properties.

For WebCenter applications that are deployed to a server cluster, all members of a cluster read from the same location in the MDS repository.

Typically, there is no need for administrators to examine or manually change the content of base documents (or MDS customization data) for files such as `adf-config.xml` and `connection.xml` as Oracle provides several administration tools for post deployment configuration.

Note: Oracle does not recommend editing `adf-config.xml` or `connection.xml` by hand (unless specifically instructed to do so) as this can lead to misconfiguration.

WebCenter applications and portlet producers store post-deployment configuration information in MDS but configuration information for Oracle WebCenter Discussions Server and Oracle WebCenter Wiki and Blog Server is stored in the file system (see Table 6–3).

Table 6–3 Oracle WebCenter Configuration Location

Application	Configuration Stored in MDS	Configuration Stored in the File System
WebCenter Spaces	Yes	No
Custom WebCenter applications	Yes	No
Portlet producers	Yes	No
Discussions server	No	Yes
Wiki and Blog server	No	Yes

The Oracle WebCenter Discussions Server stores configuration information in `DOMAIN_DIR/config/fmwconfig/servers/SERVER_NAME/owc_discussions_11.1.1.1.0/jive_startup.xml`. This configuration information is specific to a particular instance of the discussions server.

The Oracle WebCenter Wiki and Blog Server stores configuration information in the server's deployment directory. For example, `$DOMAIN_HOME/servers/SERVER_NAME`. Its configuration file, `application_config.script`, is located in `$WIKI_HOME/WEB-INF/classes`. For example, `DOMAIN_HOME/servers/WLS_Services/stage/owc_wiki/11.1.1.1.0/owc_wiki/WEB-INF/classes`.

6.3.1.6 Oracle WebCenter Log File Locations

The operations performed by Oracle WebCenter, Portlet Producers, and Discussion and Wiki Servers are logged directly to the WebLogic Managed Server where the application is running:

```
DOMAIN_HOME/servers/SERVER_NAME/logs/SERVER_NAME.log
```

The log files for the different WebLogic Managed Servers are also available from Oracle WebLogic Server Administration Console. To verify the logs, access Oracle WebLogic Server Administration Console `http://<admin_server_host>:<port>/console` and click on **Diagnostics-Log Files**.

In addition, a diagnostic log is produced in the log directory for the managed server. This log's granularity and logging properties can be changed through the `DOMAIN_HOME/config/fmwconfig/servers/SERVER_NAME/logging.xml` file.

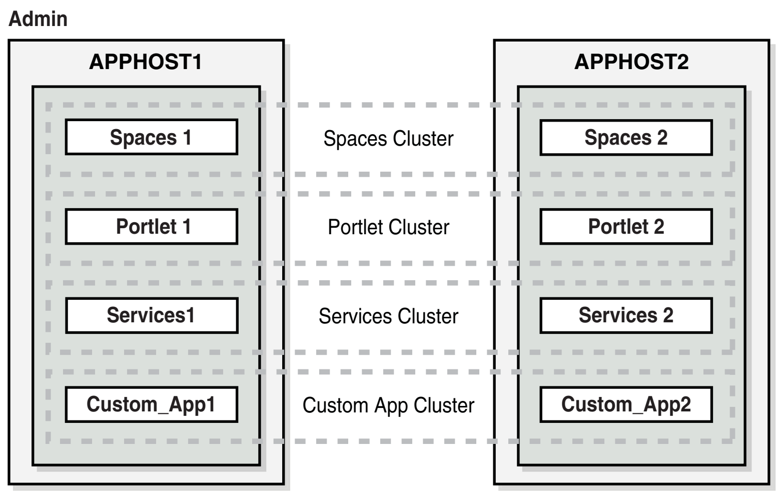
6.3.2 WebCenter High Availability Architecture and Failover Considerations

An Oracle WebLogic cluster provides high availability for applications. When one member of the cluster is unavailable, another member of the cluster handles the request.

Each of the managed servers in an Oracle WebCenter deployment can be deployed as a cluster, with different cluster members either on the same node, or on different nodes. In Figure 6–6, all requests sent to the cluster can be served equally by either member of the cluster.

The following sections contain more information on the runtime and configuration aspects of an Oracle WebCenter cluster.

Figure 6–6 Oracle WebCenter Two-Node High Availability Architecture



6.3.2.1 Oracle WebCenter Applications

All of the managed servers in Oracle WebCenter are provisioned with both system libraries and Oracle ADF libraries. In addition, Table 6–4 lists the applications which run on each of the managed servers.

Table 6–4 Oracle WebCenter Managed Servers and Applications

Managed Server	Application(s)
Spaces	webcenter webcenter-help
Portlets	portalTools wsrp-tools
Services (Discussion/Wiki Server)	owc_discussions owc_wiki

6.3.2.2 Oracle WebCenter Startup Order

When a managed server is started, applications and libraries are started in the following order:

1. Oracle System libraries, known as the JRF libraries
2. Oracle ADF libraries
3. Instrumentation applications, such as Oracle DMS
4. Oracle WebCenter applications, shown in Table 6–1

The startup order is also the order of dependency. If a dependent component does not deploy successfully, a later component may not function correctly. The successful startup of Oracle WebCenter applications themselves is independent of the availability of dependent services.

6.3.2.3 Deploying WebCenter Application on a Cluster

For an Oracle WebCenter cluster deployment, such as the one shown in Figure 6–6, follow these rules for the targeting of applications, libraries, and system resources:

- Target applications and libraries to the cluster target. For example, target the Oracle WebCenter application to the Spaces cluster.
- JDBC resources to the cluster target.

Oracle WebCenter applications and Wiki server are deployed as Oracle WebLogic *stage* applications. During the initial deployment of each application, the deployment files are received from the Administration Server and deployed locally. The exception is the Oracle WebCenter Wiki and Blogs Server application, which is deployed as a *nostage* application.

Cluster Communications

By default, each Oracle WebCenter cluster is configured as unicast. To configure your Oracle WebCenter cluster for multicast, see *Oracle Fusion Middleware Using Clusters for Oracle WebLogic Server*.

6.3.2.4 Oracle WebCenter State Replication

Oracle WebCenter relies on Oracle ADF, which has several stateful components. WebCenter itself is also a stateful application. Therefore, you must configure state replication in cluster scenarios.

For more information on how state replication works in Oracle WebLogic Server, see *Oracle Fusion Middleware Using Clusters for Oracle WebLogic Server*.

Oracle WebLogic Server supports two types of state replication:

- **In-memory replication** - Using in-memory replication, Oracle WebLogic Server copies a session state from one server instance to another. The primary server creates a primary session state on the server to which the client first connects, and a secondary replica on another WebLogic Server instance in the cluster. The replica is kept up to date so that it may be used if the server that hosts the servlet fails.
- **JDBC-based persistence** - In JDBC-based persistence, Oracle WebLogic Server maintains the HTTP session state of a servlet or JSP using file-based or database-based persistence.

Properly configuring state replication requires both configuring the environment and configuring the application properly. For information on state replication behavior under failover conditions see Section 6.3.2.7, "Expected Behavior for Application Failover."

For more information on diagnosing state replication issues see Section 6.3.2.4, "Oracle WebCenter State Replication."

6.3.2.5 Understanding the Distributed Java Object Cache

Oracle WebCenter applications use a distributed Java Object Cache for greater performance. Configure this cache across all Oracle WebCenter clusters, with one distributed cache per cluster.

Table 6–5 lists some examples of the type of objects which Oracle WebCenter places in the Java Object Cache.

Table 6–5 Oracle WebCenter Object Types for Java Object Cache

Oracle WebCenter Component	Object Cached
Discussion Forums	Topics and Forums.
Announcements	Announcements.
Presence	Presence Subscription list.
Worklist	Called Worklist items, such that cached data is used until refresh is called.
Content Integration (DocLib)	"isProvisoned" and "isConfigured" state.
Content Integration (Portal adapter)	JCR: type information and metadata obtained from the repository.
Service Framework	Userprofile. Queried usernames.
Recent Activity	Recent activity results per user.
Spaces Application	Global List of all space names, template names in the instance. List of all spaces/templates on which a user has access.
Page Service	List of pages in a scope.
WSRP Server	Preference store values of wsrp producers.
IMP service	User's presence/subscription status
Doclib	Provisioning and configuration checks for doclib service for Spaces application configuration

Collaboration

Collaboration services cache objects in the Java Object Cache on a per user session basis. These cached user sessions are destroyed when the HTTP session is destroyed.

Worklist

Worklist caches the called items so that unless the refresh button is clicked, or the every fifteen minute refresh poll is triggered, the same items are displayed as the user changes the sort and group by settings. The display is also cached by group and sort order settings, updating when a change occurs. This is read only data which, if not present, is fetched and stored on the cache.

Spaces

The list of all templates and public spaces are maintained in the Java Object Cache. This is shared by all users, in addition to the list of spaces and templates that a particular user can access. These are cahced per user. A template created on one JVM does not show up if the template cache has been initialized in another JVM, unless JOC distributes the data, or an admin in the second JVM does an explicit refresh where the cache is rebuilt.

Doclib

Doclib uses Java Object Cache to cache provisioning and configuration checks for doclib services, as applies to Spaces appplication configuration. For provisioning, cached objects are flagged as distributed. They are replicated by a correctly configured Java Object Cache in a high availability environment, the confiuguration cached state is kept locally. All cached objects are flagged to expire after one minute. Caching reduces the number of times UCM calls are made to check the state of the UCM Server, as the Spaces application repeatedly checks provisioning and configuration to control service rendering in the UI.

Recent Activity

The list of recent activity results are cached for each user to prevent a requery of results each time the recent activity taskflow or RSS feed is viewed. The cache is automatically refreshed every fifteen minutes, or can be manually refreshed using the refresh icon in the recent activity taskflow.

For Java Object Cache configuration procedures, see Section 6.4.13, "Configuring the Java Object Cache."

6.3.2.6 Oracle WebCenter Protection from Failover and Expected Behavior

An Oracle WebLogic cluster provides application high availability. If one member of the cluster is unavailable, any other available member of the cluster is able to handle the request.

Session Failover Requirements

For seamless failover of an Oracle WebCenter application, the application must meet the following conditions:

- The application is in a cluster and at least one member of the application cluster is available to serve the request.
- For stateful applications, state replication is configured correctly as described in Section 6.4.15, "Configuring Oracle WebCenter for Replication."
- If you are using Oracle HTTP Server, the server configuration is configured with the WebLogicCluster directive to balance among all available application instances.
- If you are using a hardware load balancer, the load balancer is:
  - Routing traffic to all available instances
  - Configured correctly with a health monitor to mark unavailable instances
  - Configured to support persistence of session state

6.3.2.7 Expected Behavior for Application Failover

If the environment has been configured correctly, application users do not notice when an application instance in a cluster becomes unavailable. The sequence of events in an application failover is, for example, as follows:

1. A user makes a request and is routed by a hardware load balancer to instance A of the application.
2. Instance A of the application becomes unavailable because of node failure, process failure, or network failure.
3. The hardware load balancer marks instance A as unavailable.
4. The user makes a subsequent request. The request is routed to instance B.
5. Instance B is configured as a replication partner of Instance A and has the user's session state.
6. The application resumes using the session state on Instance B and the user continues working without interruption.

Exceptions to Expected Behavior

For Oracle WebCenter, the known exceptions to these expected behaviors are as follows:

- **Oracle ADF Pop-ups** - Open pop-ups are closed on failover. This affects many components which otherwise have no exceptions. The following components are affected:
  - Composer (property inspector Popup)
  - Lists
  - Links (link deletion popup)

  When failover occurs, the action which led to the popup must be repeated in order to make it reappear. The specific ways in which this appears in Oracle WebCenter Spaces, as well as suggested remedies are listed in Table 6–6.

Table 6–6 Oracle WebCenter Troubleshooting Scenarios

Action Before Failover	After Failover	Suggested Remedy
Go to any page and click **Create Page**.	Type in a name, select a theme, and click **OK**. When you select the theme, the page creation popup is closed.	Repeat the operation.
Launch Manage Pages.	Perform any operation within the popup, except for closing the popup, for example, click **Page Actions**. When you perform any operation, the Manage Pages popup is closed.	Repeat the operation.
Launch Manage Pages, click **Page Actions** against a page, and then the **Delete** option in the menu.	Click **OK** on the confirmation popup. Clicking **OK** not only closes the confirmation popup, but the Manage Pages popup also gets closed as part of the request, and the effect of the deletion (which may have gone through) is not visible among the tabs.	Relaunch the Manage Pages popup to see if the page has been deleted. If not, try deleting once again.
Launch **Personalize Applications** popup. Perform any operation that sends a request, other than clicking **OK**, for example, expand the **Applications** node.	Perform any operation that sends a request, other than clicking **OK**, for example, expand the **Applications** node. When you perform any operation that sends a request to the server, the **Personalize Applications** popup is closed.	Repeat the operation
Launch Preferences.	Switch between the **Preferences** tabs (General, Accounts, Messaging, Search) in step ii. When you switch between the **Preference** tabs, the **Preferences** popup is closed.	Repeat the operation
Launch Manage Favorites ii. Stop the server, perform any operation other than closing the popup, for example. expand a folder, click **Edit favorite information**.	Perform any operation other than closing the popup, for example, expand a folder, and click **Edit favorite information**. When you perform any operation, the **Manage Favorites** popup is closed.	Relaunch the **Manage Favorites** popup to see if the operation was successful. If not, retry the operation.

- **Component Specific Issues** - Other issues which are specific to different components in Oracle WebCenter are listed in Table 6–7.

Table 6–7 Oracle WebCenter Exceptions to Expected Behavior

Oracle WebCenter Component	Exceptions to Expected Behavior
Community Events	When changing certain fields (start or end date/hour/minute), the event form is closed when failover occurs.
Portlets	Failures are fully transparent.
Lists	Failures are fully transparent.
Links	Failures are fully transparent.
Search	In the midst of a long-running query. The results that come back are not guaranteed (note there is no "write" operation here), the user can just reissue the query.
Tagging	Failures are fully transparent.
Recent Activities	The open/close state of the tree nodes is not replicated. After after failover, the tree of results closes all of the nodes. The nodes need to be reopened.
Worklist	Failures are fully transparent.
Doclib	When a user uploads a document and a document with the same name already exists, the user is taken to the Confirm new version screen. While on that screen, the file is stored in a temporary local location. If failover occurs at that moment, the uploaded file is lost and the user must restart the upload process.

6.3.2.8 Monitoring Logging of Application Deployments

Use Oracle WebLogic Server Administration Console to check the status of the application deployments. You can also use Oracle WebLogic Server infrastructure and Enterprise Manager Fusion Middleware Control for starting stopping, and monitoring Oracle WebCenter processes.

6.3.2.9 Oracle WebCenter Cluster-wide Configuration Changes

For Oracle WebCenter Spaces and Portlet Producers, all configuration data is stored in the MDS repository. Additional cluster deployments automatically read the latest configuration upon application startup.

For Oracle Discussion Server, the configuration information should be moved over from an existing cluster server. This is done automatically by the pack/unpack utility of Oracle WebLogic Server. Oracle recommends this procedure.

For Oracle WebCenter Wiki and Blogs Server, most of the configuration is stored in the database, however, there is also configuration specific to custom templates located in the deployment directory. For cluster installations, all Wiki Servers in the cluster should share this deployment directory.

6.3.2.10 Maintaining Configuration in a Clustered Environment

Oracle WebCenter Spaces and Portlet Producers store their configuration in the MDS repository. Any changes made to the configuration of one server in the cluster are immediately visible to all other members.

Changes to Oracle WebCenter Discussion Server are not frequent, however, when they do occur, the changes must be reapplied to other members of the cluster. You can do this by connecting directly to each Discussion Server, instead of using the load balancer, and making the necessary administration changes.

Oracle Wiki Server maintains its configuration in the database, and in the local deployment directory. You need to apply the configuration to each node.

6.4 Configuring High Availability for Oracle WebCenter

Figure 6–6 illustrates a two-node Oracle WebCenter cluster running on two Oracle WebLogic Servers in one WebLogic Server domain. Oracle WebLogic Servers are front-ended by Oracle HTTP Server which load balances incoming requests to them. An Oracle RAC database is used for storing metadata and schemas. Shared storage is used for shared Oracle Wiki and Blog Server configuration. VIPs are used for the administration server (for manual failover).

Note: Oracle strongly recommends reading the release notes for any additional installation and deployment considerations prior to starting the setup process.

Note: All command examples are for k or bash shell. No C shell examples are provided.

6.4.1 Preparing the Environment: Prerequisite Steps Before Setting up an Oracle WebCenter High Availability Configuration

The following sections provide prerequisite steps before setting up an Oracle WebCenter high availability configuration.

For information about platform-specific commands, see the *Oracle Fusion Middleware Installation Guide for Oracle WebCenter*.

6.4.1.1 Database Prerequisites

Oracle WebCenter supports the following database versions:

- Oracle Database 10g (10.2.0.4 or later)
- Oracle Database 11g (11.1.0.7 or later)

To determine the database version, execute the following query:

```
SQL>select version from sys.product_component_version where
product like 'Oracle%';
```

6.4.1.2 VIP and IP Prerequisites

To configure a virtual IP for the administration server, see Section 10.2.2.3, "Transforming the Administration Server for Cold Failover Clusters."

6.4.1.3 Installing and Configuring the Database Repository

This section describes how to install and configure the database repository.

Oracle Clusterware

- For 10*g* Release 2 (10.2), see the *Oracle Database Oracle Clusterware and Oracle Real Application Clusters Installation Guide*.

- For 11*g* Release 1 (11.1), see the *Oracle Clusterware Installation Guide*.

Automatic Storage Management

- For 10*g* Release 2 (10.2), see the *Oracle Database Oracle Clusterware and Oracle Real Application Clusters Installation Guide*.
- For 11*g* Release 1 (11.1), see the *Oracle Real Application Clusters Installation Guide*.
- When you run the installer, select the **Configure Automatic Storage Management** option in the **Select Configuration** page to create a separate Automatic Storage Management home.

Oracle Real Application Clusters

- For 10*g* Release 2 (10.2), see *Oracle Database Oracle Clusterware and Oracle Real Application Clusters Installation Guide*.
- For 11*g* Release 1 (11.1), see *Oracle Real Application Clusters Installation Guide*.

You must install the 11g (11.1.1) Oracle Fusion Middleware repository into a Real Application Cluster database before you install the Oracle Fusion Middleware components. Oracle Fusion Middleware provides a tool, Oracle Fusion Middleware Repository Creation Utility (RCU), to create the component schemas in an existing database.

You install RCU in its own, separate Oracle home.

See *Oracle Fusion Middleware Repository Creation Utility User's Guide* for more information about installing RCU.

Database Initialization Parameters

Ensure that the following initialization parameter is set to the required value. It is checked by Repository Creation Assistant.

Table 6–8 Required Initialization Parameters

Parameter	Required Value	Parameter Class
`PROCESSES`	300 or greater	Static

To check the value of the initialization parameter using SQL*Plus, you can use the SHOW PARAMETER command.

As the SYS user, issue the SHOW PARAMETER command as follows:

```
SQL> SHOW PARAMETER processes
```

Set the initialization parameter using the following command:

```
SQL> ALTER SYSTEM SET processes=300 SCOPE=SPFILE
```

Restart the database.

Note: The method that you use to change a parameter's value depends on whether the parameter is static or dynamic, and on whether your database uses a parameter file or a server parameter file. See the *Oracle Database Administrator's Guide* for details on parameter files, server parameter files, and how to change parameter values.

6.4.1.4 Installing and Configuring an LDAP Provider

For production environments, it is a mandatory requirement for Oracle WebCenter high availability topologies to have an external LDAP policy store. For more information on the supported policy stores, as well as instructions on configuring LDAP, see the *Oracle Fusion Middleware Administrator's Guide for Oracle WebCenter*.

6.4.1.5 Terminology for Directories and Directory Environment Variables

The follow list describes the directories and variables used in this chapter:

- ORACLE_BASE: This environment variable and related directory path refers to the base directory under which Oracle products are installed.
- MW_HOME: This environment variable and related directory path refers to the location where Fusion Middleware (FMW) resides.
- WL_HOME: This environment variable and related directory path contains installed files necessary to host a WebLogic Server.
- ORACLE_HOME: This environment variable and related directory path refers to the location where Oracle WebCenter is installed.
- DOMAIN_HOME: This directory path refers to the location where the Oracle WebLogic Domain information (configuration artifacts) is stored.

The values used and recommended for consistency for this directories are:

ORACLE_BASE:

```
/u01/app/oracle
```

MW HOME (Apptier):

```
ORACLE_BASE/product/fmw
```

WLS_HOME:

```
MW_HOME/wlserver_10.3
```

ORACLE_HOME:

```
MW_HOME/WC1
```

6.4.1.6 Using Oracle Fusion Middleware Repository Creation Utility to Load the Fusion Middleware Schemas in the Database

Install the 11g (11.1.1) Oracle Fusion Middleware Metadata Store and WebCenter schemas into a Real Application Cluster database before you install Oracle Fusion Middleware WebCenter components. Oracle Fusion Middleware provides a tool, Oracle Fusion Middleware Repository Creation Utility (RCU), to create the component schemas in an existing database. See the *Oracle Fusion Middleware Repository Creation Utility User's Guide* for more information about installing RCU.

6.4.1.6.1 Running RCU Run RCU to install the required metadata for Oracle Fusion Middleware 11*g*.

1. Start up RCU using the following command:

   ```
   RCU_HOME/bin/rcu &
   ```

2. In the Welcome screen, click **Next**.

3. In the Create Repository screen, select **Create** to load component schemas into a database, and click **Next**.
4. In the Database Connection Details screen, enter connection information for your database:
   - Database Type: Select **Oracle Database**.
   - Host Name: Enter the name of the node that is running the database. For an Oracle RAC database, specify the VIP name, or one of the node names as the host name: **WCDBHOST1VIRTUAL**.
   - Port: The port number for the database: **1521**
   - Service Name: Enter the service name of the database: **wcha.mycompany.com**
   - Username: **SYS**
   - Password: Enter the password for the SYS user.
   - Role: **SYSDBA**
5. Click **Next**.
6. If you receive the following warning message, click **Ignore** or **Stop**:

   The database you are connecting is with non-UTF8 charset, if you are going to use this DB for multilingual support, you may have data loss. If you are not using for multilingual support you can continue, otherwise we strongly recommend using UTF-8 database.
7. In the Select Components screen, select **Create a New Prefix**, and enter a prefix to use for the database schemas, for example, **WCHA**

   Write down the schema names so they are available in later procedures.
   - Select the following schemas:
     – AS Common schemas:

       Metadata Services

     – WebCenter Infrastructure:

       WebCenter Suite
8. Click **Next**.
9. In the Schema Passwords screen, enter passwords for the main and additional (auxiliary) schema users, and click **Next**.
10. In the Map Tablespaces screen, choose the tablespaces for the selected components, and click **Next**.
11. in the Summary screen, click **Create**.
12. In the Completion Summary screen, click **Close**.

See *Oracle Fusion Middleware Installation Concepts and Repository Creation Utility User's Guide* for more information about using RCU.

6.4.2 Installing Oracle HTTP Server on WebHost1

To install Oracle HTTP Server on WEBHOST1:

1. Verify that the servers meet the following requirements:

- The system, patch, kernel, and other requirements meet the requirements specified in the *Oracle Fusion Middleware Repository Creation Utility User's Guide*.
- This example uses port 7777. If you choose port 7777, ensure that it is not used by any service on the nodes. You can verify this by running the following command:

  Unix:

  ```
  netstat -an | grep 7777
  ```

  Windows:

  ```
  netstat -an | findstr 7777
  ```

If port 7777 is in use, choose another port, or make it available.

2. On UNIX platforms, if the `/etc/oraInst.loc` or `/var/opt/oracle/oraInst.loc` file exists, check that its contents are correct. Specifically, check that the inventory directory is correct, and that you have write permissions for that directory.

   If the `/etc/oraInst.loc` file does not exist, skip this step.
3. Start Oracle Universal Installer for Oracle Fusion Middleware 11g Webtier Utilities CD installation as follows:

   For UNIX, run this command: `./runInstaller`.

   For Windows, double-click **setup.exe**.
4. In the Specify Inventory Directory screen, enter the location for the inventory and the user group, and click **OK**.
5. Execute the root privileged actions as indicated in the dialog, and click **OK**.
6. In the Welcome screen, click **Next**.
7. In the Select Installation Type screen, select I**nstall and Configure**, and click **Next**.
8. In the Prerequisite Checks screen, ensure that all the prerequisites are met, and click **Next**.
9. In the Specify Installation Location screen, set the location to:

   `/u01/app/oracle/product/11.1.1/ohs_1`
10. Click **Next**.
11. In the Configure Components screen:
    - Select **Oracle HTTP Server**.
    - Do not select **Associate Selected Components with WebLogic Domain**.
    - Click **Next**.
12. In the Specify Component Details screen, enter the following values:
    - Instance Home Location: **app/oracle/product/11.1.1/ohs_1/instances/ohs_instance1**
    - Instance Name: **ohs_instance1**
    - OHS Component Name: **ohs1**
13. Click **Next**.

14. In the **Specify Webtier Port Details** screen, do the following:
    - Select **Specify Custom Ports**. If you specify a custom port, select **Specify Ports using Configuration File** and then use the **Browse** function to select the file.
    - Enter Oracle HTTP Server port. For example, enter **7777**.

    Click **Next**.

    Note: For more information on setting ports, refer to *Oracle Fusion Middleware Installation Guide for Oracle SOA Suite*.

    ---
15. Click **Next**.
16. In the Configuration Summary screen, ensure that the selections are correct, and click **Install**.
17. In the Installation Progress screen:

    For UNIX systems, a dialog box appears prompting you to run the `oracleRoot.sh` script. Open a command window and run the script, following the prompts.

    Click **Next**.
18. In the Configuration screen, several configuration assistants are started in succession. When the configuration assistants are finished, the Configuration Completed screen appears.
19. In the Configuration Completed screen, click **Finish** to exit.

6.4.2.1 Validating Oracle HTTP Server

To verify that Oracle HTTP Server is set up correctly, access the root URL context of the server by using the following URL a browser:

WebHost1:7777/

If Oracle HTTP Server is set up correctly, the **Hello World** page appears in the browser.

6.4.3 Installing Oracle Fusion Middleware Home

Use the following procedures to install Oracle Fusion Middleware components:

- Oracle WebLogic Server (see "Section 6.4.3.1, "Installing Oracle WebLogic Server"")
- Oracle WebCenter (see Section 6.4.3.2, "Installing Oracle Fusion Middleware for Oracle WebCenter"")

6.4.3.1 Installing Oracle WebLogic Server

To install Oracle WebLogic Server on all nodes in the application tier:

1. On UNIX platforms, if the `/etc/oraInst.loc` or `/var/opt/oracle/oraInst.loc` file exists, check that its contents are correct. Specifically, check that the inventory directory is correct and that you have write permissions for that directory.

    If the `/etc/oraInst.loc` file does not exist, skip this step.
2. Start Oracle WebLogic Server Installer.

    On UNIX, (Linux used in this example):

```
APPHOST1> server103_linux32.bin
```

On Windows:

```
APPHOST1> server103_win32.exe
```

3. In the Welcome screen, click **Next**.
4. In the Choose Middleware Home Directory screen:
   - Select **Create a New Middleware Home**
   - For the **Middleware Home Directory** field, enter **ORACLE_BASE/product/fmw**
   - Click **Next**.
5. In the Register for Security Updates screen, enter your contact information for security update notifications, and click **Next**.
6. In the Choose Install Type screen, select **Custom**, and click **Next**.
7. In the JDK Selection screen, select only **JROCKIT SDK1.6.0_05**, and click **Next**.
8. In the Choose Product Installation Directories screen, accept the following directory:

   ```
   ORACLE_BASE/product/fmw/wlserver_10.3
   ```

   Click **Next**.
9. In the Installation Summary screen, click **Next**.
10. In the Installation Complete screen, deselect **Run QuickStart**, and click **Done**.

6.4.3.2 Installing Oracle Fusion Middleware for Oracle WebCenter

To install Oracle Fusion Middleware for Oracle WebCenter on all the nodes in the application tier:

1. Start Oracle Fusion Middleware for Oracle Fusion Middleware 11g WebCenter Installer:

   On UNIX, (Linux used in this example):

   ```
   APPHOST1> runInstaller
   ```

   On Windows:

   ```
   APPHOST1> setup.exe
   ```

   When Oracle Fusion Middleware 11g WebCenter Installer prompts you for a **JRE/JDK location** enter Oracle SDK location created in the Oracle WebLogic Server installation, for example, **ORACLE_BASE/product/fmw/jrockit_160_05_R27.6.2-20**.
2. In the Welcome screen, click **Next**.
3. In the Prerequisite Check screen, verify that the checks complete successfully, and click **Next**.
4. In the Specify Installation Location screen:
   - For Middleware Home, enter **ORACLE_BASE/product/fmw**
   - For Oracle Home Directory, enter the directory you want to use, for example, **wc**

Click **Next**.

5. In the Installation Summary screen, click **Install**.
6. In the Installation Complete screen, click **Finish**.

6.4.4 Enabling the Administration Server VIP

For information about configuring virtual IPs for the administration server, see Chapter 10, "Active-Passive Topologies for Oracle Fusion Middleware High Availability."

6.4.5 Running Oracle Fusion Middleware Configuration Wizard on APPHOST1 to Create the WebLogic Server WebCenter Domain

Run Oracle Fusion Middleware Configuration Wizard from the WebCenter directory in the Middleware home to create a domain containing the Administration Server and Oracle WebCenter components. Ensure that the database where you installed the repository is running. For Oracle RAC databases, Oracle recommends having all the instances running.

1. Start Oracle WebLogic Server Configuration Wizard from the *ORACLE_HOME*/common/bin directory using the following command:

   ```
   APPHOST1> ./config.sh
   ```

2. In the Welcome screen, select **Create a New WebLogic Domain**, and click **Next**.
3. In the Select Domain Source screen, select **Generate a domain configured automatically to support the following products**, and select the following products:

   When you select Webcenter Spaces, WSM Policy Manager and Oracle JRF are selected automatically.

   - **Oracle WebCenter Spaces - 11.1.1.0**
   - **Oracle Enterprise Manager - 11.1.1.0**
   - **Oracle Portlet Producers - 11.1.1.0**
   - **Oracle Wiki and Blog Server - 11.1.1.0**
   - **Oracle WebCenter Discussion Server - 11.1.1.0**
   - **Oracle JRF - 11.1.1.0**

   Click **Next**.

4. Enter the **Domain Name**, **Domain Location**, and **Application Location** and click **Next**.
5. In the Configure Administrator Username and Password screen, enter the username and password to be used for the domain's administrator, and click **Next**.
6. In the Configure Server Start Mode and JDK screen, make the following selections:

   - WebLogic Domain Startup Mode: select **Production Mode**
   - JDK Selection: select **JROCKIT SDK1.6.0_05**.

   Click **Next**.

7. In the Configure JDBC Data Sources screen. select **Configure selected component schemas as RAC multi data source schemas in the next pane**.

The Repository Creation Utility creates the necessary schemas in the Oracle database. You provide a custom prefix for these schemas. Table 6–9 lists the data sources, the schemas used and the managed servers to which they are assigned.

a. Ensure that the following data source appears on the screen.

Table 6–9 Oracle WebCenter Data Sources

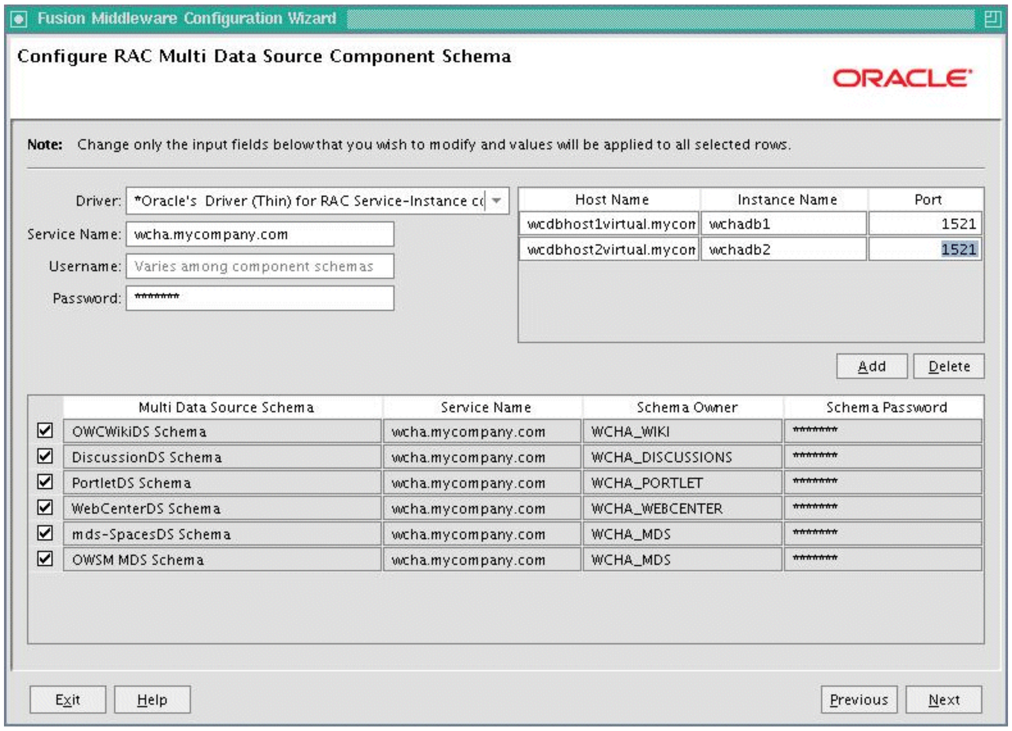
Data Source	Schema Name
OWCWikiDS	Prefix_wiki
DiscussionsDS	Prefix_discussions
PortletDS	Prefix_portlet
WebCenterDS	Prefix_webcenter
mds-SpacesDS	Prefix_mds
OWSM MDS	Prefix_mds

b. Select **Configure all data sources as RAC multi data sources in the next panel**.

c. Click **Next**.

8. In the Configure RAC Multi Data Source Component Schema screen, click the **Add** button to add the **Host name**, **Instance name**, and **Listen port** of both RAC nodes.

Figure 6–7 Configure RAC Multi Data Source Component Schema Screen



Fusion Middleware Configuration Wizard

Configure RAC Multi Data Source Component Schema

Note: Change only the input fields below that you wish to modify and values will be applied to all selected rows.

Driver: *Oracle's Driver (Thin) for RAC Service-Instance c
Service Name: wcha.mycompany.com
Username: Varies among component schemas
Password: ********

Host Name	Instance Name	Port
wcdbhost1virtual.mycon	wchadb1	1521
wcdbhost2virtual.mycon	wchadb2	1521

	Multi Data Source Schema	Service Name	Schema Owner	Schema Password
☑	OWCWikiDS Schema	wcha.mycompany.com	WCHA_WIKI	********
☑	DiscussionDS Schema	wcha.mycompany.com	WCHA_DISCUSSIONS	********
☑	PortletDS Schema	wcha.mycompany.com	WCHA_PORTLET	********
☑	WebCenterDS Schema	wcha.mycompany.com	WCHA_WEBCENTER	********
☑	mds-SpacesDS Schema	wcha.mycompany.com	WCHA_MDS	********
☑	OWSM MDS Schema	wcha.mycompany.com	WCHA_MDS	********

Leave one schema checked and uncheck the other schemas.

a. In the **Driver** drop-down list, select **Oracle's Driver (Thin) for RAC Service-Instance connections**.

b. Enter the **Service name**.

c. Enter the **Prefix_Username** for Oracle WebCenter schemas.

d. Enter the **Password** for the schemas.

e. Click Add to enter the details for the first RAC instance.

f. Update each multi data source schema by selecting one data source at a time and adding the appropriate details.

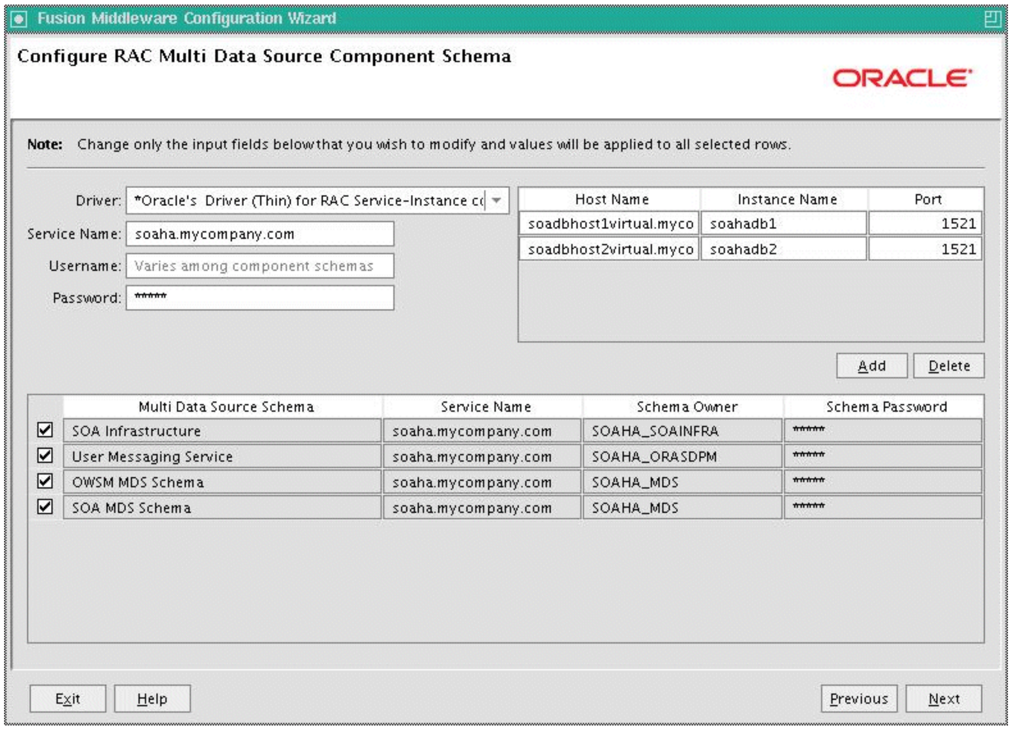
Click **Next**.

9. In the Test JDBC Data Sources screen, the connections are tested automatically. The Status column displays the results. Ensure that all connections were successful. If not, click **Previous** to return to the previous screen and correct your entries.

   Click **Next** when all the connections are successful.

10. In the Select Optional Configuration screen, select the following:

Figure 6–8 Select Optional Configuration Screen



Configure RAC Multi Data Source Component Schema

Note: Change only the input fields below that you wish to modify and values will be applied to all selected rows.

Driver: *Oracle's Driver (Thin) for RAC Service-Instance c
Service Name: soaha.mycompany.com
Username: Varies among component schemas
Password: ******

Host Name	Instance Name	Port
soadbhost1virtual.myco	soahadb1	1521
soadbhost2virtual.myco	soahadb2	1521

	Multi Data Source Schema	Service Name	Schema Owner	Schema Password
☑	SOA Infrastructure	soaha.mycompany.com	SOAHA_SOAINFRA	******
☑	User Messaging Service	soaha.mycompany.com	SOAHA_ORASDPM	******
☑	OWSM MDS Schema	soaha.mycompany.com	SOAHA_MDS	******
☑	SOA MDS Schema	soaha.mycompany.com	SOAHA_MDS	******

- **Administration Server**
- **Managed Servers, Clusters and Machines**

11. In the Customize Server and Cluster Configuration screen, select **Yes**, and click **Next**.

12. In the Configure the Administration Server screen, enter the following values:

    - Name: **AdminServer**
    - Listen Address: Enter the VIP address used in Section 6.4.4.
    - Listen Port: **7001**
    - SSL listen port: N/A
    - SSL enabled: leave unchecked

Click **Next**.

13. In the Configure Managed Servers screen, add the following managed servers:

Table 6–10 Configuring Managed Servers

Name	Listen Address	Listen Port	SSL Listen Port	SSL Enabled
WLS_Portlet	Hostname of APPHOST1	8889	n/a	unchecked
WLS_Portlet2	Hostname of APPHOST2	8889	n/a	unchecked
WLS_Spaces	Hostname of APPHOST1	8888	n/a	unchecked
WLS_Spaces2	Hostname of APPHOST2	8888	n/a	unchecked
WLS_Services	Hostname of APPHOST1	8890	n/a	unchecked
WLS_Services2	Hostname of APPHOST2	8890	n/a	unchecked

Click **Next**.

14. In the Configure Clusters screen, add the following cluster:
    - Name: **Portlet_Cluster**
    - Cluster Messaging Mode: **unicast**
    - Cluster Address Enabled: **leave blank**
    - Name: **Spaces_Cluster**
    - Cluster Messaging Mode: **unicast**

    Note: By default, each Oracle WebCenter cluster is configured as unicast. To configure your Oracle WebCenter cluster for multicast, see *Oracle Fusion Middleware Using Clusters for Oracle WebLogic Server*.

    - Cluster Address Enabled: **leave blank**
    - Name: **Services_Cluster**
    - Cluster Messaging Mode: **unicast**
    - Cluster Address Enabled: **leave blank**

    Click **Next**.

15. In the Assign Servers to Clusters screen, assign the following servers to clusters:
    - **Spaces_Cluster**

      WLS_Spaces

      WLS_Spaces2
    - **Portlet_Cluster**

      WLS_Portlet

      WLS_Portlet2
    - **Services_Cluster**

      WLS_Services

      WLS_Services2

Click **Next**.

16. In the Configure Machines screen:

    - Delete the **LocalMachine** that appears by default.
    - Click the **Unix Machine** tab, and add the following machines:

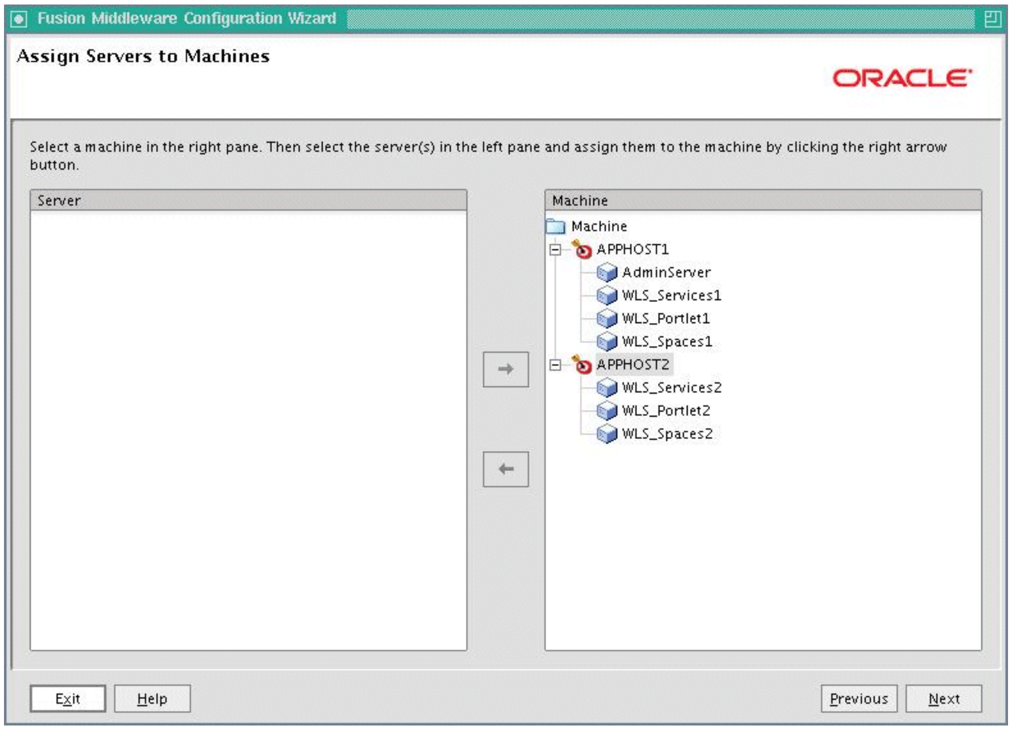
Table 6–11 Configuring Machines

Name	Node Manager Listen Address
APPHOST1	Hostname of APPHOST1
APPHOST2	Hostname of APPHOST2

Click **Next**.

17. In the Assign Servers to Machines screen, assign servers to machines as follows:

Figure 6–9 Assign Servers to Machines Screen



Fusion Middleware Configuration Wizard

Assign Servers to Machines

Select a machine in the right pane. Then select the server(s) in the left pane and assign them to the machine by clicking the right arrow button.

Server

Machine
- Machine
  - APPHOST1
    - AdminServer
    - WLS_Services1
    - WLS_Portlet1
    - WLS_Spaces1
  - APPHOST2
    - WLS_Services2
    - WLS_Portlet2
    - WLS_Spaces2

- APPHOST1: **WLS_Spaces1**, **WLS_Porlet1**, **WLS_Services1**
- APPHOST2: **WLS_Spaces2**, **WLS_Porlet2**, **WLS_Services2**

Click **Next**.

18. In the Configuration Summary screen, click **Create**.

19. In the Creating Domain screen, click **Done**.

6.4.6 Creating boot.properties for the Administration Server and for Managed Servers on APPHOST1

This is an optional step for enabling the Administration Server to start up without prompting you for the administrator username and password. Create a boot.properties file for the Administration Server and for the managed servers on APPHOST1.

For the Administration Server:

1. Create the following directory:

   ```
   APPHOST1> mkdir -p MW_HOME/wls/user_
   projects/domains/wcdomain/servers/AdminServer/security
   ```

2. Use a text editor to create a file called boot.properties in the directory created in the previous step, and enter the following lines in the file:

   ```
   username=adminuser
   password=password
   ```

> **Note:** When you start up the Administration Server, the username and password entries in the file are encrypted.
>
> For security reasons, minimize the time the entries in the file are left unencrypted. After you edit the file, start up the server as soon as possible in order for the entries to be encrypted.

For the WLS_Spaces1 managed server:

1. Create the following directories:

   ```
   APPHOST1> mkdir ORACLE_BASE/product/fmw/user_
   projects/domains/wcdomain/servers/WLS_Spaces1

   APPHOST1> mkdir ORACLE_BASE/product/fmw/user_
   projects/domains/wcdomain/servers/WLS_Spaces1/security
   ```

2. Use a text editor to create a file called boot.properties in the security directory created in the previous step, and enter the following lines in the file:

   ```
   username=adminuser
   password=password
   ```

3. Repeat Steps 2 and 3 for the WLS_Portlet1 and WLS_Services1 Managed Servers on APPHOST1.

> **Note:** When you start up the administration server, the username and password entries in the file are encrypted.
>
> For security reasons, minimize the time the entries in the file are left unencrypted. After you edit the file, start up the server as soon as possible in order for the entries to be encrypted.

6.4.7 Starting the System in APPHOST1

This section describes procedures for starting the system in APPHOST1

6.4.7.1 Starting the Administration Server on APPHOST1

To Start the Administration Server on APPHOST1 run the following commands:

```
APPHOST1> cd ORACLE_BASE/product/fmw/user_projects/domains/wcdomain/bin
```

```
APPHOST1> ./startWebLogic.sh
```

6.4.7.2 Validating the Administration Server

To verify that the Administration Server is properly configured:

1. In a browser, go to `http://VIP1:7001/console`.
2. Log in as the administrator.
3. Verify that the WLS_Spaces1 and WLS_Spaces2 managed servers are listed.
4. Verify that the Spaces_Cluster cluster is listed.
5. Verify that you can access Enterprise Manager at http://VIP1:7001/em.

6.4.7.3 Disabling Host name Verification for the Administration Server and the Managed Servers for APPHOST1 and APPHOST2

This step is required if you have not set up SSL communication between the Administration Server and the Node Manager. If SSL is not set up, you receive an error message unless you disable host name verification.

You can re-enable host name verification when you have set up SSL communication between the Administration Server and the Node Manager.

To disable host name verification on APPHOST1:

1. In Oracle WebLogic Server Administration Console, select **Administration Server**, **SSL**, and then **Advanced**.
2. Click **Lock and Edit**.
3. When prompted, save the changes and activate them.
4. Set **Hostname Verification** to **None**.
5. Select **WLS_Spaces1**, **SSL**, and then **Advanced**.
6. Set **Hostname Verification** to **None**.
7. Repeat Step 3 for WLS_Portlet1 and WLS_Services1.

To disable host name verification on APPHOST2:

1. In Oracle WebLogic Server Administration Console, select **WLS_Spaces2**, **SSL** , and then **Advanced**.
2. Set **Hostname Verification** to **None**.
3. Repeat Steps 1 and 2 for WLS_Portlet2 and WLS_Services2.

6.4.7.4 Starting Node Manager on APPHOST1

Perform these steps to start Node Manager on APPHOST1:

1. Set the JAVA_OPTIONS environment variable to define the `StartScriptEnabled` property before starting Node Manager. On APPHOST1, run the following command:

   ```
   APPHOST1> export JAVA_OPTIONS=-DStartScriptEnabled=true
   ```

   Note: You must use the `StartScriptEnabled` property to avoid class loading failures and other problems.

   ---

2. Start Node Manager:

```
APPHOST1> cd ORACLE_BASE/admin/DOMAIN_NAME/mserver/DOMAIN_NAME/servers

APPHOST1> ./startNodeManager.sh
```

After you have started Node Manager for the first time, you can edit the `nodemanager.properties` file to set the `StartScriptEnabled` property. The `nodemanager.properties` file does not exist until Node Manager is started for the first time.

To set the StartScriptEnabled property in the `nodemanager.properties` file:

- Add the following line to the `ORACLE_BASE/wls/wlserver_10.3/common/nodemanager/nodemanager.properties` file:

  ```
  StartScriptEnabled=true
  ```

When this property is set in the `nodemanager.properties` file, you no longer need to define it in the JAVA_OPTIONS environment variable.

6.4.8 Install WebLogic Server and Oracle WebCenter on APPHOST2

Repeat the procedures for installing WebLogic Server and Oracle WebCenter for APPHOST2, start with Section 6.4.2, "Installing Oracle HTTP Server on WebHost1". The directory paths for binary files and domains used when installing new nodes must be exactly the same as those used for first node. If these paths and domains are not exactly the same as those used for the first node, failover is does not occur.

6.4.9 Propagating the Domain Configuration to APPHOST2 with pack/unpack Utilities

Follow these steps to propagate the domain configuration to APPHOST2 using Pack/Unpack utilities:

1. Run the following pack command on APPHOST1 to create a template pack:

```
APPHOST1> cd ORACLE_BASE/product/fmw/wlserver_10.3/common/bin
APPHOST1> ./pack.sh -managed=true -domain=ORACLE_BASE/product/fmw/user_
projects/domains/wcdomain/
-template=wcdomaintemplate.jar
-template_name=wc_domain_template
```

2. Run the following command on APPHOST1 to copy the template file created in the previous step to APPHOST2 using, in this example, scp:

```
APPHOST1> scp wcdomaintemplate.jar
 APPHOST2:ORACLE_BASE/product/fmw/wlserver_10.3/common/bin
```

3. Run the unpack command on APPHOST2 to unpack the propagated template:

```
APPHOST2> cd
 ORACLE_BASE/product/fmw/wlserver_10.3/common/bin
APPHOST2> ./unpack.sh
 -domain=ORACLE_BASE/product/fmw/user_projects/domains/wcdomain/
 -template=wcdomaintemplate.jar
```

6.4.10 Starting Node Manager on APPHOST2

To start the Node Manager on APPHOST2, repeat the steps from Section 6.4.7.4, "Starting Node Manager on APPHOST1." on APPHOST2.

6.4.11 Configuring Oracle HTTP Server for the Administration Server and Oracle WebCenter Managed Servers

Enable Oracle HTTP Server to route to the Administration Server that contains Oracle WebCenter managed servers, by setting the WebLogicCluster parameter to the list of nodes in the cluster.

1. Add the following lines to the `OHS_HOME/instances/ohs_instance1/config/OHS/ohs1/mod_wl_ohs.conf` file:

```
# Spaces
<Location /webcenter>
    WebLogicCluster apphost1.com:8888,apphost2.com:8889
    SetHandler weblogic-handler
</Location>

<Location /webcenterhelp>
    WebLogicCluster apphost1.com:8888,apphost2.com:8888
    SetHandler weblogic-handler
</Location>

<Location /rss>
    WebLogicCluster apphost1.com:8888,apphost2.com:8888
    SetHandler weblogic-handler
</Location>

# Portlet
<Location /portalTools>
    WebLogicCluster apphost1.com:8889,apphost2.com:8889
    SetHandler weblogic-handler
</Location>

<Location /wsrp-tools>
    WebLogicCluster apphost1.com:8889,apphost2.com:8889
    SetHandler weblogic-handler
</Location>

# Discussions and Wiki
<Location /owc_discussions>
    WebLogicCluster apphost1.com:8890,apphost2.com:8890
    SetHandler weblogic-handler
</Location>

<Location /owc_wiki>
    WebLogicCluster apphost1.com:8890,apphost2.com:8890
    SetHandler weblogic-handler
</Location>
```

2. Restart Oracle HTTP Server on WEBHOST1:

```
WEBHOST1> OHS_HOME/instances/ohs_instance1/bin/opmnctl restartproc
ias-component=OHS_COMPONENT1
```

6.4.11.1 Validating Access through Oracle HTTP Server

Verify the URLS to ensure that appropriate routing and failover is working from the HTTP Server to Oracle WebCenter cluster.

1. Start WLS_Spaces1, WLS_Spaces2, WLS_Portlet1 and WLS_Portlet2 from the WebLogic Server Administration Console as follows:

   a. Access the Administration Console at the following URL

      http://APHHOST1/console

   b. Click **Servers**.
   c. Open the **Control** tab.
   d. Select **WLS_Spaces1**, **WLS_Spaces2**, **WLS_Portlet1** and **WLS_Portlet2**.
   e. Click **Start**.
   f. Verify direct access to the managed servers using the following URLs:

      apphost1:8888/webcenter

      apphost2:8888/webcenter

      apphost1:8889/portaltools

      apphost2:8889/portaltools

2. While WLS_Spaces2 and WLS_Portlet2 are running, stop WLS_Spaces1 and WLS_Portlet1 from Oracle WebLogic Server Administration Console.
3. Access the following URLs and verify the appropriate functionality:
   - WebHost1:7777/webcenter
   - WebHost1:7777/portalTools
4. Start WLS_Spaces1 and WLS_Portlet1 from the WebLogic Server Administration Console.
5. Stop WLS_Spaces2 and WLS_Portlet2.
6. Access the following URLs and verify the appropriate functionality:
   - WebHost1:7777/webcenter
   - WebHost1:7777/portalTools

6.4.12 Configuring Manual Failover of the Administration Server to APPHOST2

For information about configuring the administration server for high availability, see Section 10.2.2.3, "Transforming the Administration Server for Cold Failover Clusters."

6.4.13 Configuring the Java Object Cache

The Java Object Cache (JOC) should be configured among all the servers running WebCenter Spaces. This local cache is provided to increase the performance of Oracle WebCenter Spaces.

The Java Object Cache can be configured using the *MW_HOME/webcenter/scripts/configure-joc.py* script. This is a Python script which can be used to configure JOC in the managed servers. The script runs in WLST online mode and expects Admin Server to be up and running.

> **Note:** After configuring the Java Object Cache using the wlst commands or *configure-joc.py* script, all affected managed servers should be restarted for the configurations to take effect.

Usage

1. Connect to the Administration Server using the command-line Oracle Weblogic Scripting Tool (WLST), for example:

   ```
   MW_HOME/wc/common/bin/wlst.sh
   $ connect()
   ```

   Enter the Oracle Weblogic Administration user name and password when prompted.

2. After connecting to the Administration Server using `wlst`, start the script using the `execfile` command, for example:

   ```
   wls:/mydomain/serverConfig>execfile('MW_HOME/webcenter/scripts/
   configure-joc.py')
   ```

3. All the input parameters are prompted by the script. The script can be used to perform the following JOC configurations:

   a. Configure JOC for all the managed servers for a given cluster.

      Enter 'y' when the script prompts whether you want to specify a cluster name, and also specify the cluster name and discover port, when prompted. This discovers all the managed servers for the given cluster and configure the JOC. The discover port is common for the entire JOC configuration across the cluster. For example:

      ```
      Do you want to specify a cluster name (y/n) <y>
      Enter Cluster Name : Spaces_Cluster
      Enter Discover Port : 9999
      ```

   b. Configure JOC for all specified managed servers.

      Enter 'n' when the script prompts whether you want to specify a cluster name, and also specify the managed server and discover port, when prompted. For example:

      ```
      Do you want to specify a cluster name (y/n) <y>n
      Enter Managed Server and Discover Port (eg WLS_Spaces1:9999, WLS_
      Spaces2:9999) : WLS_Spaces1:9999,WLS_Spaces2:9999
      ```

      This example configures JOC only for the specified managed server (that is, WLS_Spaces1 & WLS_Spaces2). The discover port is specified with the managed server (for example, WLS_Spaces1:2222).

   c. Exclude JOC configuration for some managed servers.

      The script allows you to specify the list of managed servers for which the JOC configuration "DistributeMode" will be set to 'false'. Enter 'y' when the script prompts whether you want to exclude any servers from JOC configuration, and enter the managed server names to be excluded, when prompted. For example:

      ```
      Do you want to exclude any server(s) from JOC configuration (y/n) <n>y
      Exclude Managed Server List (eg Server1,Server2) : WLS_Spaces1,WLS_Spaces3
      ```

   d. Disable the distribution mode for all managed servers.

      The script allows you to disable the distribution to all the managed servers for a specified cluster. Specify 'false' when the script prompts for the distribution mode. By default, the distribution mode is set to 'true'.

6.4.14 Running CacheWatcher to verify Java Object Cache

You can execute a utility called CacheWatcher to verify your java object cache configuration. To execute CachWatcher:

For MW_ORA_HOME below use the location of the Oracle WebCenter installation (FMW_HOME/Oracle_WC1)

1. For MW_ORA_HOME use the location of the Oracle WebCenter installation, for example, *MW_HOME*/Oracle_WC1.
2. For DOMAIN_HOME, use the full path to the location of your domain, under *MW_HOME*/user_projects/domains. Server_Name refers to a managed server name, for example, WLS_Portlet1.
3. On any machine, for example, APPHOST1, execute the following:

```
"java -classpath WLS_HOME/WebCenter_oracle_home/modules/
oracle.javacache_11.1.1/cache.jar:WLS_HOME/WebCenter_oracle_
home/modules/oracle.odl_11.1.1/ojdl.jar oracle.ias.cache.CacheUtil
watch -config=DOMAIN_HOME/config/fmwconfig/servers/Server_Name/javacache.xml"
```

4. On APPHOST2, execute the following:

```
"java -classpath MW_HOME/modules/oracle.javacache_11.1.1/
cache.jar:MW_HOME/modules/oracle.odl_11.1.1/ojdl.jar
oracle.ias.cache.CacheUtil watch -config=absolute_path_of_your_config_file"
```

5. At the CacheWatcher's prompt, type `lc` and then press **Enter**.

   A group view appears.

   The group view indicates the number of cache members. The following is an example of a CacheWatcher run:

```
java -classpath MW_HOME/modules/oracle.javacache_11.1.1/
cache.jar:MW_HOME/modules/oracle.odl_11.1.1/ojdl.jar
oracle.ias.cache.CacheUtil watch -config=DOMAIN_HOME
/config/fmwconfig/servers/WLS_Spaces_Server/javacache.xml"
cache> lc
VID:    2
Size:   2
Column: 0123456789
CL:     JJ
GRP0:   11
...
Member Table:
#1. J57161:145.87.9.15:86AAC2E:11B28D57BE4:-7FFE, 0
#2. J35578:145.87.9.14:-9E92850:11B28D5F7E0:-7FFF, 1
```

   After starting CacheWatcher on each host, find two members from the view.

Note: CacheWatcher itself is considered a distribute cache instance. To terminate the CacheWatcher shell, type `exit` and press **Enter**.

6.4.15 Configuring Oracle WebCenter for Replication

Use the procedures in this section to configure Oracle WebCenter for replication.

Clustering Requirement

The application must be deployed to an Oracle WebLogic Cluster. This automatically establishes a replication channel for the multiple instances of the application.

Note: In a Unicast cluster, the default replication channel is configured using the Listen address of each managed server. Therefore, the Listen address should be configured to be a specific IP address or host name, instead of being configured to listen on `Any`.

Oracle ADF Replication

Oracle WebCenter relies on Oracle ADF components, therefore, essential that Oracle ADF is configured properly. The following tag should be present in the `adf-config.xml` file, one of the Application Resources, for a stateful application:

```
<adfc:adf-controller-config>
<adfc:adf-scope-ha-support>true</adfc:adf-scope-ha-support>
</adfc:adf-controller-config>
```

In Oracle WebCenter applications, this is already been enabled by default.

Application Replication

Applications must also have replication enabled. Oracle WebLogic Server allows several types of persistent stores for replication. For more information on persistent stores, see*Oracle Fusion Middleware Using Clusters for Oracle WebLogic Server*.

For Oracle WebCenter, applications are enabled by default with the following setting in the `weblogic.xml` file:

```
<session-descriptor>
<persistent-store-type>replicated_if_clustered</persistent-store-type>
</session-descriptor>
```

The `replicated_if_clustered` setting disables replication for stand-alone application environments, and uses in-memory replication within a cluster environment.

Ensure that any custom application is configured for in-memory replication.

6.4.16 Scaling the Topology

You can scale out and scale up an Oracle WebCenter topology. When you "scale up" the topology, you add new managed servers to nodes that are already running one or more managed servers. When you "scale out" the topology, you add new managed servers to new nodes.

6.4.16.1 Scaling Up the Topology (Adding Managed Servers to Existing Nodes)

In this case, you already have a node that runs a managed server configured with WebCenter components. The node contains a Middleware home and a WebCenter directory in shared storage.

You can use the existing installations (Middleware home, and domain directories) for creating new Oracle WebCenter and servers. There is no need to install WebCenter binaries in a new location, or to run pack and unpack.

Follow these steps for scaling up the topology:

1. Using the Administration Console, clone WLS_Spaces1 or WLS_Portlet1 or WLS_Services1 into a new managed server. The source managed server to clone should be one that already exists on the node where you want to run the new managed server.

   To clone a managed server:

   a. Select **Environment** -> **Servers** from the Administration Console.

   b. Select the managed server that you want to clone (for example, WLS_Spaces1 or WLS_Portlet1).

   c. Select **Clone**.

   Name the new managed server *SERVER_NAME*n, where n is a number to identify the new managed server.

2. For the listen address, assign the host name or IP to use for this new managed server.

   Ensure the port number for this managed server is availble on this node.

3. If the managed server being scaled out is WLS_Services, then follow these extra steps:

   a. Start up the managed server at least once, to create the managed servers deployment directories for Oracle Wiki. The managed server can then be shut down.

   b. Create a soft link from this new servers config directories to that of the shared Wiki Server directories:

   ```
   $ ln -s DOMAIN_HOME/servers/WLS_ServicesN/stage/owc_wiki/11.1.1.1.0/owc_
   wiki/attachments/ /shared/owc_wiki/attachments
   $ ln -s DOMAIN_HOME/servers/WLS_ServicesN/stage/owc_wiki/11.1.1.1.0/owc_
   wiki/templates/ /shared/owc_wiki/templates
   ```

4. Reconfigure the Oracle HTTP Server module with the new member in the cluster. See Oracle HTTP Server configuration.

6.4.16.2 Scaling Out the Topology (Adding Managed Servers to New Nodes)

In scaling out your topology, you add new managed servers configured with Oracle WebCenter applications to new nodes.

Before performing the steps in this section, check that you meet these requirements:

- In your topology, there are existing nodes running managed servers configured with Oracle WebCenter applications.
- The new node can access the existing home directories for WebLogic Server and Oracle WebCenter. You use the existing installations in shared storage for creating a new managed server. There is no need to install WebLogic Server or WebCenter binaries in a new location, or to run pack and unpack.

Follow these steps for scaling out the topology:

1. On the new node, mount the existing Middleware home, which includes the WebCenter installation and the domain directory, and ensure that the new node has access to this directory, just like the rest of the nodes in the domain.
2. Create a new machine for the new node that will be used, and add the machine to the domain.

3. Update the machine's Node Manager's address to map the IP of the node that is being used for scale out.
4. Using the Administration Console, clone WLS_Spaces1/WLS_Portlet1/WLS_Services1 into a new managed server and name it WLS_(SERVER_TYPE)*n*, where *n* is a number.

   The steps assume that you are adding a new server to node n, where no managed server was running.
5. For the listen address of the managed server, assign the host name or IP to use for the new managed server.
6. If the managed server being scaled out is WLS_Services then follow these extra steps:
   a. Start up the managed server at least once, to create the managed servers deployment directories for Oracle Wiki. The managed server can then be shut down.
   b. Create a soft link from this new servers config directories to that of the shared Wiki Server directories:

      ```
      $ ln -s DOMAIN_HOME/servers/WLS_ServicesN/stage/owc_wiki/11.1.1.1.0/owc_
      wiki/attachments/ /shared/owc_wiki/attachments
      $ ln -s DOMAIN_HOME/servers/WLS_ServicesN/stage/owc_wiki/11.1.1.1.0/owc_
      wiki/templates/ /shared/owc_wiki/templates
      ```
7. Reconfigure the Oracle HTTP Server module with the new member in the cluster (see Oracle HTTP Server configuration).
8. Start Node Manager on the new node: using the installation in shared storage from the already existing nodes, start Node Manager passing as parameter the host name of the new node:

   ```
   NEW_NODE> MW_HOME/wlserver_10.3/server/bin/startNodeManager <new_node_ip>
   ```

   If you used the paths shown in Section 6.4.3.1, "Installing Oracle WebLogic Server," *MW_HOME* would be `ORACLE_BASE/product/fmw`.
9. Start and test the just added managed server from the Administration Console.

   Access the application on the newly created managed server (http://HOST:port/webcenter). The application should be functional.
10. Add the New managed server to the Java Object Cache Cluster (see Section 6.4.13, "Configuring the Java Object Cache").

6.4.17 Troubleshooting Oracle WebCenter High Availability

This section describes procedures for troubleshooting possible issues with Oracle WebCenter.

6.4.17.1 Troubleshooting Oracle WebCenter Deployment Issues

Oracle WebCenter Applications are deployed when the managed server is first started. Use Oracle WebLogic Server Administration Console first to check that all application deployments were successful:

Click **Deployments** in the left hand pane. The right hand pane shows the application deployments and their status. The state of all applications, assuming all the servers are running, should be ACTIVE.

If an application deployment has failed, the server logs may provide some indication of why the application was not deployed successfully. The server logs are located in the `DOMAIN_HOME/servers/{SERVER_NAME}/logs` directory. Common issues include:

- Unavailability of external resources, such as database resources. Examine the error, fix it, and attempt to redeploy the application.
- The appropriate applications or libraries are not targeted correctly to the right managed server or Cluster.

6.4.17.2 Troubleshooting Oracle WebCenter Replication and Failover Issues

State Replication is most prominent in failover scenarios. A user working on one server may discover that, upon failover:

- Windows may be closed or state might be reset.
- Screens may require a reset.
- The application may be redirecting to the logon screen.

The following steps provide guidance in troubleshooting and diagnosing state replication issues.

1. Confirm that this is not a known replication issue.

   See Section 6.3.2.7, "Expected Behavior for Application Failover" for possible expected behaviors. Before proceeding to further diagnose the issue, first confirm that the failover behavior is not an expected behavior.

2. Check load balancer settings.

   For replication and failover to function correctly, the load balancer must be configured with the appropriate persistence settings. For more details on configuring Hardware Load Balancers for Oracle WebLogic Server, see *Oracle Fusion Middleware Using Clusters for Oracle WebLogic Server*.

3. Check the cluster status.

   Replication occurs within the context of a cluster. For failover to be successful, there must be at least one other healthy member of the cluster available. You can check luster status in one of two ways:

   - Check the cluster status using Oracle WebLogic Server Administration Console - In the Left-hand pane, click on **Servers**. Verify the state of all servers in the cluster.
   - Check the cluster status using weblogic.Admin utility The weblogic.Admin command can be used to query the state of all servers in a specific cluster. For example:

     ```
     $ java weblogic.Admin -url Adminhost:7001 -username <username> -password
      <password>  CLUSTERSTATE -clustername Spaces_Cluster
     ```

     This example returns:

     ```
     There are 2 server(s) in cluster: Spaces_Cluster
     The alive servers and their respective states are listed below:
     WLS_Spaces---RUNNING
     WLS_Spaces2---RUNNING
     ```

4. Check cluster communications.

Although Cluster members may all be running, there may be communication issues which prevent them from communicating replication information to each other. There are two types of cluster communication configurations. Troubleshooting depends on the cluster type:

- Checking Unicast cluster communications - For Unicast clusters, managed servers must be able to access each other's hosts and each other's default listening port.

  Ensure that all individual managed servers have their Listen Address set correctly. You can find this setting by selecting **Configuration**, **General** for each managed server.

- Checking Multicast cluster communications - For multicast clusters, servers must be able to intercept the same multicast traffic. Ensure that multicast is configured correctly by running the WebLogic utility `utils.MulticastTest` on each machine. For example:

  ```
  $ java utils.MulticastTest -H
  ```

5. Confirm application configuration.

   Oracle WebCenter applications are configured correctly by default. For user applications, in-memory replication take place only with the proper configuration in `weblogic.xml`:

   ```
   <session-descriptor>
   <persistent-store-type>replicated_if_clustered</persistent-store-type>
   </session-descriptor>
   ```

   A persistent-store-type of replicated is also acceptable.

6. Enable Debug.

   Check the server logs for any unusual messages on managed server startup. In particular, if the managed server is unable to locate other members of the cluster. The server logs are located in the `DOMAIN_HOME/servers/{SERVER_NAME}/logs` directory.

   For further debugging, enable the flags `DebugCluster`, `DebugClusterAnnouncements`, `DebugFailOver`, `DebugReplication`, and `DebugReplicationDetails`. Each flag can be enabled with the weblogic.Admin utility:

   ```
   $ java weblogic.Admin -url Adminhost:7001 -username <username> -password
    <password> SET -type ServerDebug -property DebugCluster true
   ```

6.4.17.3 Troubleshooting Lost Changes to Policies

Problem: Policies are refreshed at an interval defined in `jps-config.xml` by the variable `oracle.security.jps.ldap.policystore.refresh.interval`. By default, this interval is ten minutes. If a server is lost, any recent policy changes that occurred since the last refresh may appear to be lost. For example, roles created immediately before the failover may be seen as unavailable.

Solution: The policies are still in the back-end policy store but the cache needs to be refreshed for the policy information to be restored. Re-try after a few minutes or login and logout of the application to force a policy cache refresh.

6.5 Configuring High Availability for Custom Oracle ADF and WebCenter Applications

This section describes procedures for configuring custom Oracle ADF and WebCenter applications for high availability

6.5.1 Configuring a Custom Application Cluster

The following are the necessary steps for configuring a cluster for deploying Oracle WebCenter Custom Applications;

1. Access Oracle WebLogic Server Administration Console at `http://server_name:7001/console`.
2. Type the admin username and password to login.
3. On the **Home** page, click **Servers** from the **Domain Configuration** section.
4. On the Summary of the page section, click **New**.
5. Type in your own server name for your Custom App Server, and a port number, for example **8898**.
6. Leave other selections as default, and click **Finish** to generate the managed server.
7. From the **Creation Summary** page, click the server you just created. Click the **Machine** drop-down and select **LocalMachine**.
8. Click **Save** at the bottom of the page.
9. Repeat steps 4 to 8 to create more managed servers as required.
10. To create a cluster, on the **Home** page, click **Clusters**.
11. On the right-hand pane, click **New** to create a new cluster.
12. Give the cluster a name and leave the default of **Unicast**.
13. Click **OK** to create the cluster.
14. Click the cluster name, then click the **Servers** tab.
15. Click the **Add** button to add servers.
16. From the drop-down list, select the previously created managed servers, and then click **Finish**.

6.5.2 Provisioning the Custom Application Cluster

Once you have created the cluster, provision it with the correct libraries, applications, and data sources.

1. Click the **Deployment** link from the **Domain Structure** section.
2. Notice several shared libraries are deployed, make sure the following libraries are targeted to the newly created cluster:
   a. Click the library link.
   b. Click the **Target** tab on top.
   c. Click the check box of the newly created cluster.
   d. Click the **Save** button.

   The shared library lists to set the target to are:

- adf.oracle.domain(1.0,11.1.1.0.0)
- adf.oracle.domain.webapp(1.0,11.1.1.1.0)
- jsf(1.2,1.2.9.1)
- jstl(1.2,1.2.0.1)
- ohw-rcf(5,5.0)
- ohw-uix(5,5.0)
- UIX(11,11.1.1.1.0)
- oracle.adf.dconfigbeans(1.0,11.1.1.0.0)
- oracle.dconfig-infra
- oracle.jrf.system.filter
- oracle.jsp.next(11.1.1,11.1.1)
- oracle.sdp.client(11.1.1,11.1.1)
- oracle.soa.workflow.wc(11.1.1,11.1.1)
- oracle.webcenter.framework(11.1.1,11.1.1)
- oracle.webcenter.framework.view(11.1.1,11.1.1)
- oracle.webcenter.skin(11.1.1,11.1.1)
- oracle.wsm.seedpolicies(11.1.1,11.1.1)
- oracle.portlet-producer.jpdk(11.1.1,11.1.1)
- oracle.portlet-producer.wsrp(11.1.1,11.1.1)

Set the target for each one of these libraries to target to the new managed server.

In addition, set the target to deploy to the Cluster for the following applications:

- DMS Application

3. From left pane of Oracle WebLogic Server Administration Console, click **Services**, **JDBC**, and then **Data Sources**. Make sure the following data sources are targeted to the newly created Cluster:

   - mds-OWSM

   Note: If you created your own MDS schema for your custom applications, assign the new MDS schema to the new Cluster, and omit the two pre-defined MDS schemas.

4. From the left pane of Oracle WebLogic Server Administration Console, click **Environment**, and then **Startup & Shutdown classes**. The following list of classes become available:

   - Audit Loader Startup Class
   - DMS-Startup
   - JMX Framework Startup Class
   - JOC-Startup
   - JPS-Startup Class

- JRF Startup Class
- ODL-Startup
- OWSM Startup Class

Target all those classes to the new Cluster by clicking:

The name of each startup/shutdown class, **Target** tab, the **CustomManagedServer** checkbox, and then click **Save**.

When you complete these steps, you can start the managed server using Oracle WebLogic Server Administration Console and you can now deploy custom applications. Deploy custom applications to the Cluster target, not the managed server target.

5. Target all those classes to the new Cluster by clicking the name of each startup/shutdown class, the **Target** tab, and the **CustomManagedServer** checkbox.
6. Click **Save**
7. Activate the changes.

When you complete these steps, you can start the managed server using Oracle WebLogic Server Administration Console and you can now deploy custom applications. Deploy custom applications to the Cluster target, not the managed server target.

To continue configuring the WebCenter high availability deployment, see the *Oracle Fusion Middleware Administrator's Guide for Oracle WebCenter*.

7 Configuring High Availability for Identity Management Components

The Oracle Identity Management products enable you to configure and manage the identities of users, devices, and services across diverse servers, to delegate administration of these identities, and to provide end users with self-service privileges. These products also enable you to configure single sign-on across applications and to process users' credentials to ensure that only users with valid credentials can log into and access online resources.

It is critical to configure high availability for the Oracle Identity Management products because other enterprise-level applications depend on them. If the Oracle Identity Management products fail, applications depending on them will also fail.

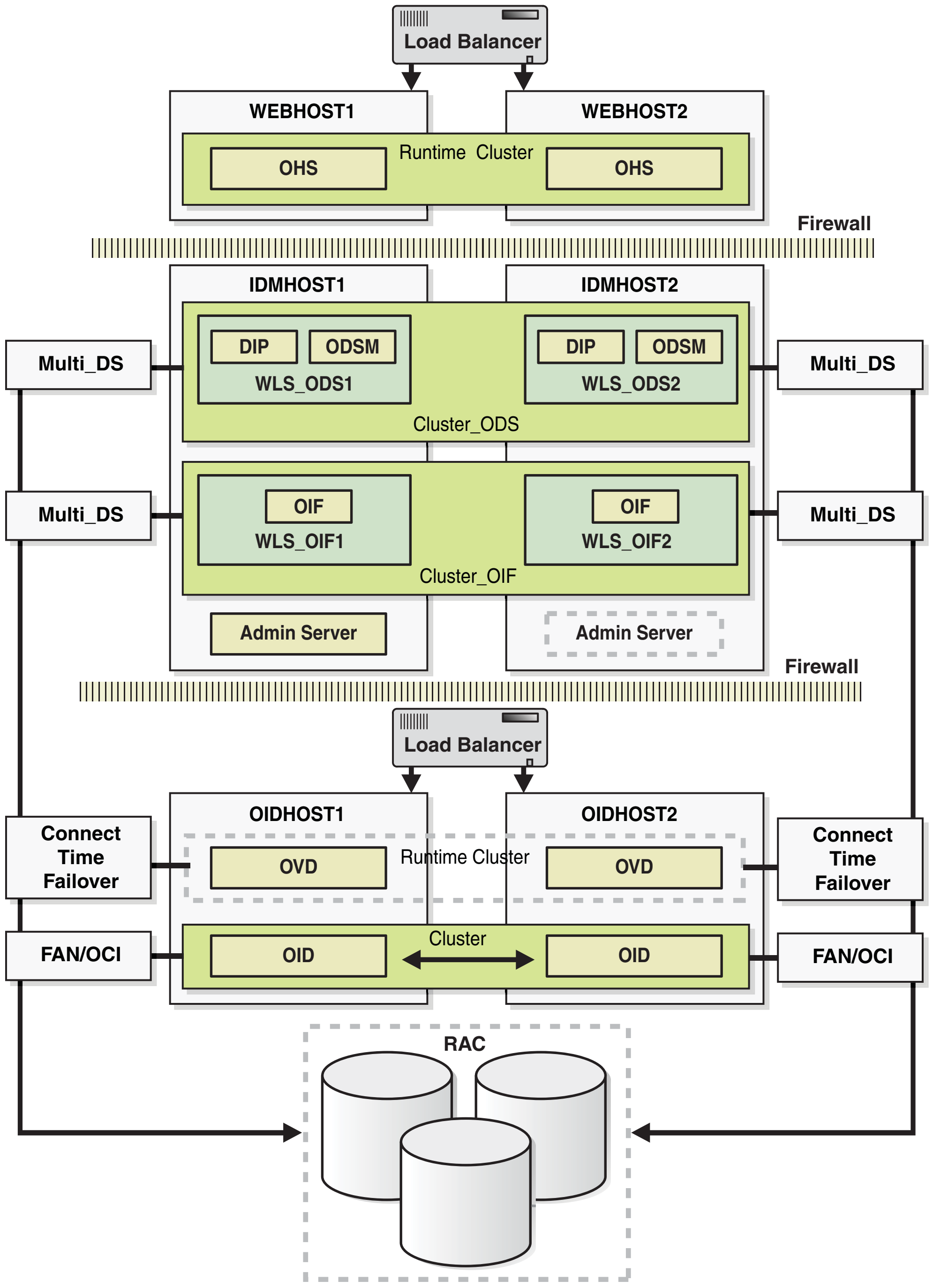
This chapter discusses configuring the Identity Management products for high availability in an active-active configuration.

This chapter includes the following topics:

7.1 Identity Management Product Components and High Availability Concepts

Figure 7–1 shows the Oracle Fusion Middleware 11*g* Oracle Identity Management high availability architecture. This architecture includes a web tier, application tier, and directory tier.

Figure 7–1 Oracle Fusion Middleware 11g Oracle Identity Management High Availability Architecture



In Figure 7–1, the web tier includes the WEBHOST1 and WEBHOST2 computers.

An Oracle HTTP Server instance is installed on WEBHOST1, and an Oracle HTTP Server instance is installed on WEBHOST2. A load balancing router routes requests to the Oracle HTTP Server instances on WEBHOST1 and WEBHOST2.

The application tier includes the IDMHOST1 and IDMHOST2 computers.

On IDMHOST1, the following installations have been performed:

- An Oracle Directory Services Manager instance and an Oracle Directory Integration Platform instance have been installed in the WLS_ODS1 Managed Server. The RAC database has been configured in a JDBC multi data source to protect the instances from RAC node failure.
- An Oracle Identity Federation instance has been installed in the WLS_OIF1 Managed Server. The RAC database has been configured in a JDBC multi data source to protect the instance from RAC node failure.
- A WebLogic Administration Server has been installed. Under normal operations, this is the active Administration Server. The Administration Server is a singleton application.

On IDMHOST2, the following installations have been performed:

- An Oracle Directory Services Manager instance and an Oracle Directory Integration Platform instance have been installed in the WLS_ODS2 Managed Server. The RAC database has been configured in a JDBC multi data source to protect the instances from RAC node failure.

  The instances in the WLS_ODS2 Managed Server on IDMHOST2 and the instances in the WLS_ODS1 Managed Server on IDMHOST1 are configured as the CLUSTER_ODS cluster.
- An Oracle Identity Federation instance has been installed in the WLS_OIF2 Managed Server. The RAC database has been configured in a JDBC multi data source to protect the instance from RAC node failure.

  This Oracle Identity Federation instance and the one in the WLS_OIF1 Managed Server on IDMHOST1 are configured as the CLUSTER_OIF cluster.
- A WebLogic Administration Server has been installed. Under normal operations, this is the passive Administration Server. You will make this Administration Server active if the Administration Server on IDMHOST1 becomes unavailable.

The directory tier includes the OIDHOST1 and OIDHOST2 computers.

On OIDHOST1, an Oracle Internet Directory instance and an Oracle Virtual Directory instance have been installed. Transparent Application Failover (TAF) is used to connect the Oracle Internet Directory instance with the RAC database that serves as the security metadata repository. The database is enabled for server-side TAF and HA Events Notification.

On OIDHOST2, an Oracle Internet Directory instance and an Oracle Virtual Directory instance have been installed. Transparent Application Failover (TAF) is used to connect the Oracle Internet Directory instance with the RAC database that serves as the security metadata repository. The database is enabled for server-side TAF and HA Events Notification.

The Oracle Internet Directory instances on OIDHOST1 and OIDHOST2 have been configured as a cluster. The Oracle Virtual Directory instances on OIDHOST1 and OIDHOST2 have also been configured as a cluster.

Note: It is possible to configure 10*g* Oracle Single Sign-On and 10g Delegated Administration Services with 11*g* Oracle Internet Directory. See Section 8.3, "Setting up Multimaster Replication" for more information.

7.1.1 About the 11*g* Oracle Identity Management Products

Table 7–1 summarizes the Oracle Identity Management products that you can install using the suite-level installation program for 11*g*. See the introductory chapter of the *Oracle Fusion Middleware Quick Installation Guide for Oracle Identity Management* for details:

Table 7–1 The 11g Identity and Access Management Product Suite

Product	Description
Oracle Internet Directory	Oracle Internet Directory is an LDAP Version 3-enabled service that enables fast retrieval and centralized management of information about dispersed users, network configuration, and other resources.
Oracle Virtual Directory	Oracle Virtual Directory is an LDAP Version 3-enabled service that provides an abstracted view of one or more enterprise data sources. Oracle Virtual Directory consolidates multiple data sources into a single directory view, enabling you to integrate LDAP-aware applications with diverse directory server data stores.
Oracle Directory Integration Platform	The Oracle Directory Integration Platform is a J2EE application that enables you to synchronize data between various directories and Oracle Internet Directory. Oracle Directory Integration Platform includes services and interfaces that allow you to deploy synchronization solutions with other enterprise repositories. It can also be used to provide Oracle Internet Directory interoperability with third party metadirectory solutions.
Oracle Identity Federation	Oracle Identity Federation enables companies to provide services and share identities across their respective security domains, while providing protection from unauthorized access.
Oracle Directory Services Manager	Oracle Directory Services Manager is a unified graphical user interface (GUI) for Oracle Virtual Directory and Oracle Internet Directory. Oracle Directory Services Manager simplifies the administration and configuration of Oracle Virtual Directory and Oracle Internet Directory by allowing you to use web-based forms and templates. Oracle Directory Services Manager is available from either the Oracle Enterprise Manager Fusion Middleware Control or from its own URL.

7.2 Prerequisites for Oracle Identity Management High Availability Configuration

This section describes the prerequisite steps that must be completed before setting up an Oracle Identity Management high availability configuration.

7.2.1 Database Prerequisites

This section describes the database prerequisites.

Database versions supported

- For Oracle Database 10g Release 2 (10.2.x), 10.2.0.4 and higher is supported.
- For Oracle Database 11g Release 1 (11.1.x), 11.1.0.7 and higher is supported.

To determine the database version, execute this query:

```
SQL>select version from sys.product_component_version where
product like 'Oracle%';
```

7.2.2 Installing and Configuring the Database Repository

This section provides links to instructions for installing and configuring a database repository.

Oracle Clusterware

- For 10*g* Release 2 (10.2), see the *Oracle Database Oracle Clusterware and Oracle Real Application Clusters Installation Guide*.
- For 11*g* Release 1 (11.1), see *Oracle Clusterware Installation Guide*.

Automatic Storage Management

- For 10*g* Release 2 (10.2), see *Oracle Database Oracle Clusterware and Oracle Real Application Clusters Installation Guide*.
- For 11*g* Release 1 (11.1), see *Oracle Clusterware Installation Guide*.
- When you run the installer, select the **Configure Automatic Storage Management** option in the **Select Configuration** page to create a separate Automatic Storage Management home.

Oracle Real Application Clusters

- For 10*g* Release 2 (10.2), see *Oracle Database Oracle Clusterware and Oracle Real Application Clusters Installation Guide*.
- For 11*g* Release 1 (11.1), see *Oracle Real Application Clusters Installation Guide*.

Many of the Oracle Fusion Middleware components require the existence of schemas in a database prior to installation. Oracle Fusion Middleware provides a tool, the Repository Creation Utility (RCU), to create the component schemas in an existing database. For high availability environments, these schemas must be created and loaded into a Real Application Clusters (RAC) database.

See the next section for information on using RCU to load the Oracle Identity Management schemas into a RAC database repository. This is a required step before you install the Oracle Identity Management components that are used in the high availability configurations described in this chapter.

7.2.3 Installing the Repository Creation Utility Software

The Repository Creation Utility (RCU) ships on its own CD as part of the Oracle Fusion Middleware 11*g* kit. You can use the RCU CD to run RCU and install schemas in a database repository.

Before you install any of the Oracle Identity Management components described in this chapter, run RCU to create the schemas used by Oracle Identity Management and Management Services into a RAC database. These schemas are required for the high availability Oracle Identity Management configurations described in this chapter. For detailed instructions about using RCU to install the required Oracle Internet Directory schemas, see Section 7.3.2.3.2, "Using RCU to Create Oracle Internet Directory Schemas in the Repository."

If you will be installing Oracle Identity Federation, you must also run RCU to create the schemas used by Oracle Identity Federation into a RAC database. For detailed instructions about using RCU to install the required Oracle Identity Federation schemas, see Section 7.7.2.3.1, "Using RCU to Create Oracle Identity Federation Schemas in the Repository."

When you install an Oracle Fusion Middleware product, RCU is installed in the Oracle home directory for that product and you will then be able to run RCU from that Oracle home.

For addition information about running and installing RCU, see *Oracle Fusion Middleware Installation Planning Guide* and *Oracle Fusion Middleware Repository Creation Utility User's Guide*.

7.2.4 Configuring the Database for Oracle Fusion Middleware 11g Metadata

This section describes how to configure the database for Oracle Fusion Middleware 11g metadata.

7.2.4.1 Initialization Parameters

The value of the static PROCESSES initialization parameter must be 500 or greater for Oracle Internet Directory. This value is checked by the Repository Creation Utility.

To check the value, you can use the SHOW PARAMETER command in SQL*Plus:

```
prompt> sqlplus "sys/password as sysdba"
SQL> SHOW PARAMETER processes
```

One common method of changing the parameter value is to use a command similar to the following and then stop and restart the database to make the parameter take effect:

```
prompt> sqlplus "sys/password as sysdba"
SQL> ALTER SYSTEM SET PROCESSES=500 SCOPE=SPFILE;
```

The method that you use to change a parameter's value depends on whether the parameter is static or dynamic, and on whether your database uses a parameter file or a server parameter file. See the *Oracle Database Administrator's Guide* for details on parameter files, server parameter files, and how to change parameter values.

7.2.4.2 Database Examples in This Chapter

Table 7–2 shows the values used for database configuration examples in this chapter.

Table 7–2 Values Used for Database Configuration Examples in This Chapter

Parameter	Value
DB_NAME	idmdb
INSTANCE_NAMES	idmdb1, idmdb2
SERVICE_NAME	idmedg.mycompany.com

7.2.4.3 Database Services

Oracle recommends using the Oracle Enterprise Manager Cluster Managed Services Page to create database services that client applications will use to connect to the database. For complete instructions on creating database services, see the chapter on Workload Management in the *Oracle Real Application Clusters Administration and Deployment Guide*.

You can also use SQL*Plus to configure your RAC database to automate failover for Oracle Internet Directory using the following instructions. Note that each of the following commands only has to be run on one node in the cluster:

1. Use the CREATE_SERVICE subprogram to both create the database service and enable high availability notification and configure server-side Transparent Application Failover (TAF) settings:

   ```
   prompt> sqlplus "sys/password as sysdba"

   SQL> EXECUTE DBMS_SERVICE.CREATE_SERVICE
   (SERVICE_NAME => 'idmedg.mycompany.com',
   NETWORK_NAME => 'idmedg.mycompany.com',
   AQ_HA_NOTIFICATIONS => TRUE,
   FAILOVER_METHOD => DBMS_SERVICE.FAILOVER_METHOD_BASIC,
   FAILOVER_TYPE => DBMS_SERVICE.FAILOVER_TYPE_SELECT,
   FAILOVER_RETRIES => 5, FAILOVER_DELAY => 5);
   ```

   The EXECUTE DBMS_SERVICE command above must be entered on a single line to execute properly.

2. Add the service to the database and assign it to the instances using `srvctl`:

   ```
   prompt> srvctl add service -d idmdb -s idmedg -r idmdb1,idmdb2
   ```

3. Start the service using `srvctl`:

   ```
   prompt> srvctl start service -d idmdb -s  idmedg
   ```

Note: For more information about the SRVCTL command, see the *Oracle Real Application Clusters Administration and Deployment Guide*.

If you already have a service created in the database, make sure that it is enabled for high availability notifications and configured with the proper server-side Transparent Application Failover (TAF) settings. Use the DBMS_SERVICE package to modify the service to enable high availability notification to be sent through Advanced Queuing (AQ) by setting the AQ_HA_NOTIFICATIONS attribute to TRUE and configure server-side Transparent Application Failover (TAF) settings, as shown below:

```
prompt> sqlplus "sys/password as sysdba"

SQL> EXECUTE DBMS_SERVICE.MODIFY_SERVICE
(SERVICE_NAME => 'idmedg.mycompany.com',
AQ_HA_NOTIFICATIONS => TRUE,
FAILOVER_METHOD => DBMS_SERVICE.FAILOVER_METHOD_BASIC,
FAILOVER_TYPE => DBMS_SERVICE.FAILOVER_TYPE_SELECT,
FAILOVER_RETRIES => 5, FAILOVER_DELAY => 5);
```

The EXECUTE DBMS_SERVICE command above must be entered on a single line to execute properly.

Note: For more information about the DBMS_SERVICE package, see the *Oracle Database PL/SQL Packages and Types Reference*.

7.2.4.4 Verifying Transparent Application Failover (TAF)

This section describes how to validate the Transparent Application Failover (TAF) configuration settings made earlier.

After the Oracle Internet Directory process has been started, you can query the FAILOVER_TYPE, FAILOVER_METHOD, and FAILED_OVER columns in the V$SESSION_VIEW to obtain information about connected clients and their TAF status.

For example, use the following SQL statement to verify that TAF is correctly configured:

```
SELECT MACHINE, FAILOVER_TYPE, FAILOVER_METHOD, FAILED_OVER, COUNT(*)
FROM V$SESSION
GROUP BY MACHINE, FAILOVER_TYPE, FAILOVER_METHOD, FAILED_OVER;
```

The output before failover is similar to this:

```
MACHINE              FAILOVER_TYPE FAILOVER_M  FAI   COUNT(*)
-------------------- ------------- ---------- ---- ----------
oidhost1             SELECT        BASIC       NO          11
oidhost1             SELECT        BASIC       NO           1
```

The output after failover is similar to this:

```
MACHINE              FAILOVER_TYPE FAILOVER_M  FAI   COUNT(*)
-------------------- ------------- ---------- ---- ----------
oidhost2             SELECT        BASIC       NO          11
oidhost2             SELECT        BASIC       NO           1
```

7.2.4.5 Configuring Virtual Server Names and Ports for the Load Balancer

This section describes the network prerequisites for deploying an Oracle Identity Management high availability environment.

7.2.4.5.1 Load Balancers All components in the Oracle Identity Management software stack require a hardware load balancer when deployed in a high availability configuration. The hardware load balancer should have the following features:

- Ability to load-balance traffic to a pool of real servers through a virtual host name:

  Clients access services using the virtual host name (instead of using actual host names). The load balancer can then load balance requests to the servers in the pool.

- Port translation configuration
- Monitoring of ports (HTTP, HTTPS, LDAP, LDAPS)
- Virtual servers and port configuration

  Ability to configure virtual server names and ports on your external load balancer, and the virtual server names and ports must meet the following requirements:

  – The load balancer should allow configuration of multiple virtual servers. For each virtual server, the load balancer should allow configuration of traffic management on more than one port. For example, for Oracle Internet Directory clusters, the load balancer needs to be configured with a virtual server and ports for LDAP and LDAPS traffic.

  – The virtual server names must be associated with IP addresses and be part of your DNS. Clients must be able to access the load balancer through the virtual server names.

- Ability to detect node failures and immediately stop routing traffic to the failed node
- Resource monitoring / port monitoring / process failure detection

The load balancer must be able to detect service and node failures (through notification or some other means) and to stop directing non-Oracle Net traffic to the failed node. If your load balancer has the ability to automatically detect failures, you should use it.

- Fault-tolerant mode

  It is highly recommended that you configure the load balancer to be in fault-tolerant mode.

- Other

  It is highly recommended that you configure the load balancer virtual server to return immediately to the calling client when the back-end services to which it forwards traffic are unavailable. This is preferred over the client disconnecting on its own after a timeout based on the TCP/IP settings on the client machine.

- Sticky routing capability

  Ability to maintain sticky connections to components based on cookies or URL.

- SSL acceleration

  This feature is recommended, but not required.

Table 7–3 shows the virtual server names to use for the external load balancer in the Oracle Identity Management high availability environment.

Table 7–3 Virtual Server Names for the External Load Balancer

Component	Virtual Server Name
Oracle Internet Directory	oid.mycompany.com
Oracle Virtual Directory	ovd.mycompany.com
Oracle Identity Federation	oif.mycompany.com
WebLogic Server Administration Console	admin.mycompany.com
Oracle Enterprise Manager Fusion Middleware Control	admin.mycompany.com
Oracle Directory Services Manager Console	admin.mycompany.com

7.2.4.5.2 Virtual Server Names This section describes the virtual server names that should be set up for the high availability deployments described in this chapter.

oid.mycompany.com

This virtual server acts as the access point for all LDAP traffic to the Oracle Internet Directory servers in the directory tier. Traffic to both the SSL and non-SSL ports is configured. The clients access this service using the address oid.mycompany.com:636 for SSL and oid.mycompany.com:389 for non-SSL.

> **Note:** Oracle recommends that you configure the same LDAP port for SSL connections on the virtual server and on the computers on which Oracle Internet Directory is installed.
>
> This is a requirement for most 10*g* Oracle Fusion Middleware products that need to use Oracle Internet Directory via the load balancer.

Monitor the heartbeat of the Oracle Internet Directory processes on OIDHOST1 and OIDHOST2. If an Oracle Internet Directory process stops on OIDHOST1 or OIDHOST2, or if either host OIDHOST1 or OIDHOST2 is down, the load balancer must continue to route the LDAP traffic to the surviving computer.

ovd.mycompany.com

This virtual server acts as the access point for all LDAP traffic to the Oracle Virtual Directory servers in the directory tier. Traffic to both the SSL and non-SSL port is configured. The clients access this service using the address ovd.mycompany.com:7501 for SSL and ovd.mycompany.com:6501 for non-SSL.

Monitor the heartbeat of the Oracle Virtual Directory processes on OVDHOST1 and OVDHOST2. If an Oracle Virtual Directory process stops on OVDHOST1 or OVDHOST2, or if either host OVDHOST1 or OVDHOST2 is down, the load balancer must continue to route the LDAP traffic to the surviving computer.

oif.mycompany.com

This virtual server acts as the access point for all HTTP traffic to the Oracle Identity Federation servers in the application tier. Traffic to both the SSL and non-SSL port is configured. The clients access this service using the address oif.mycompany.com:4444 for SSL and ovd.mycompany.com:7499 for non-SSL.

Monitor the heartbeat of the Oracle Internet Federation Server processes on OIFHOST1 and OIFHOST2. If an Oracle Internet Federation Server process stops on OIFHOST1 or OIFHOST2, or if either host OIFHOST1 or OIFHOST2 is down, the load balancer must continue to route traffic to the surviving computer.

admin.mycompany.com

This virtual server acts as the access point for all internal HTTP traffic that gets directed to the administration services. The incoming traffic from clients is non-SSL enabled. Thus, the clients access this service using the address admin.mycompany.com:80 and in turn forward these to ports 7777 on WEBHOST1 and WEBHOST2. The services accessed on this virtual host include the WebLogic Administration Server Console, Oracle Enterprise Manager and Oracle Directory Services Manager.

In addition, ensure that the virtual server names are associated with IP addresses and are part of your Domain Name System (DNS). The computers on which Oracle Fusion Middleware is running must be able to resolve these virtual server names.

7.3 Oracle Internet Directory High Availability

This section provides an introduction to Oracle Internet Directory and describes how to design and deploy a high availability environment for Oracle Internet Directory.

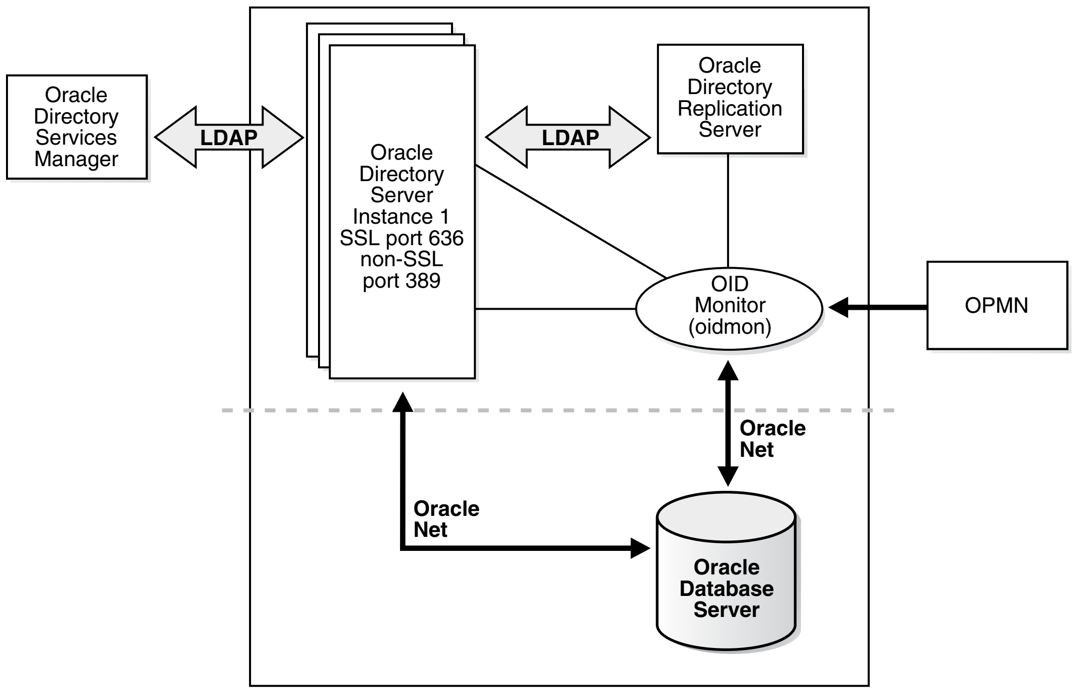
7.3.1 Oracle Internet Directory Component Architecture

Oracle Internet Directory is an LDAP store that can be used by Oracle components such as Directory Integration Platform, Oracle Directory Services Manager, JPS, and also by non-Oracle components. These components connect to Oracle Internet Directory using the LDAP or LDAPS protocols.

The Oracle directory replication server uses LDAP to communicate with an Oracle directory (LDAP) server instance. To communicate with the database, all components use OCI/Oracle Net Services. Oracle Directory Services Manager and the command-line tools communicate with the Oracle directory servers over LDAP.

Figure 7–2 shows Oracle Internet Directory in a non-high availability architecture.

Figure 7–2 Oracle Internet Directory in a Non-High Availability Architecture


Oracle Directory Services Manager
LDAP
Oracle Directory Server Instance 1 SSL port 636 non-SSL port 389
LDAP
Oracle Directory Replication Server
OID Monitor (oidmon)
OPMN
Oracle Net
Oracle Net
Oracle Database Server

An Oracle Internet Directory node consists of one or more directory server instances connected to the same directory store. The directory store—that is, the repository of the directory data—is an Oracle database.

Figure 7–2 shows the various directory server elements and their relationships running on a single node.

Oracle Net Services is used for all connections between the Oracle database server and:

- The Oracle directory server non-SSL port (389 for this topology)
- The Oracle directory server SSL-enabled port (636 for this topology)
- The OID Monitor

LDAP is used for connections between the directory server and:

- Oracle Directory Services Manager
- Oracle directory replication server

The Oracle directory server instance and the Oracle directory replication server connect to OID Monitor by way of the operating system.

As shown in Figure 7–2, an Oracle Internet Directory node includes the following major elements:

Table 7–4 An Oracle internet Directory Node

Element	Description
Oracle directory server instance	Also called either an LDAP server instance or a directory server instance, it services directory requests through a single Oracle Internet Directory dispatcher process listening at specific TCP/IP ports. There can be more than one directory server instance on a node, listening on different ports.
Oracle directory replication server	Also called a replication server, it tracks and sends changes to replication servers in another Oracle Internet Directory system. There can be only one replication server on a node. You can choose whether to configure the replication server. If there are multiple instances of Oracle Internet Directory that use the same database, only one of them can be running replication. This is true even if the Oracle Internet Directory instances are on different nodes. The replication sever process is a process within Oracle Internet Directory. It only runs when replication is configured. For more information about Oracle Internet Directory replication, refer to Chapter 8, "Configuring Identity Management for Maximum High Availability.".
Oracle Database Server	Stores the directory data. Oracle strongly recommends that you dedicate a database for use by the directory. The database can reside on the same node as the directory server instances.
Oracle Process Manager and Notification Server (OPMN)	Manages Oracle Internet Directory as an Oracle Fusion Middleware component. OPMN uses the directives in the OID component snippet in `ORACLE_INSTANCE/opmn.xml` and invokes OIDMON and OIDCTL as required. The command-line utility is `opmnctl`.
OID Monitor (OIDMON)	Initiates, monitors, and terminates the LDAP server and replication server processes. When you invoke process management commands, such as oidctl or opmnctl, or when you use Fusion Middleware Control to start or stop server instances, your commands are interpreted by this process. OIDMON also monitors servers and restarts them if they have stopped running for abnormal reasons. OIDMON starts a default instance of OIDLDAPD. If the default instance of OIDLDAPD is stopped using the OIDCTL command, then OIDMON stops the instance. When OIDMON is restarted by OPMN, OIDMON restarts the default instance. All OID Monitor activity is logged in the file `ORACLE_INSTANCE/diagnostics/log/OID/component_id/oidmon-xxxx.log`. This file is on the Oracle Internet Directory server file system. OID Monitor checks the state of the servers through mechanisms provided by the operating system.
OID Control Utility (OIDCTL)	Communicates with OID Monitor by placing message data in Oracle Internet Directory server tables. This message data includes configuration parameters required to run each Oracle directory server instance. Normally used from the command line only to stop and start the replication server.

7.3.1.1 Oracle Internet Directory Component Characteristics

Oracle Internet Directory, which is Oracle's LDAP store, is a C-based component that uses a database as its persistence store. It is a stateless process and stores all of the data and the majority of its configuration information in the back-end database. It uses Oracle Net Services to connect to the database.

7.3.1.1.1 Runtime Processes Oracle Internet Directory has the following runtime processes:

- OIDLDAPD: This is the main process for Oracle Internet Directory. OIDLDAPD consists of a dispatcher process and a server process. The dispatcher process spawns the OIDLDAPD server processes during startup. Each OIDLDAPD dispatcher process has its own SSL and non-SSL ports for receiving requests. Every OID instance has one dispatcher and one server process by default. The number of server processes spawned for an instance is controlled by the `orclserverprocs` attribute.
- OIDMON: OIDMON is responsible for the process control of an Oracle Internet Directory instance. This process starts, stops, and monitors Oracle Internet Directory. During startup OIDMON spawns the OIDLDAPD dispatcher process and the replication server process, if replication is configured for the instance.
- Replication server process: This is a process within Oracle Internet Directory that runs only when replication is configured. The replication server process is spawned by OIDMON during startup.
- OPMN: The Oracle Process Manager and Notification Server (OPMN) is a daemon process that monitors Oracle Fusion Middleware components, including Oracle Internet Directory. Oracle Enterprise Manager Fusion Middleware Control uses OPMN to stop or start instances of Oracle Internet Directory. If you stop or start Oracle Internet Directory components from the command line, you use opmnctl, the command-line interface to OPMN.

  OPMN is responsible for the direct start, stop, restart and monitoring of OIDMON. It does not start or stop the server process directly.

7.3.1.1.2 Process Lifecycle OPMN is responsible for the direct start, stop, restart and monitoring of the daemon process, OIDMON (`*ORACLE_HOME*/bin/oidmon`). OIDMON is responsible for the process control of an Oracle Internet Directory instance. In 11g Release 1 (11.1.1), you can have multiple instances of Oracle Internet Directory on the same Oracle instance on the same node. For details, refer to *Oracle Fusion Middleware Administrator's Guide for Oracle Internet Directory*.

Process Status Table

Oracle Internet Directory process information is maintained in the ODS_PROCESS_STATUS table in the ODS database user schema. OIDMON reads the contents of the table at a specified interval and acts upon the intent conveyed by the contents of that table. The interval is controlled by the value of the sleep command line argument used at OIDMON startup, and the default value is 10 seconds.

Starting and Stopping Oracle Internet Directory

An Oracle Internet Directory instance can be started and stopped using the Oracle Enterprise Manager Fusion Middleware Control or the command opmnctl.

Start Process

The start process for Oracle Internet Directory is:

1. Upon receiving the start command, OPMN issues an oidmon start command with appropriate arguments, as specified in the opmn.xml file.
2. OIDMON then starts all Oracle Internet Directory Server instances whose information in the ODS_PROCESS_STATUS table has state value 1 or 4 and ORACLE_INSTANCE, COMPONENT_NAME, INSTANCE_NAME values matching the environment parameters set by OPMN.

Stop Process

The stop process for Oracle Internet Directory is:

1. Upon receiving the stop command, OPMN issues an oidmon stop command.
2. For each row in the ODS_PROCESS_STATUS table that matches the environment parameters ORACLE_INSTANCE, COMPONENT_NAME, and INSTANCE_NAME, the oidmon stop command kills OIDMON, OIDLDAPD, and OIDREPLD processes and updates the state to 4.

Monitoring

OPMN does not monitor server processes directly. OPMN monitors OIDMON and OIDMON monitors the server processes. The events are:

- When you start OIDMON through OPMN, OPMN starts OIDMON and ensures that OIDMON is up and running.
- If OIDMON goes down for some reason, OPMN brings it back up.
- OIDMON monitors the status of the Oracle Internet Directory dispatcher process, LDAP server processes, and replication server process and makes this status available to OPMN and Oracle Enterprise Manager Fusion Middleware Control.

7.3.1.1.3 Request Flow

Once the Oracle Internet Directory process starts up, clients access Oracle Internet Directory using the LDAP or LDAPS protocol. There is no impact on other running instances when an Oracle Internet Directory instance starts up.

Oracle Internet Directory listener/dispatcher starts a configured number of server processes at startup time. The number of server processes is controlled by the `orclserverprocs` attribute in the instance-specific configuration entry. The default value for `orclserverprocs` is 1. Multiple server processes enable Oracle Internet Directory to take advantage of multiple processor systems.

The Oracle Internet Directory dispatcher process sends the LDAP connections to the Oracle Internet Directory server process in a round robin fashion. The maximum number of LDAP connections accepted by each server is 1024 by default. This number can be increased by changing the attribute `orclmaxldapconns` in the instance-specific configuration entry, which has a DN of the form:

```
cn=componentname,cn=osdldapd,cn=subconfigsubentry
```

Database connections from each server process are spawned at server startup time, depending on the value set for the instance configuration parameters ORCLMAXCC and ORCLPLUGINWORKERS. The number of database connections spawned by each server equals ORCLMAXCC + ORCLPLUGINWORKERS + 2. The Oracle Internet Directory server processes communicate with the Oracle database server through Oracle Net Services. An Oracle Net Services listener/dispatcher relays the request to the Oracle database. For more information, refer to *Oracle Fusion Middleware Administrator's Guide for Oracle Internet Directory*.

7.3.1.1.4 Configuration Artifacts

The storage location requires a DB connect string. TNSNAMES.ORA is stored in *ORACLE_INSTANCE*/`config`. The wallet is stored in *ORACLE_INSTANCE*/`OID/admin` (The DB ODS user password is stored in the wallet).

7.3.1.1.5 External Dependencies

Oracle Internet Directory uses an Oracle database to store configuration information as well as data. It uses the ODS schema to store this information.

The Oracle directory replication server uses LDAP to communicate with an Oracle directory (LDAP) server instance. To communicate with the database, all components use OCI/Oracle Net Services. Oracle Directory Services Manager and the command-line tools communicate with the Oracle directory servers over LDAP.

7.3.1.1.6 Oracle Internet Directory Log File Log files for Oracle Internet Directory are under the following directory:

```
ORACLE_INSTANCE/diagnostics/log/OID
```

Table 7–5 shows Oracle Internet Directory processes and the log file name and location for the process.

Table 7–5 Locations of Oracle Internet Directory Process Log Files

Process	Log File Location
Directory server (oidldapd)	`ORACLE_INSTANCE/diagnostics/logs/OID/componentName/oidldapd00sPID-XXXX.log` where: 00 is the instance number (00 by default) s stands for server PID is the server process identifier XXXX is a number from 0000 to orclmaxlogfilesconfigured. Once the orclmaxlogfilesconfigured value is reached, it starts over again from 0000. When it starts over, it truncates the file to 0 bytes. `ORACLE_INSTANCE/diagnostics/logs/OID/componentName/oidstackInstNumberPID.log`
LDAP dispatcher (oidldapd)	`ORACLE_INSTANCE/diagnostics/logs/OID/componentName/oidldapd00-XXXX.log` where: 00 is the instance number (00 by default) XXXX is a number from 0000 to orclmaxlogfilesconfigured
OID Monitor (OIDMON)	`ORACLE_INSTANCE/diagnostics/logs/OID/componentName/oidmon-XXXX.log` where: XXXX is a number from 0000 to orclmaxlogfilesconfigured
Directory replication server (oidrepld)	`ORACLE_INSTANCE/diagnostics/logs/OID/componentName/oidrepld-XXXX.log` where: XXXX is a number from 0000 to orclmaxlogfilesconfigured

For more information on using the log files to troubleshoot Oracle Internet Directory issues, see Section 7.3.6, "Troubleshooting Oracle Internet Directory High Availability".

7.3.2 Oracle Internet Directory High Availability Concepts

This section provides conceptual information about using Oracle Internet Directory in a high availability two-node Cluster Configuration. See Section 7.3.2.3, "Oracle Internet Directory Prerequisites" for prerequisites and Section 7.3.3, "Oracle Internet Directory High Availability Configuration Steps" for specific steps for setting up the two-node Cluster Configuration.

7.3.2.1 Oracle Internet Directory High Availability Architecture

Figure 7–3 shows the Oracle Internet Directory Cluster Configuration high availability architecture in an active-active configuration.

Figure 7–3 Oracle Internet Directory Cluster Configuration High Availability Architecture

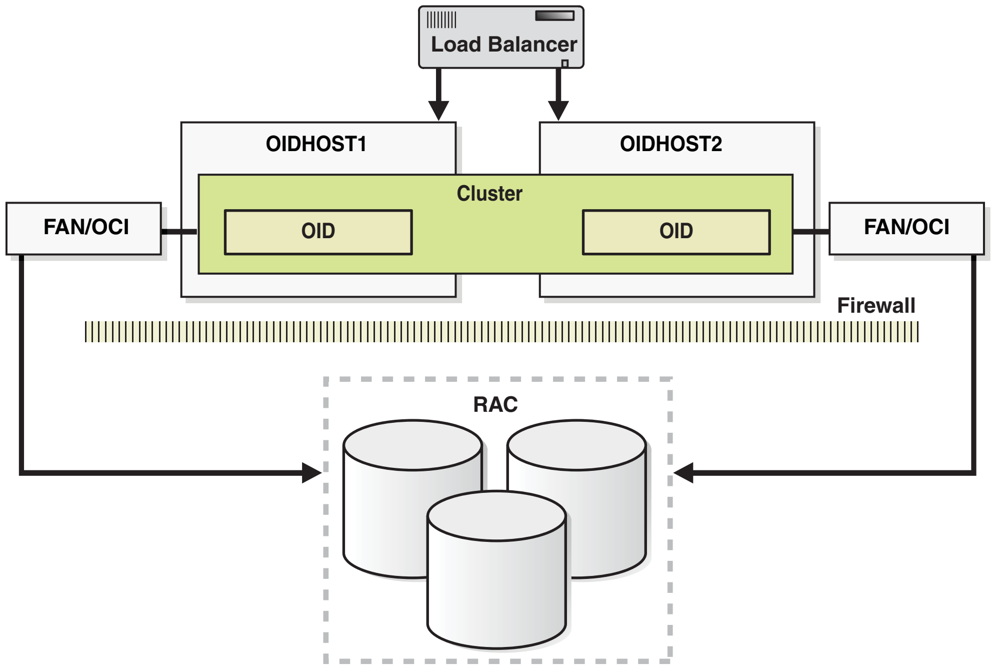


Figure 7–3 shows Oracle Internet Directory in the directory tier in a Cluster Configuration high availability architecture. Clustering is set up at installation time. The load balancing router routes LDAP client requests to the two Oracle Internet Directory instances that are clustered on OIDHOST1 and OIDHOST2.

Transparent Application Failover (TAF) is used to connect the Oracle Internet Directory instances with the RAC database that serves as the security metadata repository. The RAC database is configured in TNSNAMES.ORA. High availability event notification is used for notification when a RAC instance becomes unavailable. See Section 4.1.6.1, "Oracle Internet Directory" for more information about using Oracle Internet Directory with Oracle RAC.

7.3.2.1.1 Starting and Stopping the Cluster In the Cluster Configuration, OPMN commands are used to start each Oracle Internet Directory instance. There is no impact to Oracle Internet Directory at startup. A new database connection is spawned when Oracle Internet Directory starts.

When the cluster is stopped using OPMN, Oracle Internet Directory disconnects from the database and the Oracle Internet Directory server stops.

7.3.2.1.2 Cluster-Wide Configuration Changes Configuration changes can be done at a cluster level to any instance in the Cluster Configuration. All the nodes in the Cluster Configuration that share the same database read the same configuration information. The OIDMON process polls for configuration changes on each Oracle Internet Directory server and updates the database repository about configuration changes. OIDMON and other Oracle Internet Directory servers pull the changes from the database repository. In this way, any change made at a cluster member level is propagated to every Oracle Internet Directory server in the cluster.

The instance-specific configuration attributes for an Oracle Internet Directory LDAP server configuration are stored in this LDAP entry:

```
cn=<component-name>,cn=configsets,cn=osdldapd,cn=subconfigsubentry
```

Oracle Internet Directory server configuration aspects such as the number of servers, database connections, sizelimit, and timelimit are part of the instance-specific server configuration entry.

The configuration attributes that are common to all Oracle Internet Directory instances in a cluster are stored in the LDAP entry:

```
cn=dsaconfig,cn=configsets,cn=osdldapd,cn=oracle internet directory
```

If you want to retain instance-specific server configuration attributes for each Oracle Internet Directory instance in the cluster, then you should choose a distinct Oracle Internet Directory component name for each Oracle Internet Directory instance at install/configuration time; for example, oid1 on node1 and oid2 on node2. In this case, the configuration entries will be cn=oid1,cn=osdldapd,cn=subconfigsubentry and cn=oid2,cn=osdldapd,cn=subconfigsubentry respectively and they need to be updated separately for each Oracle Internet Directory instance.

On the other hand, if you chooses to have a common set of server configuration attributes for both Oracle Internet Directory instances in the cluster, then you should choose the same Oracle Internet Directory component name for both Oracle Internet Directory instances, for example, oid1 on both Oracle Internet Directory node1 and node2. In this case, there will be one common configuration entry cn=oid1,cn=osdldapd,cn=subconfigsubentry.

Oracle Internet Directory LDAP server instances cache certain LDAP metadata artifacts such as Schema, ACLs, and Password Policy. Multiple Oracle Internet Directory LDAP server processes on a given node keep their caches in sync via semantics built around a shared memory segment managed by Oracle Internet Directory on each node. OIDMON keeps these caches in sync across nodes by ensuring that these shared memory segments are in sync across the nodes, which is achieved using the Oracle Internet Directory database.

Oracle Internet Directory also caches metadata and metadata changes trigger notification across the nodes. The ldapmodify utility is used to change metadata. The Oracle Internet Directory server that gets the ldapmodify request for the metadata change notifies other Oracle Internet Directory servers about the change of metadata (including OIDMON). OIDMON is responsible for notifying OIDMON on other nodes about the metadata changes.

7.3.2.2 Protection from Failures and Expected Behavior

This section discusses protection from different types of failure in an Oracle Internet Directory Cluster Configuration.

7.3.2.2.1 Oracle Internet Directory Process Failure OIDMON monitors Oracle Internet Directory processes. If the Oracle Internet Directory process goes down, OIDMON tries to restart it.

OIDMON is monitored by OPMN. If OIDMON goes down, OPMN restarts OIDMON.

If an Oracle Internet Directory process cannot be restarted, then the front-ending load balancing router detects failure of Oracle Internet Directory instances in the Cluster Configuration and routes LDAP traffic to surviving instances. In case of failure, the LDAP client retries the transaction. If the instance fails in the middle of a transaction, the transaction is not committed to the database. When the failed instance comes up again, the load balancing router detects this and routes requests to all the instances.

If an Oracle Internet Directory instance in the Cluster Configuration gets hung, the load balancing router detects this and routes requests to surviving instances.

If one of the Oracle Internet Directory instances in the two-node Cluster Configuration fails (or if one of the computers hosting an instance fails), the load balancing router routes clients to the surviving Oracle Internet Directory instance.

7.3.2.2.2 Expected Client Application Behavior When Failure Occurs Oracle Internet Directory server failure is usually transparent to Oracle Internet Directory clients as they continue to get routed through the load balancer. External load balancers are typically configured to perform a health check of Oracle Internet Directory processes. If a request is received before the load balancer detects process unavailability, clients application could receive a error. If the client application performs a retry, the load balancer should route it to a healthy Oracle Internet Directory instance and the request should be successful.

In Oracle Internet Directory active-active configurations, if you are doing ldapadd operations through the LDIF file at the time of failover, your operation would fail even if you are doing this operation through a load balancer host and port. This is because Oracle Internet Directory is down for a fraction of a second. For most applications, this will not be an issue because most applications have the ability to retry the connection a fixed number of times.

7.3.2.2.3 External Dependency Failure This section describes the protection available for Oracle Internet Directory from database failure.

By default, the `tnsnames.ora` file configured in Oracle Internet Directory's ORACLE_INSTANCE ensures that Oracle Internet Directory's connections to the database are load balanced between the RAC database instances. For example, if an Oracle Internet Directory instance establishes four database connections, two connections are made to each database instance.

Oracle Internet Directory uses database high availability event notification to detect database node failure and to fail over to a surviving node.

If Transparent Application Failover (TAF) is configured, then upon a database instance failure, Oracle Internet Directory will fail over its database connections to the surviving database instance, which enables the LDAP search operations that were in progress during the failover to be continued.

If both Transparent Application Failover (TAF) and high availability event notification are configured, Transparent Application Failover (TAF) is used for failover and high availability event notifications are used only for logging the events. The high availability event notifications are logged in OIDLDAPD log file.

Oracle Internet Directory also has a mechanism to detect stale database connections, which allows Oracle Internet Directory to reconnect to the database.

If none of the database instances are available for a prolonged period, then the Oracle Internet Directory LDAP and REPL processes will automatically be shut down. However, OIDMON and OPMN will continue to ping for the database instance availability and when the database becomes available, the Oracle Internet Directory processes (LDAP and REPL) are automatically restarted by OIDMON.

While all the database instances are down, OIDMON will continue to be up and an `opmnctl status` command will show that OIDLDAPD instances are down. When a database instance becomes available, OIDMON will restart all configured Oracle Internet Directory instances.

All database failover induced activity for Oracle Internet Directory is recorded in the OIDMON log file.

7.3.2.3 Oracle Internet Directory Prerequisites

This section describes prerequisites for setting up the Oracle Internet Directory high availability architecture.

7.3.2.3.1 Synchronizing the Time on Oracle Internet Directory Nodes Before setting up Oracle Internet Directory in a high availability environment, you must ensure that the time on the individual Oracle Internet Directory nodes is synchronized.

Synchronize the time on all nodes using Greenwich Mean Time so that there is a discrepancy of no more than 250 seconds between them.

If OID Monitor detects a time discrepancy of more than 250 seconds between the two nodes, the OID Monitor on the node that is behind stops all servers on its node. To correct this problem, synchronize the time on the node that is behind in time. The OID Monitor automatically detects the change in the system time and starts the Oracle Internet Directory servers on its node.

If there are more than two nodes, the same behavior is followed. For example, assume that there are three nodes, where the first node is 150 seconds ahead of the second node, and the second node is 150 seconds ahead of the third node. In this case, the third node is 300 seconds behind the first node, so the OID Monitor will not start the servers on the third node until the time is synchronized.

7.3.2.3.2 Using RCU to Create Oracle Internet Directory Schemas in the Repository Before you can install the Oracle Internet Directory instances on OIDHOST1 and OIDHOST2, you must use the Repository Creation Utility (RCU) to create the collection of schemas used by Oracle Identity Management and Management Services.

The Repository Creation Utility (RCU) ships on its own CD as part of the Oracle Fusion Middleware 11g kit.

Follow these steps to run RCU and create the Identity Management schemas in a RAC database repository:

1. Issue this command:

   prompt> `RCU_HOME/bin/rcu &`

2. On the Welcome screen, click **Next**.
3. On the Create Repository screen, select the **Create** operation to load component schemas into an existing database.

   Click **Next**.

4. On the Database Connection Details screen, enter connection information for the existing database as follows:

   Database Type: `Oracle Database`

   Host Name: Name of the computer on which the database is running. For a RAC database, specify the VIP name or one node name. Example: `INFRADBHOST1-VIP` or `INFRADBHOST2-VIP`

   Port: The port number for the database. Example: `1521`

   Service Name: The service name of the database. Example: `idmedg.mycompany.com`

   Username: `SYS`

   Password: The SYS user password

   Role: `SYSDBA`

Click **Next**.

5. On the Select Components screen, create a new prefix and select the components to be associated with this deployment:

   Create a New Prefix: `idm` (Entering a prefix is optional if you select only **Identity Management** (Oracle Internet Directory - ODS) in the **Components** field)

   Components: Select **Identity Management** (Oracle Internet Directory - ODS). De-select all other schemas.

   Click **Next**.

6. On the Schema Passwords screen, enter the passwords for the main and additional (auxiliary) schema users.

   Click **Next**.

7. On the Map Tablespaces screen, select the tablespaces for the components.
8. On the Summary screen, click **Create**.
9. On the Completion Summary screen, click **Close**.

7.3.2.3.3 Load Balancer Virtual Server Names for Oracle Internet Directory
When Oracle Internet Directory is deployed in a high availability configuration, it is recommended to use an external load balancer to front-end the Oracle Internet Directory instances and load balance requests between the various Oracle Internet Directory instances.

Refer to Section 7.2.4.5, "Configuring Virtual Server Names and Ports for the Load Balancer" for details.

7.3.3 Oracle Internet Directory High Availability Configuration Steps

Oracle Internet Directory can be deployed in a highly availability configuration either in a standalone mode or as a part of a WebLogic Server domain.

The standalone mode should be chosen for deployments where Oracle Internet Directory will be managed entirely using command line tools, and where Oracle Enterprise Manager Fusion Middleware Control and Oracle Directory Services Manager are not deemed mandatory. Later, if desired, the standalone Oracle Internet Directory instances can be registered with a remote Oracle WebLogic Server domain using opmnctl commands. Oracle Enterprise Manager Fusion Middleware Control and Oracle Directory Services Manager can be used to manage Oracle Internet Directory instances that are configured with an Oracle WebLogic Server domain.

It is recommended that Oracle Internet Directory be set up in a clustered deployment, where the clustered Oracle Internet Directory instances access the same RAC database repository.

7.3.3.1 Configuring Oracle Internet Directory Without a WebLogic Domain

This section describes the steps to deploy Oracle Internet Directory without a WebLogic Server domain.

7.3.3.1.1 Installing Oracle Internet Directory on OIDHOST1
Make sure that the schema database is running and that RCU has been used to seed the ODS database schema, then follow these steps to install the Oracle Internet Directory instance on OIDHOST1:

1. Ensure that the system, patch, kernel and other requirements are met. These are listed in the *Oracle Fusion Middleware Installation Guide for Oracle Identity*

Management in the Oracle Fusion Middleware documentation library for the platform and version you are using.

2. Ensure that ports 389 and 636 are not in use by any service on the computer by issuing these commands for the operating system you are using. If a port is not in use, no output is returned from the command.

   On UNIX:

   ```
   netstat -an | grep "389"

   netstat -an | grep "636"
   ```

   On Windows:

   ```
   netstat -an | findstr :389

   netstat -an | findstr :636
   ```

3. If the port is in use (if the command returns output identifying the port), you must free the port.

   On UNIX:

   Remove the entries for ports 389 and 636 in the `/etc/services` file and restart the services, or restart the computer.

   On Windows:

   Stop the component that is using the port.

4. Copy the `staticports.ini` file from the `Disk1/stage/Response` directory to a temporary directory.

5. Edit the `staticports.ini` file you copied to the temporary directory to assign the following custom ports (uncomment the lines where you specify the port numbers for Oracle Internet Directory):

   ```
   # The non-SSL port for Oracle Internet Directory
   Oracle Internet Directory port = 389
   # The SSL port for Oracle Internet Directory
   Oracle Internet Directory (SSL) port = 636
   ```

6. Start the Oracle Identity Management 11g installer as follows:

   On UNIX, issue this command: **`runInstaller`**

   On Windows, double-click **`setup.exe`**

   The `runInstaller` and `setup.exe` files are in the `../install/`*`platform`* directory, where *platform* is a platform such as Linux or Win32.

   This displays the Specify Oracle Inventory screen.

7. On the Specify Inventory Directory screen, enter values for the Oracle Inventory Directory and the Operating System Group Name. For example:

   Specify the Inventory Directory: `/u01/app/oraInventory`

   Operating System Group Name: `oinstall`

   A dialog box appears with the following message:

   "Certain actions need to be performed with root privileges before the install can continue. Please execute the script /u01/app/oraInventory/createCentralInventory.sh now from another window and then press "Ok" to continue the install. If you do not have the root privileges and wish to continue the install select the "Continue installation with local inventory" option"

Log in as root and run the "/u01/app/oraInventory/createCentralInventory.sh"

This sets the required permissions for the Oracle Inventory Directory and then brings up the Welcome screen.

Note: The Oracle Inventory screen is not shown if an Oracle product was previously installed on the host. If the Oracle Inventory screen is not displayed for this installation, make sure to check and see:

1. If the `/etc/oraInst.loc` file exists
2. If the file exists, the Inventory directory listed is valid
3. The user performing the installation has write permissions for the Inventory directory

---

8. On the Welcome screen, click **Next**.
9. On the Select Installation Type screen, select **Install and Configure** and then click **Next**.
10. On the Prerequisite Checks screen, the installer completes the prerequisite check. If any fail, fix them and restart your installation.

    Click **Next**.
11. On the Select Domain screen, select **Configure without a Domain** and then click **Next**.
12. On the Specify Installation Location screen, specify the following values:
    - **Oracle Home Location:**

      `/u01/app/oracle/product/fmw/idm`
    - **Oracle Instance Location:**

      `/u01/app/oracle/admin/oid_instance1`
    - **Oracle Instance Name**:

      `oid_instance1`

    Note: Ensure that the Oracle Home Location directory path for OIDHOST1 is the same as the Oracle Home Location path for OIDHOST2. For example, if the Oracle Home Location directory path for OIDHOST1 is:

    ```
    /u01/app/oracle/product/fmw/idm
    ```

    then the Oracle Home Location directory path for OIDHOST2 must be:

    ```
    /u01/app/oracle/product/fmw/idm
    ```

    Click **Next**.

13. On the Specify Email for Security Updates screen, specify these values:

    - **Email Address**: Provide the email address for your My Oracle Support account.
    - **Oracle Support Password**: Provide the password for your My Oracle Support account.
    - Check the checkbox next to the **I wish to receive security updates via My Oracle Support** field.

    Click **Next**.

14. On the Configure Components screen, select **Oracle Internet Directory**, deselect all the other components, and click **Next**.

15. On the Configure Ports screen, select **Specify Ports Using Configuration File** and enter the filename for the `staticports.ini` file that you copied to the temporary directory.

    Click **Next**.

16. On the Specify Schema Database screen, select **Use Existing Schema** and specify the following values:

    - Connect String:

      ```
      infradbhost1-vip.mycompany.com:1521:idmdb1^infradbhost2-vip.mycompany.com:1
      521:idmdb2@idmedg.mycompany.com
      ```

      Note: The RAC database connect string information needs to be provided in the format *host1*:*port1*:*instance1*^*host2*:*port2*:*instance2*@*servicename*.

      During this installation, it is not required for all the RAC instances to be up. If one RAC instance is up, the installation can proceed.

      It is required that the information provided above is complete and accurate. Specifically, the correct host, port, and instance name must be provided for each RAC instance, and the service name provided must be configured for all the specified RAC instances.

      Any incorrect information entered in the RAC database connect string has to be corrected manually after the installation.

    - User Name: `ODS`
    - Password: `******`

    Click **Next**.

17. On the Create Oracle Internet Directory screen, specify the Realm and enter the Administrator (`cn=orcladmin`) password and click **Next**.

18. On the Installation Summary screen, review the selections to ensure that they are correct (if they are not, click **Back** to modify selections on previous screens), and click **Install**.

19. On the Installation Progress screen on UNIX systems, a dialog box appears that prompts you to run the `oracleRoot.sh` script. Open a window and run the script, following the prompts in the window.

Click **Next**.

20. On the Configuration screen, multiple configuration assistants are launched in succession; this process can be lengthy. When it completes, click **Next**.
21. On the Installation Complete screen, click **Finish** to confirm your choice to exit.

7.3.3.1.2 Oracle Internet Directory Component Names Assigned by Oracle Identity Management Installer

When you perform an Oracle Internet Directory installation using Oracle Identity Management 11*g* installer, the default component name that the installer assigns to the Oracle Internet Directory instance is oid1. You cannot change this component name.

The instance-specific configuration entry for this Oracle Internet Directory instance is `cn=oid1, cn=osdldapd, cn=subconfigsubentry`.

If you perform a second Oracle Internet Directory installation on another computer and that Oracle Internet Directory instance uses the same database as the first instance, the installer detects the previously installed Oracle Internet Directory instance on the other computer using the same Oracle database, so it gives the second Oracle Internet Directory instance a component name of oid2.

The instance-specific configuration entry for the second Oracle Internet Directory instance is `cn=oid2, cn=osdldapd, cn=subconfigsubentry`. Any change of properties in the entry `cn=oid2, cn=osdldapd, cn=subconfigsubentry` will not affect the first instance (oid1).

If a third Oracle Internet Directory installation is performed on another computer and that instance uses the same database as the first two instances, the installer gives the third Oracle Internet Directory instance a component name of oid3, and so on for additional instances on different hosts that use the same database.

Note that the shared configuration for all Oracle Internet Directory instances is `cn=dsaconfig, cn=configsets,cn=oracle internet directory`. Any change in this entry will affect all the instances of Oracle Internet Directory.

This naming scheme helps alleviate confusion when you view your domain using Oracle Enterprise Manager by giving different component names to your Oracle Internet Directory instances.

7.3.3.1.3 Installing Oracle Internet Directory on OIDHOST2

Make sure that the Oracle Internet Directory repository is running and then follow these steps to install the Oracle Internet Directory instance on OIDHOST2:

Note: The instructions in this section can also be used to scale out Oracle Internet Directory in your 11*g* Oracle Identity Management high availability configuration.

1. Ensure that the system, patch, kernel and other requirements are met. These are listed in the *Oracle Fusion Middleware Installation Guide for Oracle Identity Management* in the Oracle Fusion Middleware documentation library for the platform and version you are using.
2. On OIDHOST1, ports 389 and 636 were used for Oracle Internet Directory. The same ports should be used for the Oracle Internet Directory instance on OIDHOST2. Therefore, ensure that ports 389 and 636 are not in use by any service on OIDHOST2 by issuing these commands for the operating system you are using. If a port is not in use, no output is returned from the command.

On UNIX:

```
netstat -an | grep "389"

netstat -an | grep "636"
```

On Windows:

```
netstat -an | findstr :389

netstat -an | findstr :636
```

3. If the port is in use (if the command returns output identifying the port), you must free the port.

   On UNIX:

   Remove the entries for ports 389 and 636 in the `/etc/services` file and restart the services, or restart the computer.

   On Windows:

   Stop the component that is using the port.

4. Copy the `staticports.ini` file from the `Disk1/stage/Response` directory to a temporary directory.

5. Edit the `staticports.ini` file you copied to the temporary directory to assign the following custom ports (uncomment the lines where you specify the port numbers for Oracle Internet Directory):

   ```
   # The non-SSL port for Oracle Internet Directory
   Oracle Internet Directory port = 389
   # The SSL port for Oracle Internet Directory
   Oracle Internet Directory (SSL) port = 636
   ```

6. Start the Oracle Identity Management 11*g* Installer as follows:

   On UNIX, issue this command: **`runInstaller`**

   On Windows, double-click **`setup.exe`**

   The `runInstaller` and `setup.exe` files are in the `../install/`*`platform`* directory, where *platform* is a platform such as Linux or Win32.

   This displays the Specify Oracle Inventory screen.

7. On the Specify Inventory Directory screen, enter values for the Oracle Inventory Directory and the Operating System Group Name. For example:

   Specify the Inventory Directory: `/u01/app/oraInventory`

   Operating System Group Name: `oinstall`

   A dialog box appears with the following message:

   "Certain actions need to be performed with root privileges before the install can continue. Please execute the script /u01/app/oraInventory/createCentralInventory.sh now from another window and then press "Ok" to continue the install. If you do not have the root privileges and wish to continue the install select the "Continue installation with local inventory" option"

   Log in as root and run the "/u01/app/oraInventory/createCentralInventory.sh"

This sets the required permissions for the Oracle Inventory Directory and then brings up the Welcome screen.

Note: The Oracle Inventory screen is not shown if an Oracle product was previously installed on the host. If the Oracle Inventory screen is not displayed for this installation, make sure to check and see:

1. If the `/etc/oraInst.loc` file exists
2. If the file exists, the Inventory directory listed is valid
3. The user performing the installation has write permissions for the Inventory directory

---

8. On the Welcome screen, click **Next**.
9. On the Select Installation Type screen, select **Install and Configure** and then click **Next**.
10. On the Prerequisite Checks screen, the installer completes the prerequisite check. If any fail, fix them and restart your installation.

    Click **Next**.
11. On the Select Domain screen, select **Configure without a Domain** and then click **Next**.
12. On the Specify Installation Location screen, specify the following values:

    - **Oracle Home Location:**

      ```
      /u01/app/oracle/product/fmw/idm
      ```
    - **Oracle Instance Location:**

      ```
      /u01/app/oracle/admin/oid_instance2
      ```
    - **Oracle Instance Name**:

      ```
      oid_instance2
      ```

    Note: Ensure that the Oracle Home Location directory path for OIDHOST2 is the same as the Oracle Home Location directory path for OIDHOST1. For example, if the Oracle Home Location directory path for OIDHOST1 is:

    `/u01/app/oracle/product/fmw/idm`

    then the Oracle Home Location directory path for OIDHOST2 must be:

    `/u01/app/oracle/product/fmw/idm`

    Click **Next**.
13. On the Specify Email for Security Updates screen, specify these values:

    - **Email Address**: Provide the email address for your My Oracle Support account.

- **Oracle Support Password**: Provide the password for your My Oracle Support account.
- Check the checkbox next to the **I wish to receive security updates via My Oracle Support** field.

Click **Next**.

14. On the Configure Components screen, select **Oracle Internet Directory**, deselect all the other components, and click **Next**.
15. On the Configure Ports screen, select **Specify Ports Using Configuration File** and enter the filename for the `staticports.ini` file that you copied to the temporary directory.

    Click **Next**.
16. On the Specify Schema Database screen, select **Use Existing Schema** and specify the following values.
    - Connect String:

    ```
    infradbhost1-vip.mycompany.com:1521:idmdb1^infradbhost2-vip.mycompany.com:1
    521:idmdb2@idmedg.mycompany.com
    ```

    Note: The RAC database connect string information needs to be provided in the format *host1*:*port1*:*instance1*^*host2*:*port2*:*instance2*@*servicename*.

    During this installation, it is not required for all the RAC instances to be up. If one RAC instance is up, the installation can proceed.

    It is required that the information provided above is complete and accurate. Specifically, the correct host, port, and instance name must be provided for each RAC instance, and the service name provided must be configured for all the specified RAC instances.

    Any incorrect information entered in the RAC database connect string has to be corrected manually after the installation.

    - User Name: `ODS`
    - Password: `******`

    Click **Next**.
17. The ODS Schema in use message appears. The ODS schema chosen is already being used by the existing Oracle Internet Directory instance. Therefore, the new Oracle Internet Directory instance being configured would reuse the same schema.

    Choose **Yes** to continue.
18. On the Specify OID Administrator Password screen, specify the OID Administrator password and click **Next**.
19. On the Installation Summary screen, review the selections to ensure that they are correct (if they are not, click **Back** to modify selections on previous screens), and click **Install**.
20. On the Installation Progress screen on UNIX systems, a dialog box appears that prompts you to run the `oracleRoot.sh` script. Open a window and run the script, following the prompts in the window.

Click **Next**.

21. On the Configuration screen, multiple configuration assistants are launched in succession; this process can be lengthy. When it completes, click **Next**.

22. On the Installation Complete screen, click **Finish** to confirm your choice to exit.

7.3.3.1.4 Registering Oracle Internet Directory with a WebLogic Domain If you want to manage an Oracle Internet Directory component with Oracle Enterprise Manager Fusion Middleware Control, you must register the component and the Oracle Fusion Middleware instance that contains it with an Oracle WebLogic Server domain. You can register an Oracle Fusion Middleware instance with a WebLogic domain during installation or Oracle instance creation, but you are not required to do so. If an Oracle Fusion Middleware instance was not previously registered with a WebLogic domain, you can register it by using `opmnctl registerinstance`.

Before using the `opmnctl registerinstance` command to register an Oracle Internet Directory instance with an Oracle WebLogic Server domain, make sure that the WebLogic Server is already installed.

Then execute the `opmnctl registerinstance` command in this format (note that the ORACLE_HOME and ORACLE_INSTANCE variables have to be set for each installation before executing this command in the home directory for the Oracle Internet Directory instance):

```
opmnctl registerinstance -adminHost WLSHostName -adminPort WLSPort
-adminUsername adminUserName
```

For example:

```
opmnctl registerinstance -adminHost idmhost1.mycompany.com -adminPort 7001
-adminUsername weblogic

Command requires login to weblogic admin server (idmhost1.mycompany.com)
 Username: weblogic
 Password: *******
```

For additional details on registering Oracle Internet Directory components with a WebLogic domain, see the "Registering an Oracle Fusion Middleware Instance or Component with the WebLogic Server" section in *Oracle Fusion Middleware Administrator's Guide for Oracle Internet Directory*.

7.3.3.2 Configuring Oracle Internet Directory With a WebLogic Domain

This section describes the steps to deploy Oracle Internet Directory in a high availability configuration as part of a WebLogic Server domain.

In this configuration, Oracle Internet Directory and a WebLogic Server domain is configured on the first host, and only Oracle Internet Directory is configured on the second host. The Oracle Internet Directory instance on the second host joins the domain created on the first host.

7.3.3.2.1 Installing Oracle WebLogic Server On OIDHOST1, start the Oracle WebLogic Server installation by running the installer executable file.

Start the Oracle WebLogic Server installer as follows:

- On UNIX (Linux in the following example):

  ```
  ./server103_linux32.bin
  ```

  On Windows:

```
server103_win32.exe
```

Then follow these steps in the installer to install Oracle WebLogic Server on the computer:

1. On the Welcome screen, click **Next**.
2. On the Choose Middleware Home Directory screen, choose a directory on your computer into which the Oracle WebLogic software is to be installed.

   For the **Middleware Home Directory**, specify this value:

   ```
   /u01/app/oracle/product/fmw
   ```

   Click **Next**.
3. On the Register for Security Updates screen, enter your "My Oracle Support" User Name and Password.
4. On the Choose Install Type screen, the installation program displays a window in which you are prompted to indicate whether you wish to perform a complete or a custom installation.

   Choose **Typical** or **Custom**.

   Click **Next**.
5. On the Choose Product Installation Directories screen, specify the following value:

   WebLogic Server:

   ```
   /u01/app/oracle/product/fmw/wlserver_10.3
   ```

   Click **Next**.
6. On the Installation Summary screen, the window contains a list of the components you selected for installation, along with the approximate amount of disk space to be used by the selected components once installation is complete.

   Click **Next**.
7. On the Installation Complete screen, deselect the **Run Quickstart** checkbox.

   Click **Done**.

7.3.3.2.2 Installing Oracle Internet Directory on OIDHOST1 The schema database must be running before you perform this task. Also make sure that RCU has been used to seed the ODS database schema. Then follow these steps to install the 11*g* Oracle Internet Directory on OIDHOST1:

1. Ensure that the system, patch, kernel and other requirements are met. These are listed in the *Oracle Fusion Middleware Installation Guide for Oracle Identity Management* in the Oracle Fusion Middleware documentation library for the platform and version you are using.
2. Ensure that ports 389 and 636 are not in use by any service on the computer by issuing these commands for the operating system you are using. If a port is not in use, no output is returned from the command.

   On UNIX:

   ```
   netstat -an | grep "389"

   netstat -an | grep "636"
   ```

   On Windows:

```
netstat -an | findstr :389

netstat -an | findstr :636
```

3. If the port is in use (if the command returns output identifying the port), you must free the port.

   On UNIX:

   Remove the entries for ports 389 and 636 in the `/etc/services` file and restart the services, or restart the computer.

   On Windows:

   Stop the component that is using the port.

4. Copy the `staticports.ini` file from the `Disk1/stage/Response` directory to a temporary directory.

5. Edit the `staticports.ini` file that you copied to the temporary directory to assign the following custom ports (uncomment the lines where you specify the port numbers for Oracle Internet Directory):

   ```
   # The non-SSL port for Oracle Internet Directory
   Oracle Internet Directory port = 389
   # The SSL port for Oracle Internet Directory
   Oracle Internet Directory (SSL) port = 636
   ```

6. Start the Oracle Identity Management 11*g* Installer as follows:

   On UNIX, issue this command: **`runInstaller`**

   On Windows, double-click **`setup.exe`**

   The `runInstaller` and `setup.exe` files are in the `../install/`*`platform`* directory, where *platform* is a platform such as Linux or Win32.

   This displays the Specify Oracle Inventory screen.

7. On the Specify Inventory Directory screen, enter values for the Oracle Inventory Directory and the Operating System Group Name. For example:

   Specify the Inventory Directory: `/u01/app/oraInventory`

   Operating System Group Name: `oinstall`

   A dialog box appears with the following message:

   "Certain actions need to be performed with root privileges before the install can continue. Please execute the script /u01/app/oraInventory/createCentralInventory.sh now from another window and then press "Ok" to continue the install. If you do not have the root privileges and wish to continue the install select the "Continue installation with local inventory" option"

   Log in as root and run the "/u01/app/oraInventory/createCentralInventory.sh"

   This sets the required permissions for the Oracle Inventory Directory and then brings up the Welcome screen.

Note: The Oracle Inventory screen is not shown if an Oracle product was previously installed on the host. If the Oracle Inventory screen is not displayed for this installation, make sure to check and see:

1. If the `/etc/oraInst.loc` file exists
2. If the file exists, the Inventory directory listed is valid
3. The user performing the installation has write permissions for the Inventory directory

---

8. On the Welcome screen, click **Next**.
9. On the Select Installation Type screen, select **Install and Configure** and then click **Next**.
10. On the Prerequisite Checks screen, the installer completes the prerequisite check. If any fail, fix them and restart the installation.

    Click **Next**.
11. On the Select Domain screen, select **Create New Domain**. Specify the values for.
    - **User Name**: weblogic
    - **Password**: <password for the weblogic user>
    - **Confirm Password**: <confirm the password for the weblogic user>
    - **Domain Name**: IDMDomain
12. On the Specify Installation Location screen, specify the following values:
    - **Oracle Middleware Home Location**:

      ```
      /u01/app/oracle/product/fmw
      ```
    - **Oracle Home Directory:** idm
    - **Oracle Instance Location:**

      ```
      /u01/app/oracle/admin/oid_inst1
      ```
    - **Oracle Instance Name**: oid_inst1

    Note: Ensure that the Oracle Home directory path for OIDHOST1 is the same as the Oracle Home directory path for OIDHOST2. For example, if the Oracle Home directory path for OIDHOST1 is:

    `/u01/app/oracle/product/fmw/idm`

    then the Oracle Home directory path for OIDHOST2 must be:

    `/u01/app/oracle/product/fmw/idm`

    Click **Next**.
13. On the Specify Email for Security Updates screen, specify these values:
    - **Email Address**: Provide the email address for your My Oracle Support account.
    - **Oracle Support Password**: Provide the password for your My Oracle Support account.

- Check the checkbox next to the **I wish to receive security updates via My Oracle Support** field.

  Click **Next**.

14. On the Configure Components screen, select **Oracle Internet Directory & Management Components**.

    Click **Next**.

15. On the Configure Ports screen, select **Specify Ports Using Configuration File** and enter the filename for the `staticports.ini` file that you copied to the temporary directory.

    Click **Next**.

    Note: The default Oracle WebLogic Server clustering mode set by the installer is unicast (not multicast).

    ---

16. On the Specify Schema Database screen, select **Use Existing Schema** and specify the following values:

    - Connect String:

      ```
      infradbhost1-vip.mycompany.com:1521:idmdb1^infradbhost2-vip.mycompany.com:1
      521:idmdb2@idmedg.mycompany.com
      ```

    Note: The RAC database connect string information needs to be provided in the format *host1*:*port1*:*instance1*^*host2*:*port2*:*instance2*@*servicename*.

    During this installation, it is not required for all the RAC instances to be up. If one RAC instance is up, the installation can proceed.

    It is required that the information provided above is complete and accurate. Specifically, the correct host, port, and instance name must be provided for each RAC instance, and the service name provided must be configured for all the specified RAC instances.

    Any incorrect information entered in the RAC database connect string has to be corrected manually after the installation.

    - User Name: `ODS`
    - Password: `******`

    Click **Next**.

17. On the Configure OID screen, specify the Realm and enter the Administrator (`cn=orcladmin`) password and click **Next**.

18. On the Installation Summary screen, review the selections to ensure that they are correct (if they are not, click **Back** to modify selections on previous screens), and click **Install**.

19. On the Installation Progress screen on UNIX systems, a dialog box appears that prompts you to run the `oracleRoot.sh` script. Open a window and run the script, following the prompts in the window.

Click **Next**.

20. On the Configuration screen, multiple configuration assistants are launched in succession; this process can be lengthy. When it completes, click **Next**.
21. On the Installation Complete screen, click **Finish** to confirm your choice to exit.

> **Note:** For information on the Oracle Internet Directory component names assigned by Oracle Identity Management 11*g* Installer, refer to Section 7.3.3.1.2, "Oracle Internet Directory Component Names Assigned by Oracle Identity Management Installer."

7.3.3.2.3 Creating boot.properties for the Administration Server on OIDHOST1 This section describes how to create a `boot.properties` file for the Administration Server on OIDHOST1. The `boot.properties` file enables the Administration Server to start without prompting for the `administrator` username and password.

Follow these steps to create the `boot.properties` file:

1. On OIDHOST1, go the following directory:

```
MW_HOME/user_projects/domains/domainName/servers/AdminServer/security
```

   For example:

```
cd /u01/app/oracle/product/fmw/user_
projects/domains/IDMDomain/servers/AdminServer/security
```

2. Use a text editor to create a file called `boot.properties` under the `security` directory. Enter the following lines in the file:

```
username=adminUser
password=adminUserPassword
```

> **Note:** When you start the Administration Server, the username and password entries in the file get encrypted.
>
> For security reasons, minimize the time the entries in the file are left unencrypted. After you edit the file, you should start the server as soon as possible so that the entries get encrypted.

3. Stop the Administration Server if it is running.

   See the "Starting and Stopping Oracle Fusion Middleware" chapter of the *Oracle Fusion Middleware Administrator's Guide* for information on starting and stopping WebLogic Servers.

4. Start the Administration Server on OIDHOST1 using the startWebLogic.sh script located under the `MW_HOME/user_projects/domains/domainName/bin` directory.
5. Validate that the changes were successful by opening a web browser and accessing the following pages:
   - WebLogic Server Administration Console at:

```
http://oidhost1.mycompany.com:7001/console
```

   - Oracle Enterprise Manager Fusion Middleware Control at:

```
http://oidhost1.mycompany.com:7001/em
```

Log into these consoles using the `weblogic` user credentials.

7.3.3.2.4 Installing Oracle Internet Directory on OIDHOST2 Make sure that the Oracle Internet Directory repository is running and then follow these steps to install the Oracle Internet Directory instance on OIDHOST2:

1. Ensure that the system, patch, kernel and other requirements are met. These are listed in the *Oracle Fusion Middleware Installation Guide for Oracle Identity Management* in the Oracle Fusion Middleware documentation library for the platform and version you are using.
2. On OIDHOST1, ports 389 and 636 were used for Oracle Internet Directory. The same ports should be used for the Oracle Internet Directory instance on OIDHOST2. Therefore, ensure that ports 389 and 636 are not in use by any service on OIDHOST2 by issuing these commands for the operating system you are using. If a port is not in use, no output is returned from the command.

   On UNIX:

   ```
   netstat -an | grep "389"

   netstat -an | grep "636"
   ```

   On Windows:

   ```
   netstat -an | findstr :389

   netstat -an | findstr :636
   ```

3. If the port is in use (if the command returns output identifying the port), you must free the port.

   On UNIX:

   Remove the entries for ports 389 and 636 in the `/etc/services` file and restart the services, or restart the computer.

   On Windows:

   Stop the component that is using the port.
4. Copy the `staticports.ini` file from the `Disk1/stage/Response` directory to a temporary directory.
5. Edit the `staticports.ini` file that you copied to the temporary directory to assign the following custom ports (uncomment the lines where you specify the port numbers for Oracle Internet Directory):

   ```
   # The non-SSL port for Oracle Internet Directory
   Oracle Internet Directory port = 389
   # The SSL port for Oracle Internet Directory
   Oracle Internet Directory (SSL) port = 636
   ```

6. Start the Oracle Identity Management 11*g* Installer as follows:

   On UNIX, issue this command: **`runInstaller`**

   On Windows, double-click **`setup.exe`**

   The `runInstaller` and `setup.exe` files are in the `../install/`*`platform`* directory, where *platform* is a platform such as Linux or Win32.

This displays the Specify Oracle Inventory screen.

7. On the Specify Inventory Directory screen, enter values for the Oracle Inventory Directory and the Operating System Group Name. For example:

   Specify the Inventory Directory: `/u01/app/oraInventory`

   Operating System Group Name: `oinstall`

   A dialog box appears with the following message:

   "Certain actions need to be performed with root privileges before the install can continue. Please execute the script /u01/app/oraInventory/createCentralInventory.sh now from another window and then press "Ok" to continue the install. If you do not have the root privileges and wish to continue the install select the "Continue installation with local inventory" option"

   Log in as root and run the "/u01/app/oraInventory/createCentralInventory.sh"

   This sets the required permissions for the Oracle Inventory Directory and then brings up the Welcome screen.

   Note: The Oracle Inventory screen is not shown if an Oracle product was previously installed on the host. If the Oracle Inventory screen is not displayed for this installation, make sure to check and see:

   1. If the `/etc/oraInst.loc` file exists
   2. If the file exists, the Inventory directory listed is valid
   3. The user performing the installation has write permissions for the Inventory directory

8. On the Welcome screen, click **Next**.

9. On the Select Installation Type screen, select **Install and Configure** and then click **Next**.

10. On the Prerequisite Checks screen, the installer completes the prerequisite check. If any fail, fix them and restart the installation.

    Click **Next**.

11. On the Select Domain screen, select **Extend Existing Domain** and specify the following values.

    - **HostName**: `idmhost1.mycompany.com` (This is the host where the WebLogic Administration Server is running.)
    - **Port**: `7001` (This is the WebLogic Administration Server port)
    - **User Name**: `weblogic`
    - **Password**: `<password for the weblogic user>`

12. On the Specify Installation Location screen, specify the following values:

    - **Oracle Middleware Home Location**:

      ```
      /u01/app/oracle/product/fmw
      ```

    - **Oracle Home Directory:** idm
    - **WebLogic Server Directory**:

```
/u01/app/oracle/product/fmw/wlserver_10.3
```

- **Oracle Instance Location:**

  ```
  /u01/app/oracle/admin/oid_inst2
  ```

- **Oracle Instance Name**: oid_inst2

Note: Ensure that the Oracle Home Location directory path for OIDHOST1 is the same as the Oracle Home Location path for OIDHOST2. For example, if the Oracle Home Location directory path for OIDHOST1 is:

```
/u01/app/oracle/product/fmw/idm
```

then the Oracle Home Location directory path for OIDHOST2 must be:

```
/u01/app/oracle/product/fmw/idm
```

Click **Next**.

13. On the Specify Email for Security Updates screen, specify these values:

    - **Email Address**: Provide the email address for your My Oracle Support account.
    - **Oracle Support Password**: Provide the password for your My Oracle Support account.
    - Check the checkbox next to the **I wish to receive security updates via My Oracle Support** field.

    Click **Next**.

14. On the Configure Components screen, select **Oracle Internet Directory** and click **Next**.

15. On the Configure Ports screen, select **Specify Ports Using Configuration File** and enter the filename for the `staticports.ini` file that you copied to the temporary directory.

    Click **Next**.

16. On the Specify Schema Database screen, select **Use Existing Schema** and specify the following values:

    - Connect String:

      ```
      infradbhost1-vip.mycompany.com:1521:idmdb1^infradbhost2-vip.mycompany.com:1
      521:idmdb2@idmedg.mycompany.com
      ```

Note: The RAC database connect string information needs to be provided in the format *host1*:*port1*:*instance1*^*host2*:*port2*:*instance2*@*servicename*.

During this installation, it is not required for all the RAC instances to be up. If one RAC instance is up, the installation can proceed.

It is required that the information provided above is complete and accurate. Specifically, the correct host, port, and instance name must be provided for each RAC instance, and the service name provided must be configured for all the specified RAC instances.

Any incorrect information entered in the RAC database connect string has to be corrected manually after the installation.

- User Name: `ODS`
- Password: `******`

Click **Next**.

17. The ODS Schema in use message appears. The ODS schema chosen is already being used by the existing Oracle Internet Directory instance. Therefore, the new Oracle Internet Directory instance being configured would reuse the same schema.

    Choose **Yes** to continue.

18. On the Specify OID Admin Password screen, specify the Oracle Internet Directory Administrator password and click **Next**.
19. On the Installation Summary screen, review the selections to ensure that they are correct (if they are not, click **Back** to modify selections on previous screens), and click **Install**.
20. On the Installation Progress screen on UNIX systems, a dialog box appears that prompts you to run the `oracleRoot.sh` script. Open a window and run the script, following the prompts in the window.

    Click **Next**.

21. On the Configuration screen, multiple configuration assistants are launched in succession; this process can be lengthy. When it completes, click **Next**.
22. On the Installation Complete screen, click **Finish** to confirm your choice to exit.

7.3.4 Validating Oracle Internet Directory High Availability

Use the `ldapbind` command-line tool to ensure that you can connect to each Oracle Internet Directory instance and the LDAP Virtual Server. The `ldapbind` tool enables you to determine whether you can authenticate a client to a server.

Note: See the "Configuring Your Environment" section of *Oracle Fusion Middleware User Reference for Oracle Identity Management* for a list of the environment variables you must set before using the `ldapbind` command.

For non-SSL:

```
ldapbind -h oidhost1.mycompany.com -p 389 -D "cn=orcladmin" -q
ldapbind -h oidhost2.mycompany.com -p 389 -D "cn=orcladmin" -q
ldapbind -h oid.mycompany.com -p 389 -D "cn=orcladmin" -q
```

Note: The -q option above prompts the user for a password. LDAP tools have been modified to disable the options -w *password* and -P *password* when the environment variable LDAP_PASSWORD_PROMPTONLY is set to TRUE or 1. Use this feature whenever possible.

For SSL:

```
ldapbind -h oidhost1.mycompany.com -p 636 -D "cn=orcladmin" -q -U 1
ldapbind -h oidhost2.mycompany.com -p 636 -D "cn=orcladmin" -q -U 1
ldapbind -h oid.mycompany.com -p 636 -D "cn=orcladmin" -q -U 1
```

where -U is an optional argument used to specify the SSL authentication mode. These are the valid values for the SSL authentication mode:

- 1 = No authentication required
- 2 = One way authentication required. With this option, you must also supply a wallet location (-W "*file:/home/my_dir/my_wallet*") and wallet password (-P *wallet_password*).
- 3 = Two way authentication required. With this option, you must also supply a wallet location (-W "*file:/home/my_dir/my_wallet*") and wallet password (-P *wallet_password*).

For more information about the `ldapbind` command, see the ldapbind section in *Oracle Fusion Middleware User Reference for Oracle Identity Management*.

For information about setting up SSL for Oracle Internet Directory, see "Configuring Secure Sockets Layer (SSL)" in the *Oracle Fusion Middleware Administrator's Guide for Oracle Virtual Directory* manual.

WebLogic Server Administration Console:

```
http://oidhost1.mycompany.com:7001/console
```

Oracle Enterprise Manager Fusion Middleware Console:

```
http://oidhost1.mycompany.com:7001/em
```

7.3.5 Oracle Internet Directory Failover and Expected Behavior

This section includes steps for performing a failover of Oracle Internet Directory and for performing a failover of RAC.

7.3.5.1 Performing an Oracle Internet Directory Failover

Follow these steps to perform a failover of an Oracle Internet Directory instance and to check the status of Oracle Internet Directory:

1. On OIDHOST1, use the `opmnctl` command to stop the Oracle Internet Directory instance:

   ```
   ORACLE_INSTANCE/bin/opmnctl stopproc ias-component=oid1
   ```

2. On OIDHOST2, check the status of Oracle Internet Directory using the load balancing router:

   > **Note:** See the "Configuring Your Environment" section of *Oracle Fusion Middleware User Reference for Oracle Identity Management* for a list of the environment variables you must set before using the `ldapbind` command.

   ```
   ldapbind -h oid.mycompany.com -p 389 -D "cn=orcladmin" -q
   ```

   > **Note:** The -q option above prompts the user for a password. LDAP tools have been modified to disable the options -w *password* and -P *password* when the environment variable LDAP_PASSWORD_ PROMPTONLY is set to TRUE or 1. Use this feature whenever possible.

3. On OIDHOST1, use the `opmnctl` command to start the Oracle Internet Directory instance:

   ```
   ORACLE_INSTANCE/bin/opmnctl start (if OPMN is not running)
   ORACLE_INSTANCE/bin/opmnctl startproc ias-component=oid1
   ```

4. On OIDHOST2, use the `opmnctl` command to stop the Oracle Internet Directory instance:

   ```
   ORACLE_INSTANCE/bin/opmnctl stopproc ias-component=oid1
   ```

5. On OIDHOST1, check the status of Oracle Internet Directory using the load balancing router:

   ```
   ldapbind -h oid.mycompany.com -p 389 -D "cn=orcladmin" -q
   ```

6. On OIDHOST2, use the `opmnctl` command to start the Oracle Internet Directory instance:

   ```
   ORACLE_INSTANCE/bin/opmnctl start (if OPMN is not running)
   ORACLE_INSTANCE/bin/opmnctl startproc ias-component=oid1
   ```

7.3.5.2 Performing a RAC Failover

Follow these steps to perform a RAC failover:

1. Use the `srvctl` command to stop a database instance:

   ```
   srvctl stop instance -d db_unique_name -i inst_name_list
   ```

2. Use the `srvctl` command to check the status of the database:

   ```
   srvctl status database -d db_unique_name -v
   ```

3. Check the status of Oracle Internet Directory:

Note: See the "Configuring Your Environment" section of *Oracle Fusion Middleware User Reference for Oracle Identity Management* for a list of the environment variables you must set before using the `ldapbind` command.

```
ldapbind -h oid_host -p 389 -D "cn=orcladmin" -q
ldapbind -h oid_host -p 389 -D "cn=orcladmin" -q
ldapbind -h oid.mycompany.com -p 389 -D "cn=orcladmin" -q
```

Note: The -q option above prompts the user for a password. LDAP tools have been modified to disable the options -w *password* and -P *password* when the environment variable LDAP_PASSWORD_PROMPTONLY is set to TRUE or 1. Use this feature whenever possible.

---

4. Use the `srvctl` command to start the database instance:

   ```
   srvctl start instance -d db_unique_name -i inst_name_list
   ```

7.3.6 Troubleshooting Oracle Internet Directory High Availability

This section provides information that can help you troubleshoot Oracle Internet Directory high availability issues:

- Log files for Oracle Internet Directory are found in the following directory:

  ```
  ORACLE_INSTANCE/diagnostics/log/OID
  ```

- The order in which log files should be examined when troubleshooting is:
  1. oidmon-xxx.log
  2. oidldapd01-xxxx.log
  3. oidldapd01s-xxxx.log
- This section shows some of the error messages that may be related to high availability, and their meaning:

  Error: ORA-3112, ORA-3113 errors in the log file

  Cause: one of the database node is down, OID connects again to surviving node.

  Action: See why database node went down or Oracle process got killed

  Error: Failing Over...Please stand by in the log file

  Cause: OID server received a notification from the Oracle process that one of the database node is down. OID will connect to the surviving node.

  If the failover is successful you would see this message:

  Failover ended...resuming services.

  If the failover was not successful, you would see these errors:

  a. Tried 10 times, now quitting from failover function...

  b. Bad Failover Event:

c. Forcing Failover abort as setting of DB parameters for the session failed

If high availability event notification is enabled, you would see a message similar to the following:

```
HA Callback Event
Thread Id: 8
Event type: 0
HA Source: OCI_HA_INSTANCE
Host name: dbhost1
Database name: orcl
Instance name: orcl1
Timestamp: 14-MAY-09 03.25.24 PM -07:00
Service name: orcl.us.oracle.com
HA status: DOWN - TAF Capable
```

If TAF is disabled, HA status will be shown as "DOWN."

Action: See why database node went down.

Error: Time Difference of at least 250 sec found between node1 and node2.

Cause: There is time difference between the two nodes

Action: Synchronize the system time.

Error: Node=% did not respond for configured %d times, Failing over...

Cause: One of the OID nodes (oidmon) is not responding.

Action: See if the node is alive or OIDMON process is running.

For more information about troubleshooting Oracle Internet Directory, see *Oracle Fusion Middleware Administrator's Guide for Oracle Internet Directory*.

7.3.7 Additional Oracle Internet Directory High Availability Issues

This section describes issues for Oracle Internet Directory in a high availability environment.

7.3.7.1 Changing the Password of the ODS Schema Used by Oracle Internet Directory

You can change the Oracle Internet Directory database schema password (that is, the password of the ODS user in the database) using the Oracle Internet Directory Database Password Utility `(oidpasswd)` from any of the Oracle Internet Directory nodes. However, since the ODS schema password is stored in a password wallet under the ORACLE_INSTANCE of each OID instance, the password wallet must be updated in each Oracle Internet Directory node.

To change the ODS database user password, invoke the following command on one of the Oracle Internet Directory nodes:

```
oidpasswd connect=database-connection-string change_oiddb_pwd=true
```

On all other Oracle Internet Directory nodes, invoke the following command to synchronize the password wallet:

```
oidpasswd connect=database-connection-string create_wallet=true
```

If you change the ODS password on one Oracle RAC node by using the OID Database Password Utility (`oidpasswd`), then you must do one of the following to update the wallet *`ORACLE_HOME`*`/ldap/admin/oidpwdlldap1` on the other Oracle RAC nodes:

- Invoke the OID Database Password Utility on all the other nodes to update the wallet file only. This applies to replication password changes also, but for replication password changes you would use the Replication Environment Management Tool to update the password instead of the OID Database Password Utility.
- Copy the changed wallet to the other nodes.

If you run the oidpasswd command on one node only, and do not update the wallet on all the Oracle RAC nodes, the Oracle Internet Directory instance on the second node will not be able to start on the other nodes. You will see this error in the OIDMON log file:

```
[gsdsiConnect] ORA-1017, ORA-01017: invalid username/password; logon denied.
```

As mentioned above, the fix is to copy the `oidpwdlldap1` file to the other Oracle RAC nodes, or to invoke the `oidpasswd` tool with the `create_wallet=true` option on the other nodes.

7.4 Oracle Virtual Directory High Availability

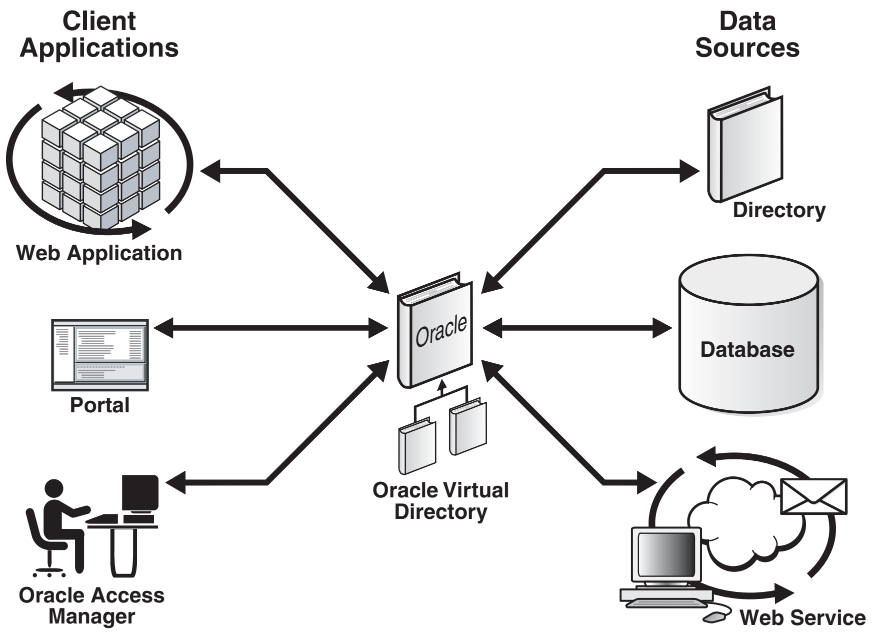
This section provides an introduction to Oracle Virtual Directory and describes how to design and deploy a high availability environment for Oracle Virtual Directory.

7.4.1 Oracle Virtual Directory Component Architecture

Oracle Virtual Directory is an LDAP version 3 enabled service that provides virtualized abstraction of one or more enterprise data sources into a single directory view. Oracle Virtual Directory provides the ability to integrate LDAP-aware applications into diverse directory environments while minimizing or eliminating the need to change either the infrastructure or the applications. Oracle Virtual Directory supports a diverse set of clients, such as Web Applications and portals, and it can connect to directories, databases and Web Services as shown in Figure 7–4.

Figure 7–4 shows Oracle Virtual Directory in a non-high availability architecture.

Figure 7–4 Oracle Virtual Directory in a Non-High Availability Architecture



The Oracle Virtual Directory server is written in Java and internally it is organized into multiple layers. These layers are logical layers—Oracle Virtual Directory appears as a single complete service to the administrator and to clients.

7.4.1.1 Oracle Virtual Directory Runtime Considerations

OPMN is used to start, monitor, and manage the Oracle Virtual Directory process, and to restart the Oracle Virtual Directory process if it goes down. For information on using OPMNCTL to start and stop Oracle Virtual Directory instances, see *Oracle Fusion Middleware Administrator's Guide for Oracle Virtual Directory*.

OPMN invokes the JVM to start the VDEServer process with the required parameters. JVM parameters are configured in opmn.xml (oracle.security.jps.config is used for the JPS Config File Location, vde.soTimeoutBackend is used to control orphan server connections.

You can also use the Oracle Enterprise Manager Fusion Middleware Control to start and stop Oracle Virtual Directory instances. For information, see *Oracle Fusion Middleware Administrator's Guide for Oracle Virtual Directory*.

Except for JPS, which is installed when Oracle Virtual Directory is installed, Oracle Virtual Directory does not have external dependencies. It can run by itself.

Oracle Virtual Directory can be configured to store LDAP objects in the local file system. This feature can be used by JPS and other components.

Oracle Virtual Directory provides two types of listeners: LDAP and HTTP. Both listeners support SSL/TLS on top of their basic protocols. The LDAP layer also provides the ability to support LDAP-SASL to support digital certificate authentication.

The LDAP(S) protocols provide LDAPv2/v3 based services, and the HTTP(S) protocols provide one or more services such as DSMLv2, or basic white page functions provided by an XSLT enabled Web Gateway.

Based on the nature of the operation, client connections can either be persistent or short-lived.

7.4.1.2 Oracle Virtual Directory Component Characteristics

This section describes the various configuration artifacts for Oracle Virtual Directory. The following Oracle Virtual Directory configuration files are located under `ORACLE_INSTANCE/config/OVD/`*OVDComponentName*:

- server.os_xml:

  Oracle Virtual Directory provides the ability to regulate items such as the number of entries the server can return for an anonymous user or for an authenticated user. You can also limit inbound transaction traffic, which can be used to protect proxied sources from Denial Of Service attacks or to limit LDAP traffic to control access to a limited directory infrastructure resource. These properties and others are configured in server.os_xml.

- listeners.os_xml:

  Oracle Virtual Directory provides services to clients through connections known as Listeners. Oracle Virtual Directory supports two types of Listeners: LDAP and HTTP. An Oracle Virtual Directory configuration can have any number of listeners or it can even have zero Listeners, thus restricting access to only the administrative gateway. Most Oracle Virtual Directory deployments will need no more than two HTTP Listeners and two LDAP Listeners, where one Listener is for SSL and one for non-SSL for each protocol. The Listener configuration file is listeners.os_xml.

- adapters.os_xml:

  To present the single virtual directory view of data in multiple and various data repositories, Oracle Virtual Directory must connect to those repositories so it can virtualize the data and route data to and from the repositories. Oracle Virtual Directory uses adapters to connect to its underlying data repositories. Each adapter manages a namespace in the directory identified by a specific parent distinguished name (DN). There is no limit to how many adapters you can configure. You can also combine and overlap adapters to present a customized directory tree. The adapters configuration file is adapters.os_xml.

  Note: For information on configuring a No-Authorization SSL connection between Oracle Virtual Directory and a proxy LDAP directory, see *Oracle Fusion Middleware Administrator's Guide for Oracle Virtual Directory*. The procedure described in that manual can be used for any proxy LDAP directory configured to support anonymous ciphers.

- acls.os_xml

  Oracle Virtual Directory provides granular access controls that can be applied uniformly across all connected data stores and which are compliant with the Internet Engineering Task Force's RFC 2820, Access Control Requirements for LDAP. The access control rules are modeled on the IETF's internet draft titles LDAP Access Control Model for LDAPv3, (March 2, 2001 draft).

  Oracle Virtual Directory provides virtualized abstraction of one or more enterprise data sources into a single directory view. Accordingly, Access Control Lists (ACLs) and adapter namespaces are independent of each other. Removing all entries in a namespace, or changing the root value of an adapter, will not effect ACLs automatically. ACLs and adapter namespaces must be configured independently of each other. The ACL configuration file is acls.os_xml.

Oracle Virtual Directory instance-specific data is stored in the ORACLE_INSTANCE home. The wallet is also stored in the instance home.

If a single Oracle Virtual Directory instance fails, use OPMN to restart the instance.

7.4.1.2.1 Oracle Virtual Directory Log File The log files for an Oracle Virtual Directory instance are stored in the following directory in the instance home:

```
ORACLE_INSTANCE/diagnostics/logs/OVD/OVDComponentName/
```

For more information on using the Oracle Virtual Directory log files to troubleshoot Oracle Virtual Directory issues, see Section 7.4.6, "Troubleshooting Oracle Virtual Directory High Availability".

7.4.2 Oracle Virtual Directory High Availability Concepts

This section provides conceptual information about using Oracle Virtual Directory in a high availability two-node cluster. See Section 7.4.2.2, "Oracle Virtual Directory Prerequisites" for prerequisites and Section 7.4.3, "Oracle Virtual Directory High Availability Configuration Steps" for specific steps for setting up the two-node cluster.

7.4.2.1 Oracle Virtual Directory High Availability Architecture

Figure 7–5 shows the Oracle Virtual Directory high availability architecture in an active-active configuration.

Figure 7–5 Oracle Virtual Directory in a High Availability Environment

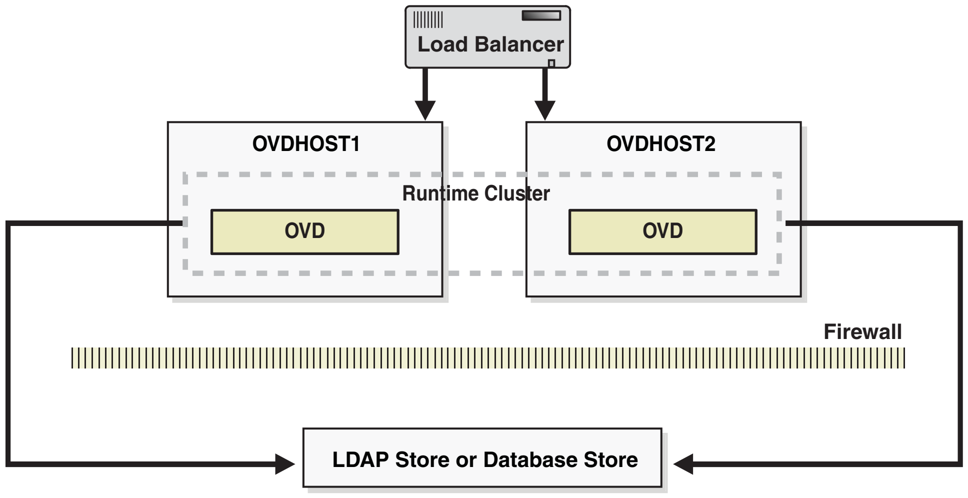


Figure 7–5 shows Oracle Virtual Directory in the directory tier in a high availability architecture. The two-node cluster is set up at installation time. The load balancing router routes requests to the two Oracle Virtual Directory instances that are clustered on OVDHOST1 and OVDHOST2. Fast Connection Failover (FCF) is used for notification when a RAC instance becomes unavailable.

The two computers have the same Oracle Virtual Directory configuration. The Oracle Virtual Directory configuration for each instance is stored in its ORACLE_INSTANCE. Each Oracle Virtual Directory Instance configuration must be updated separately by using Oracle Directory Services Manager to connect to each Oracle Virtual Directory instance one at a time. The alternate approach is to update the configuration of one Oracle Virtual Directory instance and then use cloning to copy the configuration to the other Oracle Virtual Directory instance or instances. See the "Cloning Oracle Fusion Middleware" chapter in *Oracle Fusion Middleware Administrator's Guide* for more information about cloning.

Note: Oracle Directory Services Manager should be used to manage Oracle Virtual Directory configuration. Oracle Directory Services Manager should not connect to Oracle Virtual Directory through the load balancer to perform configuration updates to an Oracle Virtual Directory instance; instead, it should connect explicitly to a physical Oracle Virtual Directory instance to perform a configuration update for that instance.

OPMN is used to start and stop Oracle Virtual Directory instances. When a cluster that includes multiple Oracle Virtual Directory instances is started, there is no impact on the individual Oracle Virtual Directory instances in the cluster.

The load balancing router detects a hang or failure of an Oracle Virtual Directory instance and routes LDAP and HTTP traffic to surviving instances. When the instance becomes available again, the load balancing router detects this and routes traffic to all the surviving instances.

If an instance fails in the middle of a transaction, the transaction is not committed to the back end.

If one Oracle Virtual Directory instance in the two-node Oracle Virtual Directory cluster fails, the load balancing router detects this and reroutes the LDAP client traffic to the surviving instance or instances in the cluster. When the Oracle Virtual Directory instance comes up again, the load balancing router detects this and reroutes the LDAP client traffic instance to the surviving instance or instances in the cluster.

7.4.2.1.1 Oracle Virtual Directory High Availability Connect Features Oracle Virtual Directory offers multiple high availability capabilities, including:

- Fault Tolerance and Failover: Oracle Virtual Directories provide fault tolerance in two forms:
  - They can be configured in fault tolerant configurations.
  - They can manage flow to fault tolerant proxied sources.

  Multiple Oracle Virtual Directories can be quickly deployed simply by copying, or even sharing configuration files. When combined with round-robin DNS, redirector, or cluster technology, Oracle Virtual Directory provides a complete fault-tolerant solution.

  For each proxied directory source, Oracle Virtual Directory can be configured to access multiple hosts (replicas) for any particular source. It intelligently fails over between hosts and spreads the load between them. Flexible configuration options allow administrators to control percentages of a load to be directed towards specific replica nodes and to indicate whether a particular host is a read-only replica or a read-write server (master). This avoids unnecessary referrals resulting from attempts to write to a read-only replica.
- Load Balancing: Oracle Virtual Directory was designed with powerful load balancing features that allow it to spread load and manage failures between its proxied LDAP directory sources.

Oracle Virtual Directory's virtual directory tree capability allows large sets of directory information to be broken up into multiple distinct directory servers. Oracle Virtual Directory is able to recombine the separated data sets back into one virtual tree by combining the separate directory tree branches.

If you have multiple LDAP servers for a particular source, the Oracle Virtual Directory LDAP Adapter can load balance and failover for these servers on its own. This load balancing and failover happens transparently to the client and does not require any additional hardware or changes to the client connecting to Oracle Virtual Directory.

The Database adapter supports load balancing and failover if the underlying JDBC driver provides this functionality. Additionally, Oracle Virtual Directory is certified for use with Oracle Real Application Clusters (RAC). See Section 4.1.5, "JDBC Clients" for more information about using Oracle Virtual Directory with RAC.

Oracle Virtual Directory Routing also provides load balancing capabilities. Routing allows search filters to be included in addition to the search base to determine optimized search targets. In this load balancing approach, Oracle Virtual Directory automatically routes queries to the appropriate virtualized directory sources enabling the ability to work with many millions of directory entries.

Note: Oracle Virtual Directory's value is as a virtualization and proxy service, not as a directory store. If you need a highly available directory storage system, Oracle recommends using Oracle Internet Directory.

The log files for an Oracle Virtual Directory instance are stored in the following directory in the instance home:

```
ORACLE_INSTANCE/diagnostics/logs/OVD/OVDComponentName/
```

For more information on using the Oracle Virtual Directory log files to troubleshoot Oracle Virtual Directory issues, see Section 7.4.6, "Troubleshooting Oracle Virtual Directory High Availability".

7.4.2.2 Oracle Virtual Directory Prerequisites

This section describes prerequisites for Oracle Virtual Directory.

Note: Oracle requires that you synchronize the system clocks on the cluster nodes when you are using Oracle Virtual Directory in a high availability deployment.

7.4.2.2.1 Load Balancer Virtual Server Names for Oracle Virtual Directory When Oracle Virtual Directory is deployed in a high availability configuration, it is recommended to use an external load balancer to front-end the Oracle Virtual Directory instances and load balance the LDAP requests between the various Oracle Virtual Directory instances.

Refer to Section 7.2.4.5, "Configuring Virtual Server Names and Ports for the Load Balancer" for details.

7.4.3 Oracle Virtual Directory High Availability Configuration Steps

Oracle Virtual Directory can be deployed in a highly available configuration either in a standalone mode or as a part of a WebLogic Server domain.

Since Oracle Directory Services Manager and Oracle Enterprise Manager Fusion Middleware Control are required to manage Oracle Virtual Directory effectively, it is recommended that Oracle Virtual Directory be deployed as part of an Oracle WebLogic Server domain. For information on configuring Oracle Virtual Directory and Oracle Directory Services Manager in a collocated configuration, see Section 7.8, "Collocated Architecture High Availability."

In a high availability environment, it is recommended that Oracle Virtual Directory be set up in a clustered deployment, where the clustered Oracle Virtual Directory instances access the same RAC database repository or LDAP repository.

7.4.3.1 Configuring Oracle Virtual Directory Without a WebLogic Domain

This section describes the steps to deploy Oracle Virtual Directory without an Oracle WebLogic Server domain.

Note: When Oracle Virtual Directory is installed on a host using the **Configure without a Domain** option in the Oracle Identity Management 11*g* installer, by default the installer uses `ovd1` as the component name for the Oracle Virtual Directory instance.

During a **Configure without a Domain** installation, the installer cannot detect if another Oracle Virtual Directory instance with a component name of `ovd1` already exists on the host and is registered with a domain.

The Oracle Virtual Directory instances installed on OVDHOST1 and OVDHOST2 in Section 7.4.3.1.1, "Installing Oracle Virtual Directory on OVDHOST1" and Section 7.4.3.1.2, "Installing Oracle Virtual Directory on OVDHOST2" are installed using the **Configure without a Domain** installation option.

7.4.3.1.1 Installing Oracle Virtual Directory on OVDHOST1 Make sure that the schema database is running, then follow these steps to install the Oracle Virtual Directory instance on OVDHOST1:

1. Ensure that the system, patch, kernel and other requirements are met. These are listed in *Oracle Fusion Middleware Installation Guide for Oracle Identity Management* in the Oracle Fusion Middleware documentation library for the platform and version you are using.
2. Ensure that ports 6501 and 7501 are not in use by any service on the computer by issuing these commands for the operating system you are using. If a port is not in use, no output is returned from the command.

   On UNIX:

   ```
   netstat -an | grep "6501"

   netstat -an | grep "7501"
   ```

   On Windows:

   ```
   netstat -an | findstr :6501
   ```

```
netstat -an | findstr :7501
```

3. If the port is in use (if the command returns output identifying the port), you must free the port.

   On UNIX:

   Remove the entries for ports 6501 and 7501 in the `/etc/services` file and restart the services, or restart the computer.

   On Windows:

   Stop the component that is using the port.

4. Copy the `staticports.ini` file from the `Disk1/stage/Response` directory to a temporary directory.

5. Edit the `staticports.ini` file that you copied to the temporary directory to assign the following custom ports (uncomment the lines where you specify the port numbers for Oracle Virtual Directory):

```
# The non-SSL port for Oracle Virtual Directory
Oracle Virtual Directory port = 6501
# The SSL port for Oracle Virtual Directory
Oracle Virtual Directory (SSL) port = 7501
```

6. Start the Oracle Identity Management 11*g* Installer as follows:

   On UNIX, issue this command: **`runInstaller`**

   On Windows, double-click **`setup.exe`**

   The `runInstaller` and `setup.exe` files are in the `../install/`*`platform`* directory, where *platform* is a platform such as Linux or Win32.

   This displays the Specify Oracle Inventory screen.

7. On the Specify Inventory Directory screen, enter values for the Oracle Inventory Directory and the Operating System Group Name. For example:

   Specify the Inventory Directory: `/u01/app/oraInventory`

   Operating System Group Name: `oinstall`

   A dialog box appears with the following message:

   "Certain actions need to be performed with root privileges before the install can continue. Please execute the script /u01/app/oraInventory/createCentralInventory.sh now from another window and then press "Ok" to continue the install. If you do not have the root privileges and wish to continue the install select the "Continue installation with local inventory" option"

   Log in as root and run the "/u01/app/oraInventory/createCentralInventory.sh"

   This sets the required permissions for the Oracle Inventory Directory and then brings up the Welcome screen.

Note: The Oracle Inventory screen is not shown if an Oracle product was previously installed on the host. If the Oracle Inventory screen is not displayed for this installation, make sure to check and see:

1. If the `/etc/oraInst.loc` file exists
2. If the file exists, the Inventory directory listed is valid
3. The user performing the installation has write permissions for the Inventory directory

---

8. On the Welcome screen, click **Next**.
9. On the Select Installation Type screen, select **Install and Configure** and then click **Next**.
10. On the Prerequisite Checks screen, the installer completes the prerequisite check. If any fail, fix them and restart your installation.

    Click **Next**.
11. On the Select Domain screen, select **Configure without a Domain** and then click **Next**.
12. On the Specify Installation Location screen, specify the following values:
    - **Oracle Home Location:**

      ```
      /u01/app/oracle/product/fmw/idm_1
      ```
    - **Oracle Instance Location:**

      ```
      /u01/app/oracle/admin/ora_inst1
      ```
    - **Oracle Instance Name**:

      ```
      ora_inst1
      ```

    Note: Ensure that the Oracle Home Location directory path for OVDHOST1 is the same as the Oracle Home Location directory path for OVDHOST2. For example, if the Oracle Home Location directory path for OVDHOST1 is:

    ```
    /u01/app/oracle/product/fmw/idm_1
    ```

    then the Oracle Home Location directory path for OVDHOST2 must be:

    ```
    /u01/app/oracle/product/fmw/idm_1
    ```

    Click **Next**.
13. On the Specify Email for Security Updates screen, specify these values:
    - **Email Address**: Provide the email address for your My Oracle Support account.
    - **Oracle Support Password**: Provide the password for your My Oracle Support account.

- Check the checkbox next to the **I wish to receive security updates via My Oracle Support** field.

  Click **Next**.

14. On the Configure Components screen, select **Oracle Virtual Directory** and click **Next**.
15. On the Configure Ports screen, select **Specify Ports Using Configuration File** and enter the filename for the `staticports.ini` file that you copied to the temporary directory.

    Click **Next**.

16. Specify the following values on the Specify Virtual Directory screen:

    In the Client Listeners section, enter:

    - **LDAP v3 Name Space**: Enter the name space for Oracle Virtual Directory. The default value is `dc=us,dc=mycompany,dc=com`.

      ```
      dc=us,dc=mycompany,dc=com
      ```

    - **HTTP Web Gateway**: Select this option to enable the Oracle Virtual Directory HTTP Web Gateway.
    - **Secure**: Select this option if you enabled the HTTP Web Gateway and you want to secure it using SSL.

    In the OVD Administrator section, enter:

    - **Administrator User Name**: Enter the user name for the Oracle Virtual Directory administrator, for example: `cn=orcladmin`
    - **Password**: Enter the password for the Oracle Virtual Directory administrator. For example: `*******`
    - **Confirm Password**: Confirm the password for the Oracle Virtual Directory administrator. For example: `*******`
    - **Configure the Administrative Server in secure mode**: Select this option to secure the Oracle Virtual Directory Administrative Listener using SSL. This option is selected by default. Oracle recommends selecting this option.

    Click **Next**.

17. On the Installation Summary screen, review the selections to ensure that they are correct (if they are not, click **Back** to modify selections on previous screens), and click **Install**.
18. On the Installation Progress screen on UNIX systems, a dialog box appears that prompts you to run the `oracleRoot.sh` script. Open a window and run the script, following the prompts in the window.

    Click **Next**.

19. On the Configuration screen, multiple configuration assistants are launched in succession; this process can be lengthy. When it completes, click **Next**.
20. On the Installation Complete screen, click **Finish** to confirm your choice to exit.

7.4.3.1.2 Installing Oracle Virtual Directory on OVDHOST2

Make sure that the schema database is running, then follow these steps to install the Oracle Virtual Directory instance on OVDHOST2:

Note: The instructions in this section can also be used to scale out Oracle Virtual Directory in your 11*g* Oracle Identity Management high availability configuration.

1. Ensure that the system, patch, kernel and other requirements are met. These are listed in *Oracle Fusion Middleware Installation Guide for Oracle Identity Management* in the Oracle Fusion Middleware documentation library for the platform and version you are using.
2. On OVDHOST1, ports 6501 and 7501 were used for Oracle Virtual Directory. The same ports should be used for the Oracle Virtual Directory instance on OVDHOST2. Therefore, ensure that ports 6501 and 7501 are not in use by any service on OVDHOST2 by issuing these commands for the operating system you are using. If a port is not in use, no output is returned from the command.

   On UNIX:

   ```
   netstat -an | grep "6501"

   netstat -an | grep "7501"
   ```

   On Windows:

   ```
   netstat -an | findstr :6501

   netstat -an | findstr :7501
   ```

3. If the port is in use (if the command returns output identifying the port), you must free the port.

   On UNIX:

   Remove the entries for ports 6501 and 7501 in the `/etc/services` file and restart the services, or restart the computer.

   On Windows:

   Stop the component that is using the port.
4. Copy the `staticports.ini` file from the `Disk1/stage/Response` directory to a temporary directory.
5. Edit the `staticports.ini` file that you copied to the temporary directory to assign the following custom ports (uncomment the lines where you specify the port numbers for Oracle Virtual Directory):

   ```
   # The non-SSL port for Oracle Virtual Directory
   Oracle Virtual Directory port = 6501
   # The SSL port for Oracle Virtual Directory
   Oracle Virtual Directory (SSL) port = 7501
   ```

6. Start the Oracle Identity Management 11*g* Installer as follows:

   On UNIX, issue this command: **`runInstaller`**

   On Windows, double-click **`setup.exe`**

   The `runInstaller` and `setup.exe` files are in the `../install/`*`platform`* directory, where *platform* is a platform such as Linux or Win32.

   This displays the Specify Oracle Inventory screen.

7. On the Specify Inventory Directory screen, enter values for the Oracle Inventory Directory and the Operating System Group Name. For example:

   Specify the Inventory Directory: `/u01/app/oraInventory`

   Operating System Group Name: `oinstall`

   A dialog box appears with the following message:

   "Certain actions need to be performed with root privileges before the install can continue. Please execute the script /u01/app/oraInventory/createCentralInventory.sh now from another window and then press "Ok" to continue the install. If you do not have the root privileges and wish to continue the install select the "Continue installation with local inventory" option"

   Log in as root and run the "/u01/app/oraInventory/createCentralInventory.sh"

   This sets the required permissions for the Oracle Inventory Directory and then brings up the Welcome screen.

   Note: The Oracle Inventory screen is not shown if an Oracle product was previously installed on the host. If the Oracle Inventory screen is not displayed for this installation, make sure to check and see:

   1. If the `/etc/oraInst.loc` file exists
   2. If the file exists, the Inventory directory listed is valid
   3. The user performing the installation has write permissions for the Inventory directory

8. On the Welcome screen, click **Next**.
9. On the Select Installation Type screen, select **Install and Configure** and then click **Next**.
10. On the Prerequisite Checks screen, the installer completes the prerequisite check. If any fail, fix them and restart your installation.

    Click **Next**.
11. On the Select Domain screen, select **Configure without a Domain** and then click **Next**.
12. On the Specify Installation Location screen, specify the following values:

    - **Oracle Home Location:**

      `/u01/app/oracle/product/fmw/idm_1`

    - **Oracle Instance Location:**

      `/u01/app/oracle/admin/ora_inst2`

    - **Oracle Instance Name**:

      `ora_inst2`

Note: Ensure that the Oracle Home Location directory path for OVDHOST2 is the same as the Oracle Home Location directory path for OVDHOST1. For example, if the Oracle Home Location directory path for OVDHOST1 is:

```
/u01/app/oracle/product/fmw/idm_1
```

then the Oracle Home Location directory path for OVDHOST2 must be:

```
/u01/app/oracle/product/fmw/idm_1
```

Click **Next**.

13. On the Specify Email for Security Updates screen, specify these values:

    - **Email Address**: Provide the email address for your My Oracle Support account.
    - **Oracle Support Password**: Provide the password for your My Oracle Support account.
    - Check the checkbox next to the **I wish to receive security updates via My Oracle Support** field.

    Click **Next**.

14. On the Configure Components screen, select **Oracle Virtual Directory** and click **Next**.

15. On the Configure Ports screen, select **Specify Ports Using Configuration File** and enter the filename for the `staticports.ini` file that you copied to the temporary directory.

    Click **Next**.

16. Specify the following values on the Specify Virtual Directory screen:

    In the Client Listeners section, enter:

    - **LDAP v3 Name Space**: Enter the name space for Oracle Virtual Directory. The default value is `dc=us,dc=mycompany,dc=com`.

      ```
      dc=us,dc=mycompany,dc=com
      ```

    - **HTTP Web Gateway**: Select this option to enable the Oracle Virtual Directory HTTP Web Gateway.
    - **Secure**: Select this option if you enabled the HTTP Web Gateway and you want to secure it using SSL.

    In the OVD Administrator section, enter:

    - **Administrator User Name**: Enter the user name for the Oracle Virtual Directory administrator, for example: `cn=orcladmin`
    - **Password**: Enter the password for the Oracle Virtual Directory administrator. For example: `*******`
    - **Confirm Password**: Confirm the password for the Oracle Virtual Directory administrator. For example: `*******`

- **Configure the Administrative Server in secure mode**: Select this option to secure the Oracle Virtual Directory Administrative Listener using SSL. This option is selected by default. Oracle recommends selecting this option.

   Click **Next**.

17. On the Installation Summary screen, review the selections to ensure that they are correct (if they are not, click **Back** to modify selections on previous screens), and click **Install**.
18. On the Installation Progress screen on UNIX systems, a dialog box appears that prompts you to run the `oracleRoot.sh` script. Open a window and run the script, following the prompts in the window.

    Click **Next**.
19. On the Configuration screen, multiple configuration assistants are launched in succession; this process can be lengthy. When it completes, click **Next**.
20. On the Installation Complete screen, click **Finish** to confirm your choice to exit.

7.4.3.1.3 Registering Oracle Virtual Directory with a WebLogic Domain It is recommended that you manage your Oracle Virtual Directory component with Oracle Enterprise Manager Fusion Middleware Control. To be able to manage Oracle Virtual Directory using Oracle Enterprise Manager Fusion Middleware Control, you must register the component and the Oracle Fusion Middleware instance that contains it with an Oracle WebLogic Server domain. You can register an Oracle Fusion Middleware instance with a WebLogic domain during installation or Oracle instance creation, but you are not required to do so. If an Oracle Fusion Middleware instance was not previously registered with a WebLogic domain, you can register it by using `opmnctl registerinstance`.

Before using the `opmnctl registerinstance` command to register an Oracle Virtual Directory instance with an Oracle WebLogic Server domain, make sure that the WebLogic Server is already installed.

Then execute the `opmnctl registerinstance` command in this format (note that the ORACLE_HOME and ORACLE_INSTANCE variables have to be set for each installation before executing this command in the home directory for the Oracle Virtual Directory instance):

```
opmnctl registerinstance -adminHost WLSHostName -adminPort WLSPort
-adminUsername adminUserName
```

For example:

```
opmnctl registerinstance -adminHost idmhost1.mycompany.com -adminPort 7001
-adminUsername weblogic

Command requires login to weblogic admin server (idmhost1.mycompany.com)
 Username: weblogic
 Password: *******
```

For additional details on registering Oracle Virtual Directory components with a WebLogic domain, see the "Registering an Oracle Instance Using OPMNCTL" section in *Oracle Fusion Middleware Administrator's Guide for Oracle Virtual Directory*.

7.4.3.2 Configuring Oracle Virtual Directory With a WebLogic Domain

This section describes the steps to deploy Oracle Virtual Directory in a high availability configuration as part of a WebLogic Server domain.

In this configuration, Oracle Virtual Directory and a WebLogic Server domain is configured on the first host, and only Oracle Virtual Directory is configured on the second host. The Oracle Virtual Directory instance on the second host joins the domain created on the first host.

7.4.3.2.1 Installing Oracle WebLogic Server On OVDHOST1, start the Oracle WebLogic Server installation by running the installer executable file.

Start the Oracle WebLogic Server installer as follows:

- On UNIX (Linux in the following example):

  ```
  ./server103_linux32.bin
  ```

  On Windows:

  ```
  server103_win32.exe
  ```

Then follow these steps in the installer to install Oracle WebLogic Server on the computer:

1. On the Welcome screen, click **Next**.
2. On the Choose Middleware Home Directory screen, choose a directory on your computer into which the Oracle WebLogic software is to be installed.

   For the **Middleware Home Directory**, specify this value:

   ```
   /u01/app/oracle/product/fmw
   ```

   Click **Next**.
3. On the Register for Security Updates screen, enter your "My Oracle Support" User Name and Password.
4. On the Choose Install Type screen, t.he installation program displays a window in which you are prompted to indicate whether you wish to perform a complete or a custom installation

   Choose **Typical** or **Custom**.

   Click **Next**.
5. On the Choose Product Installation Directories screen, specify the following value:

   WebLogic Server:

   ```
   /u01/app/oracle/product/fmw/wlserver_10.3
   ```

   Click **Next**.
6. On the Installation Summary screen, the window contains a list of the components you selected for installation, along with the approximate amount of disk space to be used by the selected components once installation is complete.

   Click **Next**.
7. On the Installation Complete screen, deselect the **Run Quickstart** checkbox.

   Click **Done**.

7.4.3.2.2 Installing the First Oracle Virtual Directory The schema database must be running before you perform this task. Follow these steps to install the Oracle Virtual Directory instance on OVDHOST1:

1. Ensure that the system, patch, kernel and other requirements are met. These are listed in *Oracle Fusion Middleware Installation Guide for Oracle Identity Management* in the Oracle Fusion Middleware documentation library for the platform and version you are using.
2. Ensure that ports 6501 and 7501 are not in use by any service on OVDHOST1 by issuing these commands for the operating system you are using. If a port is not in use, no output is returned from the command.

   On UNIX:

   ```
   netstat -an | grep "6501"

   netstat -an | grep "7501"
   ```

   On Windows:

   ```
   netstat -an | findstr :6501

   netstat -an | findstr :7501
   ```

3. If the port is in use (if the command returns output identifying the port), you must free the port.

   On UNIX:

   Remove the entries for ports 6501 and 7501 in the `/etc/services` file and restart the services, or restart the computer.

   On Windows:

   Stop the component that is using the port.
4. Copy the `staticports.ini` file from the `Disk1/stage/Response` directory to a temporary directory.
5. Edit the `staticports.ini` file that you copied to the temporary directory to assign the following custom ports (uncomment the lines where you specify the port numbers for Oracle Virtual Directory):

   ```
   # The non-SSL port for Oracle Virtual Directory
   Oracle Virtual Directory port = 6501
   # The SSL port for Oracle Virtual Directory
   Oracle Virtual Directory (SSL) port = 7501
   ```

6. Start the Oracle Identity Management 11*g* Installer as follows:

   On UNIX, issue this command: **`runInstaller`**

   On Windows, double-click **`setup.exe`**

   The `runInstaller` and `setup.exe` files are in the `../install/`*`platform`* directory, where *platform* is a platform such as Linux or Win32.

   This displays the Specify Oracle Inventory screen.
7. On the Specify Inventory Directory screen, enter values for the Oracle Inventory Directory and the Operating System Group Name. For example:

   Specify the Inventory Directory: `/u01/app/oraInventory`

   Operating System Group Name: `oinstall`

   A dialog box appears with the following message:

"Certain actions need to be performed with root privileges before the install can continue. Please execute the script /u01/app/oraInventory/createCentralInventory.sh now from another window and then press "Ok" to continue the install. If you do not have the root privileges and wish to continue the install select the "Continue installation with local inventory" option"

Log in as root and run the "/u01/app/oraInventory/createCentralInventory.sh"

This sets the required permissions for the Oracle Inventory Directory and then brings up the Welcome screen.

Note: The Oracle Inventory screen is not shown if an Oracle product was previously installed on the host. If the Oracle Inventory screen is not displayed for this installation, make sure to check and see:

1. If the `/etc/oraInst.loc` file exists
2. If the file exists, the Inventory directory listed is valid
3. The user performing the installation has write permissions for the Inventory directory

---

8. On the Welcome screen, click **Next**.
9. On the Select Installation Type screen, select **Install and Configure** and then click **Next**.
10. On the Prerequisite Checks screen, the installer completes the prerequisites check. If any fail, please fix them and restart your installation.

    Click **Next**.
11. On the Select Domain screen, select **Create New Domain** and specify the following values:
    - **User Name**: weblogic
    - **Password**: <password for the weblogic user>
    - **Confirm Password**: <Confirm the password for the weblogic user>
    - **Domain Name**: IDMDomain
12. On the Specify Installation Location screen, specify the following values:
    - **Oracle Middleware Home Location**:

      ```
      /u01/app/oracle/product/fmw
      ```
    - **Oracle Home Directory**: idm
    - **WebLogic Server Directory:**

      ```
      /u01/app/oracle/product/fmw/wlserver_10.3
      ```
    - **Oracle Instance Location:**

      ```
      /u01/app/oracle/admin/ovd_inst1
      ```
    - **Oracle Instance Name**:

      ```
      ovd_inst1
      ```

Note: Ensure that the Oracle Home Location directory path for OVDHOST1 is the same as the Oracle Home Location directory path for OVDHOST2. For example, if the Oracle Home Location directory path for OVDHOST1 is:

```
/u01/app/oracle/product/fmw/idm
```

then the Oracle Home Location directory path for OVDHOST2 must be:

```
/u01/app/oracle/product/fmw/idm
```

Click **Next**.

13. On the Specify Email for Security Updates screen, specify these values:

    - **Email Address**: Provide the email address for your My Oracle Support account.
    - **Oracle Support Password**: Provide the password for your My Oracle Support account.
    - Check the checkbox next to the **I wish to receive security updates via My Oracle Support** field.

    Click **Next**.

14. On the Configure Components screen, select **Oracle Virtual Directory** and click **Next**.

Note: The Oracle Directory Services Manager and Fusion Middleware Control management components are automatically selected for this installation. You cannot deselect these choices.

---

15. On the Configure Ports screen, select **Specify Ports Using Configuration File** and enter the filename for the `staticports.ini` file that you copied to the temporary directory.

    Click **Next**.

16. Specify the following values on the Specify Virtual Directory screen:

    In the Client Listeners section, enter:

    - **LDAP v3 Name Space**: Enter the name space for Oracle Virtual Directory. The default value is `dc=us,dc=mycompany,dc=com`.

      ```
      dc=us,dc=mycompany,dc=com
      ```

    - **HTTP Web Gateway**: Select this option to enable the Oracle Virtual Directory HTTP Web Gateway.
    - **Secure**: Select this option if you enabled the HTTP Web Gateway and you want to secure it using SSL.

    In the OVD Administrator section, enter:

    - **Administrator User Name**: Enter the user name for the Oracle Virtual Directory administrator, for example: `cn=orcladmin`
    - **Password**: Enter the password for the Oracle Virtual Directory administrator. For example: `*******`

- **Confirm Password**: Confirm the password for the Oracle Virtual Directory administrator. For example: `*******`
- **Configure the Administrative Server in secure mode**: Select this option to secure the Oracle Virtual Directory Administrative Listener using SSL. This option is selected by default. Oracle recommends selecting this option.

Click **Next**.

17. On the Installation Summary screen, review the selections to ensure that they are correct (if they are not, click **Back** to modify selections on previous screens), and click **Install**.
18. On the Installation Progress screen on UNIX systems, a dialog box appears that prompts you to run the `oracleRoot.sh` script. Open a window and run the script, following the prompts in the window.

    Click **Next**.
19. On the Configuration screen, multiple configuration assistants are launched in succession; this process can be lengthy. When it completes, click **Next**.
20. On the Installation Complete screen, click **Finish** to confirm your choice to exit.

7.4.3.2.3 Creating boot.properties for the Administration Server on OVDHOST1 This section describes how to create a `boot.properties` file for the Administration Server on OVDHOST1. The `boot.properties` file enables the Administration Server to start without prompting for the `administrator` username and password.

Follow these steps to create the `boot.properties` file:

1. On OVDHOST1, go the following directory:

   ```
   MW_HOME/user_projects/domains/domainName/servers/AdminServer/security
   ```

   For example:

   ```
   cd /u01/app/oracle/product/fmw/user_
   projects/domains/IDMDomain/servers/AdminServer/security
   ```

2. Use a text editor to create a file called `boot.properties` under the `security` directory. Enter the following lines in the file:

   ```
   username=adminUser
   password=adminUserPassword
   ```

   Note: When you start the Administration Server, the username and password entries in the file get encrypted.

   For security reasons, minimize the time the entries in the file are left unencrypted. After you edit the file, you should start the server as soon as possible so that the entries get encrypted.

3. Stop the Administration Server if it is running.

   See the "Starting and Stopping Oracle Fusion Middleware" chapter of the *Oracle Fusion Middleware Administrator's Guide* for information on starting and stopping WebLogic Servers.

4. Start the Administration Server on OVDHOST1 using the startWebLogic.sh script located under the *MW_HOME*/user_projects/domains/*domainName*/bin directory.

5. Validate that the changes were successful by opening a web browser and accessing the following pages:

   - WebLogic Server Administration Console at:

     ```
     http://oidhost1.mycompany.com:7001/console
     ```

   - Oracle Enterprise Manager Fusion Middleware Control at:

     ```
     http://oidhost1.mycompany.com:7001/em
     ```

   Log into these consoles using the `weblogic` user credentials.

7.4.3.2.4 Installing an Additional Oracle Virtual Directory

The schema database must be running before you perform this task. Follow these steps to install the Oracle Virtual Directory instance on OVDHOST2:

1. Ensure that the system, patch, kernel and other requirements are met. These are listed in *Oracle Fusion Middleware Installation Guide for Oracle Identity Management* in the Oracle Fusion Middleware documentation library for the platform and version you are using.

2. On OVDHOST1, ports 6501 and 7501 were used for Oracle Virtual Directory. The same ports should be used for the Oracle Virtual Directory instance on OVDHOST2. Therefore, ensure that ports 6501 and 7501 are not in use by any service on OVDHOST2 by issuing these commands for the operating system you are using. If a port is not in use, no output is returned from the command.

   On UNIX:

   ```
   netstat -an | grep "6501"

   netstat -an | grep "7501"
   ```

   On Windows:

   ```
   netstat -an | findstr :6501

   netstat -an | findstr :7501
   ```

3. If the port is in use (if the command returns output identifying the port), you must free the port.

   On UNIX:

   Remove the entries for ports 6501 and 7501 in the `/etc/services` file and restart the services, or restart the computer.

   On Windows:

   Stop the component that is using the port.

4. Copy the `staticports.ini` file from the `Disk1/stage/Response` directory to a temporary directory.

5. Edit the `staticports.ini` file that you copied to the temporary directory to assign the following custom ports (uncomment the lines where you specify the port numbers for Oracle Virtual Directory):

   ```
   # The non-SSL port for Oracle Virtual Directory
   ```

```
Oracle Virtual Directory port = 6501
# The SSL port for Oracle Virtual Directory
Oracle Virtual Directory (SSL) port = 7501
```

6. Start the Oracle Identity Management 11*g* Installer as follows:

   On UNIX, issue this command: **`runInstaller`**

   On Windows, double-click **`setup.exe`**

   The `runInstaller` and `setup.exe` files are in the `../install/`*`platform`* directory, where *platform* is a platform such as Linux or Win32.

   This displays the Specify Oracle Inventory screen.

7. On the Specify Inventory Directory screen, enter values for the Oracle Inventory Directory and the Operating System Group Name. For example:

   Specify the Inventory Directory: `/u01/app/oraInventory`

   Operating System Group Name: `oinstall`

   A dialog box appears with the following message:

   "Certain actions need to be performed with root privileges before the install can continue. Please execute the script /u01/app/oraInventory/createCentralInventory.sh now from another window and then press "Ok" to continue the install. If you do not have the root privileges and wish to continue the install select the "Continue installation with local inventory" option"

   Log in as root and run the "/u01/app/oraInventory/createCentralInventory.sh"

   This sets the required permissions for the Oracle Inventory Directory and then brings up the Welcome screen.

   Note: The Oracle Inventory screen is not shown if an Oracle product was previously installed on the host. If the Oracle Inventory screen is not displayed for this installation, make sure to check and see:

   1. If the `/etc/oraInst.loc` file exists
   2. If the file exists, the Inventory directory listed is valid
   3. The user performing the installation has write permissions for the Inventory directory

8. On the Welcome screen, click **Next**.

9. On the Select Installation Type screen, select **Install and Configure** and then click **Next**.

10. On the Prerequisite Checks screen, the installer completes the prerequisites check. If any fail, please fix them and restart your installation.

    Click **Next**.

11. On the Select Domain screen, select **Extend Existing Domain** and specify the following values:

    - **HostName**: `ovdhost1.mycompany.com` (This is the host where the Oracle WebLogic Administration Server is running.)
    - **Port**: `7001` (This is the Administration Server port.)

- **User Name**: `weblogic`
- **Password**: `<password for the weblogic user>`

12. On the Specify Installation Location screen, specify the following values:
    - **Oracle Middleware Home Location**:

      `/u01/app/oracle/product/fmw`
    - **Oracle Home Directory**: idm
    - **WebLogic Server Directory:**

      `/u01/app/oracle/product/fmw/wlserver_10.3`
    - **Oracle Instance Location:**

      `/u01/app/oracle/admin/ovd_inst2`
    - **Oracle Instance Name**:

      `ovd_inst2`

    Note: Ensure that the Oracle Home Location directory path for OVDHOST1 is the same as the Oracle Home Location directory path for OVDHOST2. For example, if the Oracle Home Location directory path for OVDHOST1 is:

    ```
    /u01/app/oracle/product/fmw/idm
    ```

    then the Oracle Home Location directory path for OVDHOST2 must be:

    ```
    /u01/app/oracle/product/fmw/idm
    ```

    Click **Next**.

13. On the Specify Email for Security Updates screen, specify these values:
    - **Email Address**: Provide the email address for your My Oracle Support account.
    - **Oracle Support Password**: Provide the password for your My Oracle Support account.
    - Check the checkbox next to the **I wish to receive security updates via My Oracle Support** field.

    Click **Next**.

14. On the Configure Components screen, select **Oracle Virtual Directory** and click **Next**.

    Note: The Oracle Directory Services Manager and Fusion Middleware Control management components are automatically selected for the installation.

15. On the Configure Ports screen, select **Specify Ports Using Configuration File** and enter the filename for the `staticports.ini` file that you copied to the temporary directory.

    Click **Next**.

16. Specify the following values on the Specify Virtual Directory screen:

    In the Client Listeners section, enter:

    - **LDAP v3 Name Space**: Enter the name space for Oracle Virtual Directory. The default value is `dc=us,dc=mycompany,dc=com`.

      ```
      dc=us,dc=mycompany,dc=com
      ```

    - **HTTP Web Gateway**: Select this option to enable the Oracle Virtual Directory HTTP Web Gateway.
    - **Secure**: Select this option if you enabled the HTTP Web Gateway and you want to secure it using SSL.

    In the OVD Administrator section, enter:

    - **Administrator User Name**: Enter the user name for the Oracle Virtual Directory administrator, for example: `cn=orcladmin`
    - **Password**: Enter the password for the Oracle Virtual Directory administrator. For example: `*******`
    - **Confirm Password**: Confirm the password for the Oracle Virtual Directory administrator. For example: `*******`
    - **Configure the Administrative Server in secure mode**: Select this option to secure the Oracle Virtual Directory Administrative Listener using SSL. This option is selected by default. Oracle recommends selecting this option.

    Click **Next**.

17. On the Installation Summary screen, review the selections to ensure that they are correct (if they are not, click **Back** to modify selections on previous screens), and click **Install**.

18. On the Installation Progress screen on UNIX systems, a dialog box appears that prompts you to run the `oracleRoot.sh` script. Open a window and run the script, following the prompts in the window.

    Click **Next**.

19. On the Configuration screen, multiple configuration assistants are launched in succession; this process can be lengthy. When it completes, click **Next**.

20. On the Installation Complete screen, click **Finish** to confirm your choice to exit.

7.4.3.3 Configuring Oracle Virtual Directory with Highly Available Data Sources

Oracle Virtual Directory can be configured to use either an Oracle RAC database repository or a highly available LDAP repository as a data source.

This section describes how to configure Oracle Virtual Directory in a high availability environment with a RAC database repository and with an LDAP repository.

Note: When configuring Oracle Virtual Directory in a high availability topology, the configuration steps must be completed on all the nodes individually.

7.4.3.3.1 Configuring Oracle Virtual Directory with a RAC Database In an Oracle Virtual Directory high availability environment with a RAC database used as the repository, you must create and configure an Oracle Virtual Directory Database Adapter.

For introductory information about Oracle Virtual Directory Adaptors, see the "Understanding Oracle Virtual Directory Adapters" section of *Oracle Fusion Middleware Administrator's Guide for Oracle Virtual Directory*.

For information about creating Database Adapters for RAC databases, see the "Creating Database Adapters for RAC Databases" section of *Oracle Fusion Middleware Administrator's Guide for Oracle Virtual Directory*.

For information about configuring Database Adapters, see the "Configuring Database Adapters" section of *Oracle Fusion Middleware Administrator's Guide for Oracle Virtual Directory*.

7.4.3.3.2 Configuring Oracle Virtual Directory with LDAP Oracle Virtual Directory connects to a LDAP repository, such as Oracle Internet Directory, through an LDAP adapter. For high availability environments, the LDAP adapter can be configured by adding the host and port information of the nodes with a load balancing algorithm or by adding the load balancer URL of the LDAP repository.

For introductory information about Oracle Virtual Directory Adaptors, see the "Understanding Oracle Virtual Directory Adapters" section in *Oracle Fusion Middleware Administrator's Guide for Oracle Virtual Directory*.

For information about creating and configuring LDAP Adapters, see the "Configuring LDAP Adapters" section in *Oracle Fusion Middleware Administrator's Guide for Oracle Virtual Directory*.

7.4.4 Validating Oracle Virtual Directory High Availability

Oracle Virtual Directory instances are front-ended by a hardware load balancer in high availability deployments. Requests are load balanced between the Oracle Virtual Directory instances, and in case of a failure (either an instance failure or a host failure), the hardware load balancer detects the failure and routes all requests to the surviving instance or instances.

To test Oracle Virtual Directory high availability, stop one node at a time and connect to the Oracle Virtual Directory instances through the load balancer URL using a tool such as `ldapbind`. The connection should succeed all the time as long as one node is running and the topology is configured correctly.

Use the `ldapbind` command line tool to connect to the Oracle Virtual Directory instances through the load balancer virtual host. The `ldapbind` tool enables you to determine whether you can authenticate a client to a server.

Follow the steps below to validate the high availability setup of Oracle Virtual Directory on OVDHOST1 and OVDHOST2:

1. Configure your environment by following the steps in the "Configuring Your Environment" section of *Oracle Fusion Middleware User Reference for Oracle Identity Management*. That section has a list of the environment variables you must set before using the `ldapbind` command.
2. On OVDHOST1, use the `opmnctl` command to stop the Oracle Virtual Directory instance:

```
ORACLE_INSTANCE/bin/opmnctl stopproc ias-component=ovd1
```

3. On OVDHOST2, check the status of Oracle Virtual Directory by binding to the load balancing virtual address using `ldapbind`:

   Non-SSL:

   ```
   ldapbind -h ovd.mycompany.com -p 6501 -D "cn=orcladmin" -q
   ```

   Note: The -q option above prompts the user for a password. LDAP tools have been modified to disable the options -w *password* and -P *password* when the environment variable LDAP_PASSWORD_PROMPTONLY is set to TRUE or 1. Use this feature whenever possible.

4. A successful bind shows that the load balancer is routing all traffic to OVDHOST2.
5. On OVDHOST1, use the `opmnctl` command to start the Oracle Virtual Directory instance:

   ```
   ORACLE_INSTANCE/bin/opmnctl start (if OPMN is not running)
   ORACLE_INSTANCE/bin/opmnctl startproc ias-component=ovd1
   ```

6. On OVDHOST2, use the `opmnctl` command to stop the Oracle Virtual Directory instance:

   ```
   ORACLE_INSTANCE/bin/opmnctl stopproc ias-component=ovd1
   ```

7. On OVDHOST1, check the status of Oracle Virtual Directory by binding to the load balancing virtual address using `ldapbind`:

   Non-SSL:

   ```
   ldapbind -h ovd.mycompany.com -p 6501 -D "cn=orcladmin" -q
   ```

8. On OVDHOST2, use the `opmnctl` command to start the Oracle Virtual Directory instance:

   ```
   ORACLE_INSTANCE/bin/opmnctl start (if OPMN is not running)
   ORACLE_INSTANCE/bin/opmnctl startproc ias-component=ovd1
   ```

For more information about the `ldapbind` command, see the ldapbind section in *Oracle Fusion Middleware User Reference for Oracle Identity Management*.

7.4.4.1 Validating Oracle Virtual Directory High Availability Using SSL

Oracle Virtual Directory is configured to use the SSL Server Authentication Only Mode by default. When using command line tools like `ldapbind` to validate a connection using connection secured by SSL Server Authentication mode, the server certificate must be stored in an Oracle Wallet. Follow the steps below to perform this task:

1. Create an Oracle Wallet by executing the following command:

   ```
   ORACLE_HOME/bin/orapki wallet create -wallet DIRECTORY_FOR_SSL_WALLET -pwd
   WALLET_PASSWORD
   ```

2. Export the Oracle Virtual Directory server certificate by executing the following command:

   ```
   ORACLE_HOME/jdk/jre/bin/keytool -exportcert -keystore OVD_KEYSTORE_FILE
   -storepass PASSWORD -alias OVD_SERVER_CERT_ALIAS -rfc -file
   OVD_SERVER_CERT_FILE
   ```

3. Add the Oracle Virtual Directory server certificate to the Oracle Wallet by executing the following command:

   ```
   ORACLE_HOME/bin/orapki wallet add -wallet DIRECTORY_FOR_SSL_WALLET
   -trusted_cert -cert OVD_SERVER_CERT_FILE -pwd WALLET_PASSWORD
   ```

4. Follow the instructions shown in Section 7.4.4, "Validating Oracle Virtual Directory High Availability" but use the `ldapbind` command shown below to validate the high availability setup of Oracle Virtual Directory on OVDHOST1 and OVDHOST2. Use the Oracle Wallet from step 3 while executing the following command:

   ```
   ORACLE_HOME/bin/ldapbind -D cn=orcladmin -q -U 2 -h HOST -p SSL_PORT -W
   "file://DIRECTORY_FOR_SSL_WALLET" -q
   ```

5. When an Oracle Virtual Directory high availability deployment is front ended by a hardware load balancer, the wallets on all the Oracle Virtual Directory nodes must contain the client certificates of all the Oracle Virtual Directory instances that are a part of that topology. Add the client certificates of all the Oracle Virtual Directory instances in the topology to the wallets on all the nodes in the topology. This ensures that a valid connection request made through the load balancer URL does not fail.

Note: If you are using default settings after installing 11*g* Release 1 (11.1.1), you can use the following values for the variables described in this section:

- For *OVD_KEYSTORE_FILE*, use:

  ```
  ORACLE_INSTANCE/config/OVD/ovd1/keystores/keys.jks
  ```

- For *OVD_SERVER_CERT_ALIAS*, use `serverselfsigned`
- For *PASSWORD* used for the `-storepass` option, use the orcladmin account password.

7.4.5 Oracle Virtual Directory Failover and Expected Behavior

This section includes steps for performing a failover of Oracle Virtual Directory and for performing a failover of RAC.

7.4.5.1 Performing an Oracle Virtual Directory Failover

Follow these steps to perform a failover of an Oracle Virtual Directory instance and to check the status of Oracle Virtual Directory:

1. Use the `opmnctl` command to stop the Oracle Virtual Directory instance:

   ```
   ORACLE_INSTANCE/bin/opmnctl stopproc ias-component=ovd1
   ```

2. Use the `opmnctl` command to start the Oracle Virtual Directory instance:

   ```
   ORACLE_INSTANCE/bin/opmnctl start (if OPMN is not running)
   ORACLE_INSTANCE/bin/opmnctl startproc ias-component=ovd1
   ```

3. Check the status of Oracle Virtual Directory:

   ```
   ORACLE_INSTANCE/bin/opmnctl status ias-component=ovd1
   ```

7.4.5.2 Performing a RAC Failover

Follow these steps to perform a RAC failover:

1. Use the `srvctl` command to stop a database instance:

   ```
   srvctl stop instance -d db_unique_name -i inst_name_list
   ```

2. Use the `srvctl` command to check the status of the database:

   ```
   srvctl status database -d db_unique_name -v
   ```

3. Check the status of Oracle Virtual Directory:

   ```
   ORACLE_INSTANCE/bin/opmnctl status ias-component=ovd1
   ```

4. Use the `srvctl` command to start the database instance:

   ```
   srvctl start instance -d db_unique_name -i inst_name_list
   ```

Note: When Oracle Virtual Directory is configured with a Database Adapter against an Oracle RAC database, the first search performed after the RAC failover fails. However, subsequent search requests succeed without any issues.

7.4.6 Troubleshooting Oracle Virtual Directory High Availability

Information in the Oracle Virtual Directory log files can be helpful in troubleshooting Oracle Virtual Directory issues. Log files for Oracle Virtual Directory are found under the following directory:

```
ORACLE_INSTANCE/diagnostics/log/OVD/<OVDComponentName>
```

The order in which log files should be examined when troubleshooting is:

1. diagnostic.log: This file captures diagnostic messages.
2. http-errors.log: This file captures HTTP errors.
3. access.log: This file captures information about processes and clients that access the Oracle Virtual Directory instance.

7.5 Oracle Directory Integration Platform High Availability

This section provides an introduction to Oracle Directory Integration Platform and describes how to design and deploy a high availability environment for Oracle Directory Integration Platform and Oracle Directory Services Manager.

See Section 7.6, "Oracle Directory Services Manager High Availability" for more information about Oracle Directory Services Manager.

7.5.1 Oracle Directory Integration Platform Component Architecture

Oracle Directory Integration Platform is a J2EE application that enables you to integrate your applications and directories, including third-party LDAP directories, with Oracle Internet Directory.

Oracle Directory Integration Platform includes services and interfaces that allow you to deploy synchronization solutions with other enterprise repositories. It can also be used to provide Oracle Internet Directory interoperability with third party metadirectory solutions.

Oracle Directory Integration Platform provides two distinct services depending on the type of integration needed:

- Synchronization through the Oracle Directory Integration Platform Synchronization Service, which keeps connected directories consistent with the central Oracle Internet Directory.
- Notification through Oracle Directory Integration Platform Provisioning Service, which sends notifications to target applications periodically to reflect changes made to a user's status or information.

Note: Throughout the rest of this chapter, these terms will be used:

- "Synchronization Service" for Oracle Directory Integration Platform Synchronization Service
- "Provisioning Service" for Oracle Directory Integration Platform Provisioning Service

---

Figure 7–6 shows the Oracle Directory Integration Platform architecture.

Figure 7–6 Oracle Directory Integration Platform Architecture

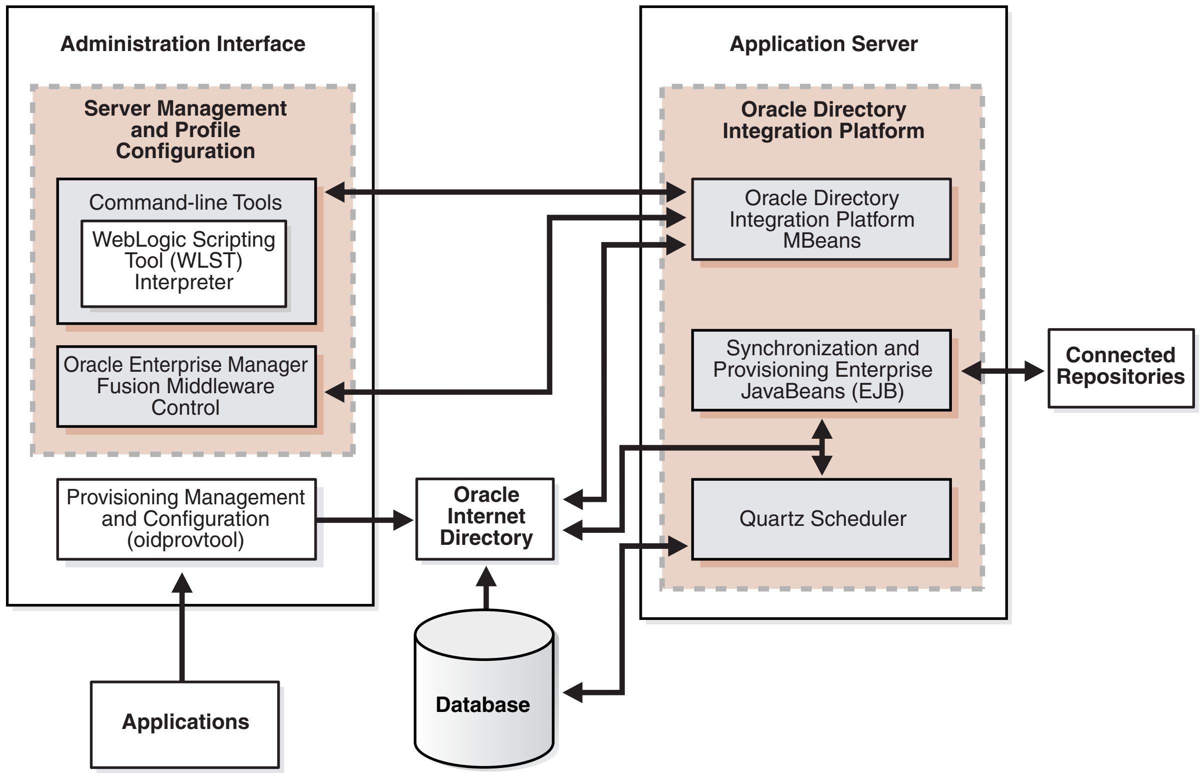

Administration Interface
Server Management and Profile Configuration
Command-line Tools
WebLogic Scripting Tool (WLST) Interpreter
Oracle Enterprise Manager Fusion Middleware Control
Provisioning Management and Configuration (oidprovtool)
Applications
Oracle Internet Directory
Database
Application Server
Oracle Directory Integration Platform
Oracle Directory Integration Platform MBeans
Synchronization and Provisioning Enterprise JavaBeans (EJB)
Quartz Scheduler
Connected Repositories

Figure 7–6 shows the various components of Oracle Directory Integration Platform. Oracle Internet Directory is synchronized with connected directories using Oracle Directory Integration Platform's Synchronization Enterprise JavaBeans (EJB) and the Quartz Scheduler.

Similarly, changes in Oracle Internet Directory are sent to various applications using Oracle Directory Integration Platform's Provisioning Enterprise JavaBeans (EJB) and the Quartz Scheduler.

7.5.1.1 Oracle Directory Integration Platform Component Characteristics

Oracle Directory Integration Platform is a J2EE application that enables you to integrate your applications and directories, including third-party LDAP directories, with Oracle Internet Directory.

Oracle Directory Integration Server provides synchronization services through the Synchronization Service and integration services through the Provisioning Service, as described below:

- The Synchronization Service provides:
  - Scheduling: Processing a synchronization profile based on a predefined schedule
  - Mapping: Executing rules for converting data between connected directories and Oracle Internet Directory
  - Data propagation: Exchanging data with connected directories by using a connector
  - Error handling
- The Provisioning Service provides:
  - Data propagation: Exchanging data with connected directories by using a connector
  - Event notification: Notifying an application of a relevant change to the user or group data stored in Oracle Internet Directory
  - Error handling

Oracle Directory Integration Platform is deployed on Oracle WebLogic Server. By default, Oracle Directory Integration Platform is installed as part of the Oracle Identity Management software stack; however, depending on your architectural requirements, it can also be installed as a standalone application.

7.5.1.1.1 Runtime Processes The Oracle Directory Integration Server provides:

- The Synchronization Service using the Synchronization Enterprise JavaBeans (EJB) and the Quartz Scheduler. The Synchronization EJB is stateless in nature.

  The Quartz scheduler invokes the Synchronization EJBs that synchronize the appropriate change to all the connected directories.

  The Synchronization Service supplies these changes using the interface and format required by the connected directory. Synchronization through the Oracle Directory Integration Platform connectors ensures that Oracle Internet Directory remains up-to-date with all the information that Oracle Internet Directory clients need.
- The Provisioning Service uses the Provisioning Enterprise JavaBeans (EJB) and the Quartz Scheduler. The Provisioning EJB is stateless in nature.

  The Quartz scheduler invokes the Provisioning EJBs that in turn notify each of the provisioned application of the changes.
- UpdateJob EJB periodically polls the Oracle Internet Directory changelog for changes in the Synchronization or Provisioning Profiles. When a profile change is detected, the Quartz scheduler jobs are updated accordingly.

7.5.1.1.2 Process Lifecycle Oracle Directory Integration Platform is deployed to an Oracle WebLogic Server as an externally managed application. By default, the Oracle Directory Integration Platform application leverages the underlying WebLogic Server infrastructure for startup, shutdown, monitoring and other lifecycle events. WebLogic Node Manager can be configured to monitor the server process and restart it in case of failure.

Oracle Directory Integration Platform is initialized when the Managed Server it is deployed on starts up. After the Oracle Directory Integration Platform application starts up, the initialization code creates an initial set of jobs for the existing directory integration profiles and then invokes the UpdateJobBean EJB.

UpdateJobBean updates the jobs submitted to the Quartz Scheduler.

Depending upon the job, the Quartz scheduler invokes either the Provisioning EJB or the Synchronization EJB. If the Provisioning EJB is invoked, it, in turn, notifies each of the provisioned application of the changes to the Oracle Internet Directory. If the Synchronization EJB is invoked, it, in turn, synchronizes the appropriate change to all the connected directories.

Starting and Stopping

The Oracle Directory Integration Platform lifecycle events can be managed using one or more of the command line tools and consoles listed below:

- Oracle WebLogic Scripting Tool (WLST)
- Oracle Enterprise Manager Fusion Middleware Control
- Oracle WebLogic Server Administration Console
- Oracle WebLogic Node Manager

7.5.1.1.3 Request Flow Oracle Directory Integration Platform does not service any external requests. Its main functionality is to support synchronization or provisioning based on the profiles created.

This section describes the information flow during synchronization and provisioning.

Oracle Directory Integration Platform Synchronization Service Flow

Oracle Directory Integration Platform relies on the directory synchronization profiles to synchronize changes between Oracle Internet Directory and the connected directories.

Depending on where the changes are made, synchronization can occur:

- From a connected directory to Oracle Internet Directory
- From Oracle Internet Directory to a connected directory
- In both directions

Synchronizing from Oracle Internet Directory to a Connected Directory

Oracle Internet Directory maintains a change log in which it stores incremental changes made to directory objects. It stores these changes sequentially based on the change log number.

Synchronization from Oracle Internet Directory to a connected directory makes use of this change log. Consequently, when running the Oracle Directory Integration Platform, you must start Oracle Internet Directory with the default setting in which change logging is enabled.

Each time the Oracle Directory Integration Platform Synchronization Service processes a synchronization profile, it:

1. Retrieves the latest change log number up to which all changes have been applied.
2. Checks each change log entry more recent than that number.
3. Selects changes to be synchronized with the connected directory by using the filtering rules in the profile.
4. Applies the mapping rules to the entry and makes the corresponding changes in the connected directory.

The appropriate entries or attributes are then updated in that connected directory. If the connected directory does not use DB, LDAP, tagged, or LDIF formats directly, then the agent identified in its profile is invoked. The number of the last change successfully used is then stored in the profile.

Periodically, Oracle Internet Directory purges the change log after all profiles have used what they need, and identifies where subsequent synchronization should begin.

Synchronizing from a Connected Directory to Oracle Internet Directory

When a connected directory uses DB, LDAP, tagged, or LDIF formats directly, changes to its entries or attributes can be automatically synchronized by the Oracle Directory Integration Platform Synchronization Service. Otherwise, the connector has an agent in its synchronization profile, which writes the changes to a file in the LDIF or tagged format. The Oracle Directory Integration Platform Synchronization Service then uses this file of connected directory data to update Oracle Internet Directory.

Provisioning Flow

Provisioning refers to the process of providing users, groups, and other objects with access to applications and other resources that are available within an environment. A provisioning-integrated application refers to an application that has registered for provisioning events and registered a provisioning-integration profile in Oracle Internet Directory.

An application can be provisioned with Oracle Directory Integration Platform Provisioning in one of the methods below:

- Synchronous provisioning
- Asynchronous provisioning
- Provisioning data flow

Each of these provisioning methods is described in its own section below.

Synchronous Provisioning

Applications that maintain user information in Oracle Internet Directory and use the Data Access Java plug-in to create, modify, and delete user entries whenever the change occurs in Oracle Internet Directory are said to be provisioned synchronously, since the Data Access Java plug-in can be invoked directly from Oracle Identity Management, including the Provisioning Console and command-line LDAP tools.

Synchronous provisioning with the Oracle Directory Integration Platform Provisioning Service can be invoked from Oracle Identity Management Tools (such as the Provisioning Console in the Oracle Fusion Middleware Control and bulkprov), synchronization with third-party directories, or command-line LDAP tools.

Synchronous provisioning with the Oracle Directory Integration Platform Provisioning Service from Oracle Identity Management Tools such as through the Provisioning

Console screen in the Oracle Fusion Middleware Control, bulk provisioning with the provProfileBulkProv command, and from third-party directories follows this process:

1. A new user entry is created in Oracle Internet Directory.
2. The Oracle Identity Management component that created the new user entry invokes the Data Access Java plug-in.
3. The Data Access Java plug-in provisions the new user account in the application.

Synchronous provisioning from command line LDAP tools follows this process:

1. A command-line LDAP tool creates a new user entry in Oracle Internet Directory.
2. At the next scheduled synchronization interval, the Oracle Directory Integration Platform identifies new user entries in Oracle Internet Directory that require provisioning.
3. The Oracle Directory Integration Platform invokes the Data Access Java plug-in.
4. The Data Access Java plug-in provisions the new user accounts in the application.

Asynchronous Provisioning

In asynchronous provisioning, the provisioning is handled by a PL/SQL plug-in and not by any component of Oracle Identity Management

The Oracle Directory Integration Platform propagates PL/SQL events to a provisioning-integrated application, which then executes a PL/SQL plug-in to process the events. Execution of a PL/SQL plug-in occurs within the application repository and not within the address space of any Oracle Identity Management component. Because provisioning is handled by a PL/SQL plug-in and not by any component of Oracle Identity Management, provisioning-integrated applications that implement a PL/SQL plug-in are provisioned asynchronously.

Asynchronous provisioning with the Oracle Directory Integration Platform Provisioning Service can be invoked from Oracle Identity Management Tools (such as Provisioning Console and bulkprov), synchronization with third-party directories, or command-line LDAP tools.

Asynchronous provisioning from Oracle Identity Management Tools such as the Provisioning Console, bulk provisioning with the provProfileBulkProv command, and third-party directories follows this process:

1. A new user entry and an associated entry containing application-specific user preferences are created in Oracle Internet Directory.
2. At the next scheduled synchronization interval, the Oracle Directory Integration Platform identifies new user entries in Oracle Internet Directory that require provisioning.
3. Provisioning events are sent from the Oracle Directory Integration Platform to the PL/SQL plug-in.

Asynchronous provisioning using command line LDAP tools follows this process:

1. A new user entry is created in Oracle Internet Directory using a command-line LDAP tool.
2. At the next scheduled synchronization interval, the Oracle Directory Integration Platform identifies new user entries in Oracle Internet Directory that require provisioning, and creates an associated entry containing application-specific user preferences.

3. Provisioning events are sent from the Oracle Directory Integration Platform to the PL/SQL plug-in.

Provisioning Data Flow

Regardless of whether it is provisioned synchronously or asynchronously, an application can invoke the Pre-Data Entry and Post-Data Entry plug-ins to enhance provisioning intelligence and implement business policies. Both plug-ins are invoked by Oracle Identity Management components such as the Oracle Internet Directory Provisioning Console and bulk provisioning with the provProfileBulkProv command.

The Pre-Data Entry plug-in populates fields according to provisioning policies. The primary purpose of this plug-in is to determine whether a user should be provisioned in an application. For example, if an organization has a a policy where only managers are provisioned for a financial application, the Pre-Data Entry plug-in can be used to identify which user entries to provision. Common user attributes are already populated when this plug-in is invoked, so it should have adequate information to make provisioning decisions.

The Post-Data Entry plug-in primarily validates data entered by users for common attributes and application-specific attributes. The validation for the plug-in must be successful for provisioning to continue.

The provisioning data flow follow this process:

1. Base user information is created.
2. The Pre-Data Entry plug-in is invoked, which populates fields according to policies.
3. The Post-Data Entry plug-in is invoked, which validates data entered by the user.
4. Depending on the provisioning approach, either asynchronous or synchronous provisioning procedures are invoked.

If provisioning is performed with the Provisioning Console, then after the Pre-Data Entry Plug-in is invoked, but before the Post-Data Entry plug-in is invoked, an administrator can modify the application attributes.

7.5.1.1.4 Configuration Artifacts Oracle Directory Integration Platform-related configuration details are maintained in the `dip-config.xml` file. This configuration file is packaged as part of the Oracle Directory Integration Platform application. The default location for the `dip-config.xml` file is:

```
MW_HOME/user_projects/domains/domainName/serverName/stage/DIP/11.1.1.1.0/
DIP/configuration
```

The parameters are managed using Config MBeans. Table 7–6 shows the configuration parameters required in `dip-config.xml` to start the Oracle Directory Integration Platform:

Table 7–6 Configuration Parameters Required to Start Directory Integration Platform

Parameter	Description
OID Host	Host name of the Oracle Internet Directory to which Oracle Directory Integration Platform needs to connect.
OID Port	The Oracle Internet Directory port

Table 7–6 (Cont.) Configuration Parameters Required to Start Directory Integration

Parameter	Description
SSL Mode	The SSL mode used to connect to Oracle Internet Directory. Valid values are: ■ 0: non-SSL ■ 1: SSL with no authentication (handshake only). This is the default mode ■ 2: SSL with server-only authentication enabled. Authentication is based on the Trust Point Certificate.
JKS Location	File system location to store the Java Keystore (JKS)
Quartz Threads	Maximum number of threads that can be used by Quartz for scheduling the processes.

Once the Oracle Directory Integration Platform server is up and running, it reads further details from Oracle Internet Directory for handling its synchronization and provisioning functions.

For information on creating synchronization profiles and provisioning profiles, see:

- "Creating Synchronization Profiles" in the *Oracle Fusion Middleware Integration Guide for Oracle Identity Management*.
- "Managing Provisioning Profiles Using oidprovtool" in the *Oracle Fusion Middleware Integration Guide for Oracle Identity Management*.

7.5.1.1.5 External Dependencies Oracle Directory Integration Platform uses an Oracle Internet Directory to store its metadata. The Quartz Scheduler uses the ODSSM schema to store its scheduling information in the database. The same database is used by Oracle Internet Directory and Oracle Directory Integration Platform. The ODSSM schema required for Oracle Directory Integration Platform is created as part of Oracle Internet Directory schema creation.

Oracle Directory Integration Platform is also dependent on the Oracle Credential Store Framework (CSF), a secure framework provided by Oracle and the Java Keystore (JKS) to store wallets and credentials used to connect to Oracle Internet Directory and third party LDAP stores over SSL.

Oracle Directory Integration Platform is also dependent on the Oracle Fusion Middleware Common Audit Framework, which is installed by default.

7.5.1.1.6 Oracle Directory Integration Platform Log File Oracle Directory Integration Platform is a J2EE application deployed on top of Oracle WebLogic Server. All log messages are logged in the server log file of the Oracle WebLogic Server that the application is deployed on. The default location of the server log is:

```
WEBLOGIC_SERVER_HOME/user_projects/domains/domainName/servers/serverName/logs/
serverName-diagnostic.log
```

7.5.2 Oracle Directory Integration Platform High Availability Concepts

This section provides conceptual information about using Oracle Directory Integration Platform in a high availability configuration.

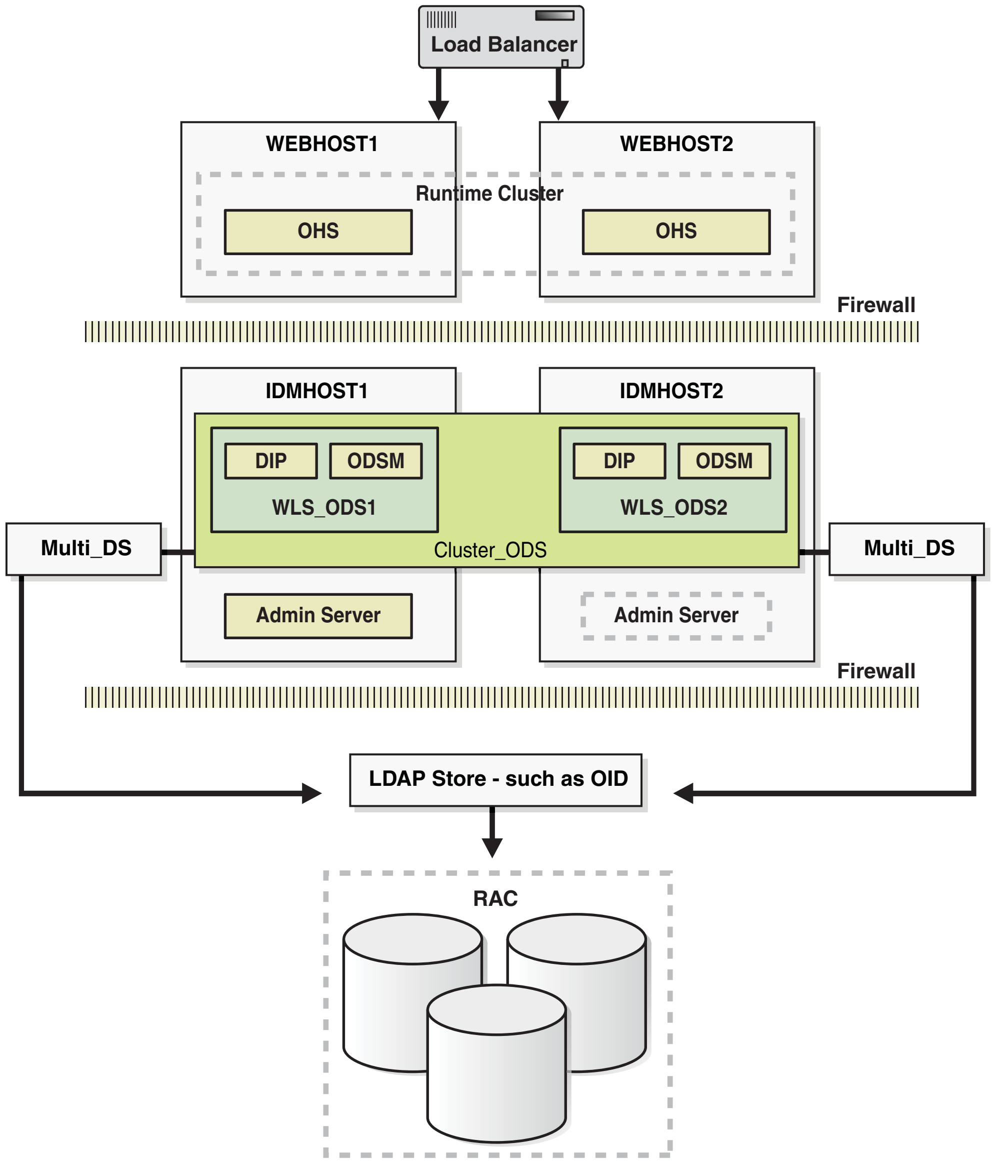
In the Oracle Directory Integration Platform high availability configuration described in this section, Oracle Directory Integration Platform and Oracle Directory Services

Manager are installed and configured on two hosts in a two-node high availability active-active configuration.

7.5.2.1 Oracle Directory Integration Platform High Availability Architecture

Figure 7–7 shows the Oracle Directory Integration Platform and Oracle Directory Services Manager high availability architecture in an active-active configuration.

Figure 7–7 Oracle Directory Integration Platform and Oracle Directory Services Manager in a High Availability Architecture



In Figure 7–7, the application tier includes the IDMHOST1 and IDMHOST2 computers.

On IDMHOST1, the following installations have been performed:

- An Oracle Directory Integration Platform instance and Oracle Directory Services Manager instance have been installed on the WLS_ODS1 Managed Server. The

RAC database has been configured in a JDBC multi data source to protect the instances from RAC node failure.

- A WebLogic Administration Server has been installed. Under normal operations, this is the active Administration Server.

On IDMHOST2, the following installations have been performed:

- An Oracle Directory Integration Platform instance and Oracle Directory Services Manager instance have been installed in the WLS_ODS2 Managed Server. The RAC database has been configured in a JDBC multi data source to protect the instances from RAC node failure.

  The instances in the WLS_ODS2 Managed Server on IDMHOST2 and the instances in the WLS_ODS1 Managed Server on IDMHOST1 are configured as the CLUSTER_ODS cluster.

- A WebLogic Administration Server has been installed. Under normal operations, this is the passive Administration Server. You will make this Administration Server active if the Administration Server on IDMHOST1 becomes unavailable.

7.5.2.1.1 Starting and Stopping the Cluster In a high availability architecture, Oracle Directory Integration Platform and Oracle Directory Services Manager are deployed on an Oracle WebLogic Cluster that has at least two servers as a part of the cluster.

By default, the WebLogic Server starts, stops and monitors the applications. By default, both the Oracle Directory Integration Platform and Oracle Directory Services Manager applications leverage the high availability features of the underlying WebLogic Clusters. In case of hardware or other failures, session state is available to other cluster nodes that can resume the work of the failed node.

In a high availability environment, WebLogic Node Manager is configured to monitor the WebLogic servers. In case of failure, Node Manager restarts the WebLogic Server. If Node Manager cannot restart the server, then the front-ending load balancing router detects failure of a WebLogic instance in the Cluster and routes traffic to surviving instances.

7.5.2.1.2 Cluster-Wide Configuration Changes When Oracle Internet Directory is deployed in an active-active high availability configuration, all the Oracle Internet Directory instances belonging to the cluster share the same database. Any changes made to Oracle Directory Integration Platform on one Oracle Internet Directory node would automatically be propagated to all the Oracle Internet Directory instances in the cluster.

The following subsections describe configuration changes made to the Oracle Directory Integration Platform application in an Oracle Internet Directory multimaster replication deployment. In a multimaster replication deployment, configuration changes need to be applied to all the nodes in the cluster manually, as described below.

Directory Integration Profiles

Changes made to directory integration profiles on one Oracle Internet Directory node are not automatically replicated to other Oracle Internet Directory nodes in a default multimaster Oracle Internet Directory replication environment. They need to be manually copied over from the primary node to the secondary nodes on a periodic basis. This allows a directory synchronization profile to execute on a secondary node in the event of a problem on the primary node.

One of the parameters used by Oracle Directory Integration Platform is `orcllastappliedchangenumber`. The value assigned to the `lastchangenumber` attribute in a directory synchronization profile depends on the directory server on which Oracle Directory Integration Platform is running. In an active-active Oracle Directory Integration Platform configuration, you must manually update the `lastchangenumber` attribute in all instances.

The next section details the steps to copy the synchronization profiles and the provisioning profiles from the primary Oracle Internet Directory to the secondary Oracle Internet Directory in a multimaster replication deployment.

Directory Synchronization Profiles

After copying an export profile to a target node the `lastchangenumber` attribute must be updated with the value from the target node. Follow the steps below to update the value:

1. Disable the synchronization profile.
2. Get the value of the `lastchangenumber` attribute on the target node using the `ldapsearch` command.
3. Use `ldapsearch` to get the LDIF dump of the profile entry.
4. Use `ldapadd` to add the profile to the other Oracle Internet Directory instance.
5. Use the `updatechgnum` operation of the manageSyncProfiles command to update the `lastchangenumber` attribute in the export profile you copied to the target node with the value you obtained in Step 2.
6. Enable the synchronization profile.

Directory Provisioning Profiles

In a default multimaster Oracle Internet Directory replication environment, the Oracle Directory Integration Platform is installed in the same location as the primary Oracle Internet Directory. The information and steps in this section are applicable only when multimaster replication is set up.

If the primary node fails, event propagation stops for all profiles located on the node. Although the events are queued and not lost while the primary node is stopped, the events will not be propagated to any applications that expect them. To ensure that events continue to be propagated even when the primary node is down for the Version 1.0 and 2.0 profiles, the directory provisioning profiles must be copied to other secondary nodes.

However, directory provisioning profiles should only be copied from the primary node to any secondary nodes immediately after an application is installed and before any user changes are made in Oracle Internet Directory.

To synchronize the directory provisioning profiles between a primary node and any secondary nodes, you need to do the following:

1. On the primary node, use the ldifwrite command to create an LDIF dump of the entries from this container:

   ```
   cn=provisioning profiles,cn=changelog subscriber,cn=oracle internet directory
   ```

2. Copy the LDIF dump to the secondary node.
3. Use the `ldapadd` command to add the profiles on the secondary node.

7.5.2.2 Protection from Failures and Expected Behavior

This section discusses protection from different types of failure in an Oracle Directory Integration Platform active-active cluster.

7.5.2.2.1 Process Failure

In a high availability environment, the Oracle Directory Integration Platform application is deployed on an Oracle WebLogic Server cluster comprised of at least two WebLogic instances.

By default, the Oracle Directory Integration Platform application leverages the high availability features of the underlying WebLogic Clusters. When Oracle Directory Integration Platform is deployed, the Quartz scheduler is started with a clustering option. When started with the clustering option, depending on the load on the node, the scheduler runs the job on any of the available nodes in the cluster. In case of hardware or other failures on one or more nodes, the scheduler runs the jobs on the available nodes.

In addition, in a high availability environment, WebLogic Node Manager is configured to monitor the WebLogic servers. In case of failure, Node Manager restarts the WebLogic Server.

Within the Oracle Directory Integration Platform application, the Quartz Scheduler invokes the Provisioning or Synchronization EJBs that do the actual work. As soon as the Quartz scheduler invokes an EJB, it tags that EJB as executing the job. In case the EJB fails, the Quartz scheduler marks the job as failed and reschedules it to be executed later by another EJB.

7.5.2.2.2 Expected Client Application Behavior When Failure Occurs

Oracle Directory Integration Platform failover is not transparent to end users.

Configuration changes made on a Oracle Directory Integration Platform instance deployed in a high availability topology are not propagated automatically to all the Oracle Directory Integration Platform instances in the topology.

It is recommended that you use the manageDIPServerConfig tool to make the same configuration changes on all Oracle Directory Integration Platform instances in the topology. This ensures that the configuration across all the Oracle Directory Integration Platform instances in the topology is uniform.

See *Oracle Fusion Middleware Integration Guide for Oracle Identity Management* for more information about the manageDIPServerConfig tool.

7.5.2.2.3 External Dependency Failure

Oracle Directory Integration Platform requires the back-end repository, Oracle Internet Directory, Credential Store Framework and the WebLogic Managed Server to be available during startup. Oracle Directory Integration Platform fails to start if any of these is not available.

7.5.2.3 Oracle Directory Integration Platform Prerequisites

This section describes prerequisites for setting up Oracle Directory Integration Platform in the high availability architecture.

Note: Oracle Directory Integration Platform uses Quartz to maintain its jobs and schedules in the database. For the Quartz jobs to be run on multiple Oracle Directory Integration Platform nodes in a cluster, it is required that the system clocks on the cluster nodes be synchronized.

Oracle Internet Directory should be installed and configured by following the instructions in these sections:

- Section 7.3.2.3, "Oracle Internet Directory Prerequisites"
- Section 7.3.3, "Oracle Internet Directory High Availability Configuration Steps"

7.5.3 Oracle Directory Integration Platform and Oracle Directory Services Manager High Availability Configuration Steps

In a high availability environment, it is recommended that Oracle WebLogic Server utilities be used for clustering, load balancing, and failover of Oracle Directory Integration Platform instances and Oracle Directory Services Manager.

This section describes how to perform the following installation and configuration on IDMHOST1 and IDMHOST2:

- Installing the WebLogic Administration Server on IDMHOST1 and IDMHOST2
- Installing and Configuring Oracle Directory Integration Platform and Oracle Directory Services Manager on IDMHOST1
- "Creating boot.properties for the Administration Server on IDMHOST1"
- Installing and Configuring Oracle Directory Integration Platform and Oracle Directory Services Manager on IDMHOST2
- Post-Installation Steps for Oracle Directory Integration Platform and Oracle Directory Services Manager
- Installing Oracle HTTP Server on WEBHOST1 and WEBHOST2

It is not required to install Oracle Directory Integration Platform and Oracle Directory Services Manager in the same installation session, on the same Managed Server, or on the same host. The installation and configuration steps in this section are one example of installing Oracle Directory Integration Platform and Oracle Directory Services Manager in a two-node active-active cluster.

7.5.3.1 Installing the WebLogic Administration Server on IDMHOST1 and IDMHOST2

On IDMHOST1 and IDMHOST2, start the Oracle WebLogic Server installation by running the installer executable file.

Start the Oracle WebLogic Server installer as follows:

- On UNIX (Linux in the following example):

  ```
  ./server103_linux32.bin
  ```

  On Windows:

  ```
  server103_win32.exe
  ```

Then follow these steps in the installer to install Oracle WebLogic Server on the computer:

1. On the Welcome screen, click **Next**.
2. On the Choose Middleware Home Directory screen, choose a directory on your computer into which the Oracle WebLogic software is to be installed.

   For the **Middleware Home Directory**, specify this value:

   ```
   /u01/app/oracle/product/fmw
   ```

   Click **Next**.
3. On the Register for Security Updates screen, enter your "My Oracle Support" User Name and Password.

4. On the Choose Install Type screen, the installation program displays a window in which you are prompted to indicate whether you wish to perform a complete or a custom installation.

   Choose **Typical** or **Custom** (**Typical** is chosen in this example)

   Click **Next**.

5. On the Choose Product Installation Directories screen, specify the following value for this field:

   - WebLogic Server:

     ```
     /u01/app/oracle/product/fmw/wlserver_10.3
     ```

   Click **Next**.

6. On the Installation Summary screen, the window contains a list of the components you selected for installation, along with the approximate amount of disk space to be used by the selected components once installation is complete.

   Click **Next**.

7. On the Installation Complete screen, deselect the **Run Quickstart** checkbox.

   Click **Done**.

7.5.3.2 Installing and Configuring Oracle Directory Integration Platform and Oracle Directory Services Manager on IDMHOST1

Follow these steps to install and configure Oracle Directory Integration Platform and Oracle Directory Services Manager on IDMHOST1:

1. Ensure that the system, patch, kernel and other requirements are met. These are listed in *Oracle Fusion Middleware Installation Guide for Oracle Identity Management* in the Oracle Fusion Middleware documentation library for the platform and version you are using.

2. Ensure that port 7006 is not in use by any service on OVDHOST1 by issuing these commands for the operating system you are using. If a port is not in use, no output is returned from the command.

   On UNIX:

   ```
   netstat -an | grep "7006"
   ```

   On Windows:

   ```
   netstat -an | findstr :7006
   ```

3. If the port is in use (if the command returns output identifying the port), you must free the port.

   On UNIX:

   Remove the entries for port 7006 in the `/etc/services` file and restart the services, or restart the computer.

   On Windows:

   Stop the component that is using the port.

4. Copy the `staticports.ini` file from the `Disk1/stage/Response` directory to a temporary directory.

5. Edit the `staticports.ini` file that you copied to the temporary directory to assign the following custom ports (uncomment the line where you specify the port numbers for Oracle Directory Services Manager):

   ```
   # The port for ODSM server
   ODS Server port = 7006
   ```

6. Start the Oracle Identity Management 11*g* Installer as follows:

   On UNIX, issue this command: **`runInstaller`**

   On Windows, double-click **`setup.exe`**

   The `runInstaller` and `setup.exe` files are in the `../install/`*`platform`* directory, where *platform* is a platform such as Linux or Win32.

   This displays the Specify Oracle Inventory screen.

7. On the Specify Inventory Directory screen, enter values for the Oracle Inventory Directory and the Operating System Group Name. For example:

   Specify the Inventory Directory: `/u01/app/oraInventory`

   Operating System Group Name: `oinstall`

   A dialog box appears with the following message:

   "Certain actions need to be performed with root privileges before the install can continue. Please execute the script /u01/app/oraInventory/createCentralInventory.sh now from another window and then press "Ok" to continue the install. If you do not have the root privileges and wish to continue the install select the "Continue installation with local inventory" option"

   Log in as root and run the "/u01/app/oraInventory/createCentralInventory.sh"

   This sets the required permissions for the Oracle Inventory Directory and then brings up the Welcome screen.

   Note: The Oracle Inventory screen is not shown if an Oracle product was previously installed on the host. If the Oracle Inventory screen is not displayed for this installation, make sure to check and see:

   1. If the `/etc/oraInst.loc` file exists
   2. If the file exists, the Inventory directory listed is valid
   3. The user performing the installation has write permissions for the Inventory directory

   ---

8. On the Welcome screen, click **Next**.

9. On the Select Installation Type screen, select **Install and Configure** and then click **Next**.

10. On the Prerequisite Checks screen, the installer completes the prerequisite check. If any fail, fix them and restart your installation.

    Click **Next**.

11. On the Select Domain screen, select **Create New Domain** and enter the domain details:

    - **User Name:** `weblogic`

- **User Password:** `******`
- **Confirm Password:** `******`
- **Domain Name:** `IDMDomain`

Click **Next**.

12. On the Specify Installation Location screen, specify the following values:
    - **Oracle Middleware Home Location:**

      `/u01/app/oracle/product/fmw`
    - **Oracle Home Location:**

      `ods`
    - **WebLogic Server Directory**:

      `/u01/app/oracle/product/fmw/wlserver_10.3`
    - **Oracle Instance Location:**

      `/u01/app/oracle/admin/ods_instance1`
    - **Oracle Instance Name**:

      `ods_instance1`

    Click **Next**.
13. On the Specify Oracle Configuration Manager Details screen, specify these values:
    - **Email Address**: Provide the email address for your My Oracle Support account.
    - **Oracle Support Password**: Provide the password for your My Oracle Support account.
    - Check the checkbox next to the **I wish to receive security updates via My Oracle Support** field.

    Click **Next**.
14. On the Configure Components screen, select **Oracle Directory Integration Platform, Management Components - Oracle Directory Services Manager**. Deselect all the other components. Ignore the warning that says Oracle Internet Directory needs to be configured as a prerequisite.

    Select the **Clustered** check box.

    Click **Next**.

    Note: The default Oracle WebLogic Server clustering mode set by the installer is unicast (not multicast).

    ---
15. On the Configure Ports screen, select **Specify Ports Using Configuration File** and enter the filename for the `staticports.ini` file that you copied to the temporary directory.

    Click **Next**.
16. On the Specify OID Details screen, specify the following:

- **Hostname:** `oid.mycompany.com`
- **Port**: 636
- **Username:** `cn=orcladmin`
- **Password:** `******`

Click **Next**.

17. On the Specify Schema Database screen, select **Use Existing Schema** and specify the following values:

- Connect String:

```
infradbhost1-vip.mycompany.com:1521:idmdb1^infradbhost2-vip.mycompany.com:1
521:idmdb2@idmedg.mycompany.com
```

Note: The RAC database connect string information needs to be provided in the format *host1*:*port1*:*instance1*^*host2*:*port2*:*instance2*@*servicename*.

During this installation, it is not required for all the RAC instances to be up. If one RAC instance is up, the installation can proceed.

It is required that the information provided above is complete and accurate. Specifically, the correct host, port, and instance name must be provided for each RAC instance, and the service name provided must be configured for all the specified RAC instances.

Any incorrect information entered in the RAC database connect string has to be corrected manually after the installation.

- User Name: `ODSSM`
- Password: `******`

Click **Next**.

18. On the Installation Summary screen, review the selections to ensure that they are correct (if they are not, click **Back** to modify selections on previous screens), and click **Install**.
19. On the Installation Progress screen on UNIX systems, a dialog box appears that prompts you to run the `oracleRoot.sh` script. Open a window and run the script, following the prompts in the window.

    Click **Next**.
20. On the Configuration screen, multiple configuration assistants are launched in succession; this process can be lengthy. When it completes, click **Next**.
21. On the Installation Complete screen, click **Finish** to confirm your choice to exit.

7.5.3.3 Creating boot.properties for the Administration Server on IDMHOST1

This section describes how to create a `boot.properties` file for the Administration Server on IDMHOST1. The `boot.properties` file enables the Administration Server to start without prompting for the `administrator` username and password.

Follow these steps to create the `boot.properties` file:

1. On IDMHOST1, go the following directory:

```
MW_HOME/user_projects/domains/domainName/servers/AdminServer/security
```

For example:

```
cd /u01/app/oracle/product/fmw/user_
projects/domains/IDMDomain/servers/AdminServer/security
```

2. Use a text editor to create a file called `boot.properties` under the `security` directory. Enter the following lines in the file:

```
username=adminUser
password=adminUserPassword
```

Note: When you start the Administration Server, the username and password entries in the file get encrypted.

For security reasons, minimize the time the entries in the file are left unencrypted. After you edit the file, you should start the server as soon as possible so that the entries get encrypted.

3. Stop the Administration Server if it is running.

   See the "Starting and Stopping Oracle Fusion Middleware" chapter of the *Oracle Fusion Middleware Administrator's Guide* for information on starting and stopping WebLogic Servers.

4. Start the Administration Server on IDMHOST1 using the `startWebLogic.sh` script located under the `*MW_HOME*/user_projects/domains/*domainName*/bin` directory.

5. Validate that the changes were successful by opening a web browser and accessing the following pages:

   - WebLogic Server Administration Console at:

     ```
     http://oidhost1.mycompany.com:7001/console
     ```

   - Oracle Enterprise Manager Fusion Middleware Control at:

     ```
     http://oidhost1.mycompany.com:7001/em
     ```

   Log into these consoles using the `weblogic` user credentials.

7.5.3.4 Installing and Configuring Oracle Directory Integration Platform and Oracle Directory Services Manager on IDMHOST2

Follow these steps to install only the Oracle Identity Management software on IDMHOST2:

1. Ensure that the system, patch, kernel and other requirements are met. These are listed in *Oracle Fusion Middleware Installation Guide for Oracle Identity Management* in the Oracle Fusion Middleware documentation library for the platform and version you are using.

2. Start the Oracle Identity Management 11*g* Installer as follows:

   On UNIX, issue this command: **`runInstaller`**

   On Windows, double-click **`setup.exe`**

The `runInstaller` and `setup.exe` files are in the `../install/`*`platform`* directory, where *platform* is a platform such as Linux or Win32.

This displays the Specify Oracle Inventory screen.

3. On the Specify Inventory Directory screen, enter values for the Oracle Inventory Directory and the Operating System Group Name. For example:

   Specify the Inventory Directory: `/u01/app/oraInventory`

   Operating System Group Name: `oinstall`

   A dialog box appears with the following message:

   "Certain actions need to be performed with root privileges before the install can continue. Please execute the script /u01/app/oraInventory/createCentralInventory.sh now from another window and then press "Ok" to continue the install. If you do not have the root privileges and wish to continue the install select the "Continue installation with local inventory" option"

   Log in as root and run the "/u01/app/oraInventory/createCentralInventory.sh"

   This sets the required permissions for the Oracle Inventory Directory and then brings up the Welcome screen.

   Note: The Oracle Inventory screen is not shown if an Oracle product was previously installed on the host. If the Oracle Inventory screen is not displayed for this installation, make sure to check and see:

   1. If the `/etc/oraInst.loc` file exists
   2. If the file exists, the Inventory directory listed is valid
   3. The user performing the installation has write permissions for the Inventory directory

4. On the Welcome screen, click **Next**.
5. On the Select Installation Type screen, select **Install & Configure** and then click **Next**.
6. On the Prerequisite Checks screen, the installer completes the prerequisite check. If any fail, please fix them and restart your installation.

   Click **Next**.
7. On the Select Domain screen, select the **Expand Cluster** option and specify the values shown in the example below:

   - **HostName**: `idmhost1.mycompany.com`
   - **Port**: `7001`
   - **UserName**: `weblogic`
   - **User Password**: `<password for the weblogic user>`

   Click **Next**.
8. On the Specify Installation Location screen, specify the following values:

   - **Oracle Middleware Home Location:**

     ```
     /u01/app/oracle/product/fmw
     ```

- **Oracle Home Directory:**

  `ods`

- **WebLogic Server Directory**:

  `/u01/app/oracle/product/fmw/wlserver_10.3`

- **Oracle Instance Location**:

  `/u01/app/oracle/admin/ods_instance2`

- **Oracle Instance Name**:

  `ods_instance2`

  Click **Next**.

9. On the Configure Components screen, de-select all products except **Oracle DIP and Management Components**.

   Click **Next**.

10. On the Installation Summary screen, review the choices you made. If you need to make any changes, click **Back**. If you made the correct selections, click **Install**.
11. On the Installation Progress screen, view the progress of the installation.
12. On UNIX systems, once the installation is done, the `oracleRoot.sh` confirmation dialog box appears. This dialog box advises you that a configuration script needs to be run as root before the installation can proceed.

    Leaving the confirmation dialog b ox open, open another shell window, log in as root, and run this script file:

    ```
    /u01/app/oracle/product/fmw/ods/oracleRoot.sh
    ```

13. On the Configuration Progress screen, view the progress of the configuration.
14. On the Installation Complete screen, click **Finish** to confirm your choice to exit.

Note: After this installation completes, if you use the WebLogic Server Administration Console to view the state of the `wls_ods2` Managed Server, its state appears as UNKNOWN.

You must perform the steps in Section 7.5.3.5, "Post-Installation Steps for Oracle Directory Integration Platform and Oracle Directory Services Manager" to complete the installation of the Oracle Directory Integration Platform and Oracle Directory Services Manager instances on IDMHOST1 and IDMHOST2.

7.5.3.5 Post-Installation Steps for Oracle Directory Integration Platform and Oracle Directory Services Manager

In the previous section, the installer created a second Managed Server, wls_ods2 on IDMHOST2. However, the Oracle Directory Integration Platform application is not deployed on IDMHOST2 and the newly created Managed Server is not automatically started. Also, the WebLogic Administration Console shows the state of the wls_ods2 Managed Server on IDMHOST2 as UNKNOWN.

Follow the post-installation steps in this section to complete the installation and configuration of the Oracle Directory Integration Platform and Oracle Directory Services Manager applications on IDMHOST2.

7.5.3.5.1 Copy the DIP Application from IDMHOST1 to IDMHOST2 Follow the steps below to copy the Oracle Directory Integration Platform application from IDMHOST1 to IDMHOST2:

1. On IDMHOST2, create the following directory structure:

   ```
   MW_HOME/user_projects/domains/IDMDomain/servers/wls_ods2/stage/DIP/11.1.1.1.0
   ```

   For example:

   ```
   mkdir -p MW_HOME/user_projects/domains/IDMDomain/servers/wls_
   ods2/stage/DIP/11.1.1.1.0/
   ```

2. Copy the DIP directory from IDMHOST1 to IDMHOST2:

   Copy the following DIP directory on IDMHOST1:

   ```
   MW_HOME/user_projects/domains/IDMDomain/servers/wls_
   ods1/stage/DIP/11.1.1.1.0/DIP
   ```

   to the following location on IDMHOST2:

   ```
   MW_HOME/user_projects/domains/IDMDomain/servers/wls_ods2/stage/DIP/11.1.1.1.0/
   ```

   For example, from IDMHOST1, execute this command:

   ```
   scp -rp MW_HOME/user_projects/domains/IDMDomain/servers/wls_ods1/stage/
   DIP/11.1.1.1.0/DIP user@IDMHOST2://MW_HOME/user_projects/domains/IDMDomain/
   servers/wls_ods2/stage/DIP/11.1.1.1.0
   ```

7.5.3.5.2 Start the Managed Server on IDMHOST2 in a Cluster Follow these steps to start the newly-created `wls_ods2` Managed Server in a cluster on IDMHOST2:

1. In a web browser, use this URL to access the Oracle WebLogic Server Administration Console:

   ```
   http://idmhost1.mycompany.com:7001/console
   ```

   Log into the console using the administrator's credentials.

2. In the left pane of the Oracle WebLogic Server Administration Console, expand **Environment** and select **Clusters**.

   See the "Starting and Stopping Oracle Fusion Middleware" chapter of the *Oracle Fusion Middleware Administrator's Guide* for information on starting and stopping WebLogic Servers.

3. Click on the link for the cluster (`cluster_ods`) containing the Managed Server (`wls_ods2`) you want to stop.

4. Select **Control**.

5. Under **Managed Server Instances in this Cluster**, select the check box next to the Managed Server (`wls_ods2`) you want to start and click **Start**.

6. On the Cluster Life Cycle Assistant page, click **Yes** to confirm.

7. Node Manager starts the server on the target machine. When the Node Manager finishes its start sequence, the server's state is indicated in the **State** column in the Servers status table. The state should be RUNNING.

7.5.3.6 Installing Oracle HTTP Server on WEBHOST1 and WEBHOST2

This section describes how to install Oracle HTTP Server on WEBHOST1 and WEBHOST2. Oracle HTTP Server is not required if Oracle Directory Integration Platform is the only component being installed.

1. Ensure that the system, patch, kernel and other requirements are met. These are listed in the *Oracle Fusion Middleware Installation Guide for Web Tier* in the Oracle Fusion Middleware documentation library for the platform and version you are using.
2. Ensure that port 7777 is not in use by any service on WEBHOST1 or WEBHOST2 by issuing these commands for the operating system you are using. If a port is not in use, no output is returned from the command.

   On UNIX:

   ```
   netstat -an | grep "7777"
   ```

   On Windows:

   ```
   netstat -an | findstr :7777
   ```

3. If the port is in use (if the command returns output identifying the port), you must free the port.

   On UNIX:

   Remove the entries for port 7777 in the `/etc/services` file and restart the services, or restart the computer.

   On Windows:

   Stop the component that is using the port.
4. Copy the `staticports.ini` file from the `Disk1/stage/Response` directory to a temporary directory.
5. Edit the `staticports.ini` file that you copied to the temporary directory to assign the following custom ports (uncomment the line where you specify the port number for Oracle HTTP Server):

   ```
   # The port for Oracle HTTP server
   Oracle HTTP Server port = 7777
   ```

6. Start the Oracle Universal Installer for Oracle Fusion Middleware 11*g* Web Tier Utilities CD installation as follows:

   On UNIX, issue this command: **`runInstaller`**.

   On Windows, double-click **`setup.exe`**.

   The `runInstaller` and `setup.exe` files are in the `../install/`*`platform`* directory, where *platform* is a platform such as Linux or Win32.

   This displays the Specify Oracle Inventory screen.
7. On the Specify Inventory Directory screen, enter values for the Oracle Inventory Directory and the Operating System Group Name. For example:

   Specify the Inventory Directory: `/u01/app/oraInventory`

   Operating System Group Name: `oinstall`

   A dialog box appears with the following message:

"Certain actions need to be performed with root privileges before the install can continue. Please execute the script /u01/app/oraInventory/createCentralInventory.sh now from another window and then press "Ok" to continue the install. If you do not have the root privileges and wish to continue the install select the "Continue installation with local inventory" option"

Log in as root and run the "/u01/app/oraInventory/createCentralInventory.sh"

This sets the required permissions for the Oracle Inventory Directory and then brings up the Welcome screen.

Note: The Oracle Inventory screen is not shown if an Oracle product was previously installed on the host. If the Oracle Inventory screen is not displayed for this installation, make sure to check and see:

1. If the `/etc/oraInst.loc` file exists
2. If the file exists, the Inventory directory listed is valid
3. The user performing the installation has write permissions for the Inventory directory

---

8. On the Welcome screen, click **Next**.
9. On the Select Installation Type screen, select **Install and Configure**, and click **Next**.
10. On the Prerequisite Checks screen, the installer completes the prerequisite check. If any fail, fix them and restart your installation.

    Click **Next**.
11. On the Specify Installation Location screen:
    - On WEBHOST1 and WEBHOST2, set the **Location** to:

      ```
      /u01/app/oracle/admin
      ```

    Click **Next**.
12. On the Configure Components screen:
    - Select **Oracle HTTP Server**.
    - Select **Associate Selected Components with Weblogic Domain**.

    Click **Next**.
13. On the Specify WebLogic Domain screen:

    Enter the location where you installed Oracle WebLogic Server. Note that the Administration Server must be running.
    - **Domain Host Name**: `IDMHOST1`
    - **Domain Port No**: `7001`
    - **User Name**: `weblogic`
    - **Password**: `******`

    Click **Next**.
14. On the Specify Component Details screen:

- Enter the following values for WEBHOST1:
  - **Instance Home Location**: `/u01/app/oracle/admin/ohs_inst1`
  - **Instance Name**: `ohs_inst1`
  - **OHS Component Name**: `ohs1`
- Enter the following values for WEBHOST2:
  - **Instance Home Location**: `/u01/app/oracle/admin/ohs_inst2`
  - **Instance Name**: `ohs_inst2`
  - **OHS Component Name**: `ohs2`

Click **Next**.

15. On the Specify Webtier Port Details screen:
    - Select **Specify Custom Ports**. If you specify a custom port, select **Specify Ports using Configuration File** and then use the Browse function to select the file.
    - Enter the Oracle HTTP Server port, for example, 7777.

    Click **Next**.
16. On the Oracle Configuration Manager screen, enter the following:
    - **Email Address**: Provide the email address for your My Oracle Support account
    - **Oracle Support Password**: Provide the password for your My Oracle Support account.
    - **I wish to receive security updates via My Oracle Support**: Click this checkbox.
17. On the Installation Summary screen, ensure that the selections are correct. If they are not, click **Back** and make the necessary fixes. After ensuring that the selections are correct, click **Next**.
18. On the Installation Progress screen on UNIX systems, a dialog appears that prompts you to run the `oracleRoot.sh` script. Open a window and run the script, following the prompts in the window.

    Click **Next**.
19. On the Configuration Progress screen, multiple configuration assistants are launched in succession; this process can be lengthy.
20. On the Configuration Completed screen, click **Finish** to exit.

7.5.3.6.1 Configuring Oracle HTTP Server for Oracle Directory Services Manager High Availability Follow the instructions in this section to configure Oracle HTTP Server for Oracle Directory Services Manager high availability.

1. Configure Oracle HTTP Server to use the load balancing router virtual hosts.

   The Oracle HTTP Server instances on WEBHOST1 and WEBHOST2 should be configured to use the virtual hosts set up in the load balancer. Refer to Section 7.2.4.5, "Configuring Virtual Server Names and Ports for the Load Balancer" for more information about the virtual hosts.

   To configure the Oracle HTTP Server instances to use the load balancing router virtual hosts, edit the `httpd.conf` file to define the virtual host directives as follows:

```
NameVirtualHost *:7777
<VirtualHost *:7777>
   ServerName admin.mycompany.com:7777
   ServerAdmin you@your.address
   RewriteEngine On
   RewriteOptions inherit
</VirtualHost>
```

The `httpd.conf` file is located under the *`ORACLE_INSTANCE`*`/config/OHS/`*`componentName`* directory on both WEBHOST1 and WEBHOST2.

2. Configure Oracle HTTP Server to route to Oracle Directory Services Manager, Oracle Enterprise Manager Fusion Middleware Control, and the Oracle WebLogic Server Administration Console.

   To enable the Oracle HTTP Server instances to route to the Oracle Directory Services Manager applications on IDMHOST1 and IDMHOST2, add the directives below to the `mod_wl_ohs.conf` file located under the *`ORACLE_INSTANCE`*`/config/OHS/`*`componentName`* directory on WEBHOST1 and WEBHOST2:

   ```
   LoadModule weblogic_module "${ORACLE_HOME}/ohs/modules/mod_wl_ohs.so"

   <IfModule mod_weblogic.c>
   WebLogicHost IDMHOST1.MYCOMPANY.COM
   WebLogicPort PORT
   </IfModule>

   <Location /odsm>
   SetHandler weblogic-handler
   WebLogicCluster IDMHOST1.MYCOMPANY.COM:<PORT>,IDMHOST2.MYCOMPANY.COM:<PORT>
   </Location>
   ```

3. Stop and start Oracle HTTP Server using the `opmnctl` command:

   ```
   ORACLE_INSTANCE/bin/opmnctl stopall
   ORACLE_INSTANCE/bin/opmnctl startall
   ```

4. Validate that you can access the consoles using the load balancing router virtual host:

   Oracle Directory Services Manager Console:

   ```
   http://admin.mycompany.com:7777/odsm
   ```

   Oracle WebLogic Server Administration Console:

   ```
   http://idmhost1.mycompany.com:7001/console
   ```

   Oracle Enterprise Manager Fusion Middleware Control:

   ```
   http://idmhost1.mycompany.com:7001/em
   ```

Note: When Oracle Directory Integration Platform deployed in a high availability configuration is enabled for SSL with server-only Authentication (that is, SSL Mode2), it is required to use a common key store that contains the certificates of all the Oracle Internet Directory instances. You can create a common keystore by importing the certificates from all the Oracle Internet Directory instances and then copying the keystore to all nodes where Oracle Directory Integration Platform is running.

For more information about using SSL with Oracle Directory Integration Platform, refer to the *Oracle Fusion Middleware Integration Guide for Oracle Identity Management*.

7.5.4 Oracle Directory Integration Platform Failover and Expected Behavior

In a high availability environment, the Oracle Directory Integration Platform application is deployed on an Oracle WebLogic Server cluster comprised of at least two WebLogic instances.

By default, the Oracle Directory Integration Platform application leverages the high availability features of the underlying WebLogic Clusters. In case of hardware or other failures, session state is available to other cluster nodes that can resume the work of the failed node.

In addition, in a high availability environment, WebLogic Node Manager is configured to monitor the WebLogic servers. In case of failure, Node Manager restarts the WebLogic Server. In case of a database instance failure, the surviving RAC node takes over any remaining processes. There may be innocuous errors in the Managed Servers logs during a RAC failover, as discussed in Section 7.5.5, "Troubleshooting Oracle Directory Integration Platform High Availability."

7.5.5 Troubleshooting Oracle Directory Integration Platform High Availability

This section describes how to deal with issues involving Oracle Directory Integration Platform high availability.

7.5.5.1 Managed Server Log File Exceptions Received for Oracle Directory Integration Platform During a RAC Failover

During a RAC failover, exceptions similar to the ones below are seen in the Managed Server log files running the Oracle Directory Integration Platform application. These errors are thrown when the multi data sources configured on the WebLogic Server platform try to verify the health of the RAC database instances during failover. These are innocuous errors and can be ignored. The Oracle Directory Integration Platform application will recover and begin to operate normally after a lag of one or two minutes. During a RAC failover, there will be no Oracle Directory Integration Platform down time if one RAC instance is running at all times.

```
RuntimeException:
[2008-11-21T00:11:10.915-08:00] [wls_ods] [ERROR] []
[org.quartz.impl.jdbcjobstore.JobStoreTX] [tid: 25] [userId: <anonymous>]
[ecid: 0000Hqy69UiFW7V6u3FCEH199aj0000009,0] [APP: DIP] ClusterManager: Error
managing cluster: Failed to obtain DB connection from data source
'schedulerDS': java.sql.SQLException: Could not retrieve datasource via JNDI
url 'jdbc/schedulerDS' java.sql.SQLException: Cannot obtain connection:
driverURL = jdbc:weblogic:pool:schedulerDS, props =
{EmulateTwoPhaseCommit=false, connectionPoolID=schedulerDS,
jdbcTxDataSource=true, LoggingLastResource=false,
dataSourceName=schedulerDS}.[[
Nested Exception: java.lang.RuntimeException: Failed to setAutoCommit to true
for pool connection

AuthenticationException while connecting to OID:
[2008-11-21T00:12:08.812-08:00] [wls_ods] [ERROR] [DIP-10581] [oracle.dip]
[tid: 11] [userId: <anonymous>] [ecid: 0000Hqy6m54FW7V6u3FCEH199apO000000,0]
[APP: DIP] DIP was not able to get the context with the given details {0}[[
javax.naming.AuthenticationException: [LDAP: error code 49 - Invalid
Credentials]
```

Most of the exceptions will be related to the scheduler or LDAP, for example:

1. Could not retrieve datasource via JNDI url 'jdbc/schedulerDS' java.sql.SQLException.

2. javax.naming.AuthenticationException: [LDAP: error code 49 - Invalid Credentials]

7.5.5.2 Dealing with Error Messages Received After Starting WebLogic Node Manager

If you receive the following error message after starting WebLogic Node Manager, follow the steps that appear below after the error message:

```
<Dec 15, 2008 8:40:05 PM> <Warning> <Uncaught exception in server handler:
javax.net.ssl.SSLKeyException: [Security:090482]BAD_CERTIFICATE alert was
received from stbee21.us.oracle.com - 152.68.64.2155. Check the peer to
determine why it rejected the certificate chain (trusted CA configuration,
hostname verification). SSL debug tracing may be required to determine the
exact reason the certificate was rejected.> javax.net.ssl.SSLKeyException:
[Security:090482]BAD_CERTIFICATE alert was received from stbee21.us.oracle.com -
152.68.64.215. Check the peer to determine why it rejected the certificate chain
(trusted CA configuration, hostname verification). SSL debug tracing may be
required to determine the exact reason the certificate was rejected.
```

1. If you have not already done so, in the Change Center of the Administration Console, click **Lock & Edit**.
2. In the left pane of the Console, expand **Servers** and AdminServer (admin).
3. Select the **Configuration > SSL > Advanced Link**.
4. Select **None** for **Hostname Verification**.
5. Click **Save** to save the setting.
6. To activate these changes, in the Change Center of the Administration Console, click **Activate Changes**.
7. Restart all servers.

If the Managed Server is started in Admin mode, perform these steps:

1. If you have not already done so, in the Change Center of the Administration Console, click **Lock & Edit**.
2. In the left pane of the Console, expand **Servers** and the name of the server that is running in ADMIN mode.
3. Select the **Control > Start/Stop** tab.
4. Select the name of the server.
5. Click **Resume**.

6. Select **Yes** to resume servers.

7.5.5.3 If the Managed Server Doesn't Start After Copying the DIP Directory from IDMHOST1 to IDMHOST2

If the Managed Server on IDMHOST2 doesn't start after you copy the DIP directory from IDMHOST1 to IDMHOST2, follow these steps instead:

1. Copy the *MW_HOME*/`user_projects` directory on IDMHOST1 to the *MW_HOME*/`user_projects` directory on IDMHOST2.
2. On IDMHOST2, cd to the following directory, and note that there is no `wls_ods` folder:

   ```
   MW_HOME/user_projects/domains/IDMDomain/servers
   ```

3. Use the following command to make sure that no WebLogic Node Manager is running:

   ```
   ps -ef | grep java
   ```

4. Start WebLogic Node Manager using this command:

   On UNIX:

   ```
   ./startNodeManager.sh
   ```

   On Windows:

   ```
   startNodeManager.cmd
   ```

5. In the WebLogic Server Administration Console, start the `wls_ods2` Managed Server, which was created earlier.
6. On IDMHOST2, cd to the following directory and note that there is a `wls_ods2` folder created:

   ```
   MW_HOME/user_projects/domains/IDMDomain/servers
   ```

7. In the Administration Console, stop the `wls_ods2` Managed Server.
8. Issue the following command and then kill the NodeManager process:

   ```
   ps -ef | grep java
   ```

9. Copy the DIP folder from `wls_ods1` to `wls_ods2` as described above.
10. Start WebLogic Node Manager again.
11. Start `wls_ods2`.

7.5.5.4 If WebLogic Node Manager Fails to Start

If WebLogic Node Manager fails to start, make sure that you have copied the following domains file from IDMHOST1 to IDMHOST2:

```
MW_HOME/wlserver_10.3/common/nodemanager/nodemanager.domains
```

7.5.5.5 Configuration Changes Do Not Automatically Propagate to All Oracle Directory Integration Platform Instances in a Highly Available Topology

When you change the configuration of an Oracle Directory Integration Platform instance in a high availability topology, the configuration change does not propagate automatically to all the Oracle Directory Integration Platform instances in the topology.

Instead, use the manageDIPServerConfig tool to make the same configuration change for each Oracle Directory Integration Platform instance in the topology, which will ensure the same configuration across all the Oracle Directory Integration Platform instances in the topology. See *Oracle Fusion Middleware Integration Guide for Oracle Identity Management* for more information about the manageDIPServerConfig tool.

7.5.5.6 Operation Cannot Be Completed for Unknown Errors Message

Sometimes the following error message appears intermittently when you use the manageSyncProfiles command:

```
OPERATION CANNOT BE COMPLETED FOR UNKNOWN ERRORS
```

When you see this error message, start and stop the Managed Server (`wls_ods1` or `wls_ods2`). If the problem persists, repeat the copy method on the second node.

7.6 Oracle Directory Services Manager High Availability

This section provides an introduction to Oracle Directory Services Manager and describes how to design and deploy a high availability environment for Oracle Directory Integration Platform and Oracle Directory Services Manager.

See Section 7.5, "Oracle Directory Integration Platform High Availability" for more information about Oracle Directory Integration Platform.

7.6.1 Oracle Directory Services Manager Component Architecture

Oracle Directory Services Manager is a unified graphical user interface (GUI) for managing instances of Oracle Internet Directory and Oracle Virtual Directory. It is a replacement for Oracle Directory Manager, which is now deprecated. Oracle Directory Services Manager enables you to configure the structure of the directory, define objects in the directory, add and configure users, groups, and other entries.

Oracle Directory Services Manager is the interface used by end users to manage their directory entries, schemas, security, and other features.

Figure 7–8 shows the Oracle Directory Services Manager architecture.

Figure 7–8 Oracle Directory Services Manager Architecture

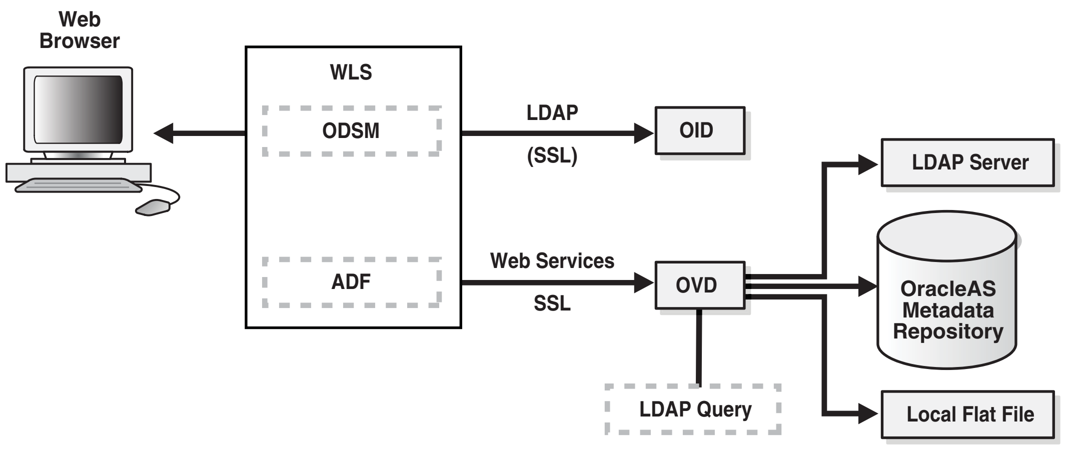


Figure 7–8 shows Oracle Directory Services Manager deployed in non-high availability architecture. Oracle Directory Services Manager is deployed on the Oracle WebLogic Server. Oracle Directory Services Manager is configured to communicate with the Oracle Virtual Directory and the Oracle Internet Directory instances it manages.

Oracle Directory Services Manager uses the HTTP(s) protocol to communicate with client browsers. It uses the LDAP(s) protocol to communicate with Oracle Internet Directory and over WebServices to communicate with Oracle Virtual Directory.

7.6.1.1 Oracle Directory Services Manager Component Characteristics

Oracle Directory Services Manager is an Oracle Application Development Framework (ADF)-based J2EE application that is deployed on top of the Oracle WebLogic Server. By default, Oracle Directory Services Manager is deployed to the Administration Server within the WebLogic domain, but depending on the requirements in your environment, it can also be deployed to a Managed Server.

Oracle Directory Services Manager can be deployed on the same node with Oracle Internet Directory or on a separate node.

Oracle Directory Services Manager can be invoked directly or from Oracle Enterprise Manager Fusion Middleware Control. Supported browsers include Firefox 2, Firefox 3, and Internet Explorer 7.

To invoke Oracle Directory Services Manager directly, enter the following URL into your browser's address field:

```
http://host:port/odsm/faces/odsm.jspx
```

where:

- *host* is the name of the Managed Server where Oracle Directory Services Manager is running.
- *port* is the Managed Server port number.

To invoke Oracle Directory Services Manager from Oracle Enterprise Manager Fusion Middleware Control:

- Select **Directory Services Manager** from the **Oracle Internet Directory** menu in the Oracle Internet Directory target, then **Data Browser**, **Schema**, **Security**, or **Advanced**.
- From the **Oracle Virtual Directory** menu in the Oracle Virtual Directory target, select **Directory Services Manager**, then **Data Browser**, **Schema**, **Adapter**, **Extensions**, or **Quick configuration wizard**.

A new browser window containing the Oracle Directory Services Manager Welcome screen will pop up.

Oracle Directory Services Manager is deployed to an Oracle WebLogic Server as an externally staged application. The WebLogic server manages the startup, shutdown, monitoring of the Oracle Directory Services Manager application. Oracle Directory Services Manager is initialized when the Managed Server starts up. Oracle Node Manager is configured to monitor the server process and restarts it in case of failure.

7.6.1.1.1 Lifecycle Management The lifecycle events for the Oracle Directory Services Manager application can be managed using one or more of the command line tools and consoles listed below:

The following tools can be used to start and stop Oracle Directory Services Manager processes:

- Oracle WebLogic Server Scripting Tool (WLST)
- Oracle Enterprise Manager Fusion Middleware Control
- WebLogic Server Administration Console
- WebLogic Node Manager

7.6.1.1.2 Oracle Directory Services Manager Log File Oracle Directory Services Manager messages are logged in the server log file of the Oracle WebLogic Server where it is running. The default location of the server log is:

```
WEBLOGIC_SERVER_HOME/user_projects/domains/domainName/servers/serverName/logs/
serverName-diagnostic.log
```

7.6.2 Oracle Directory Services Manager High Availability Concepts

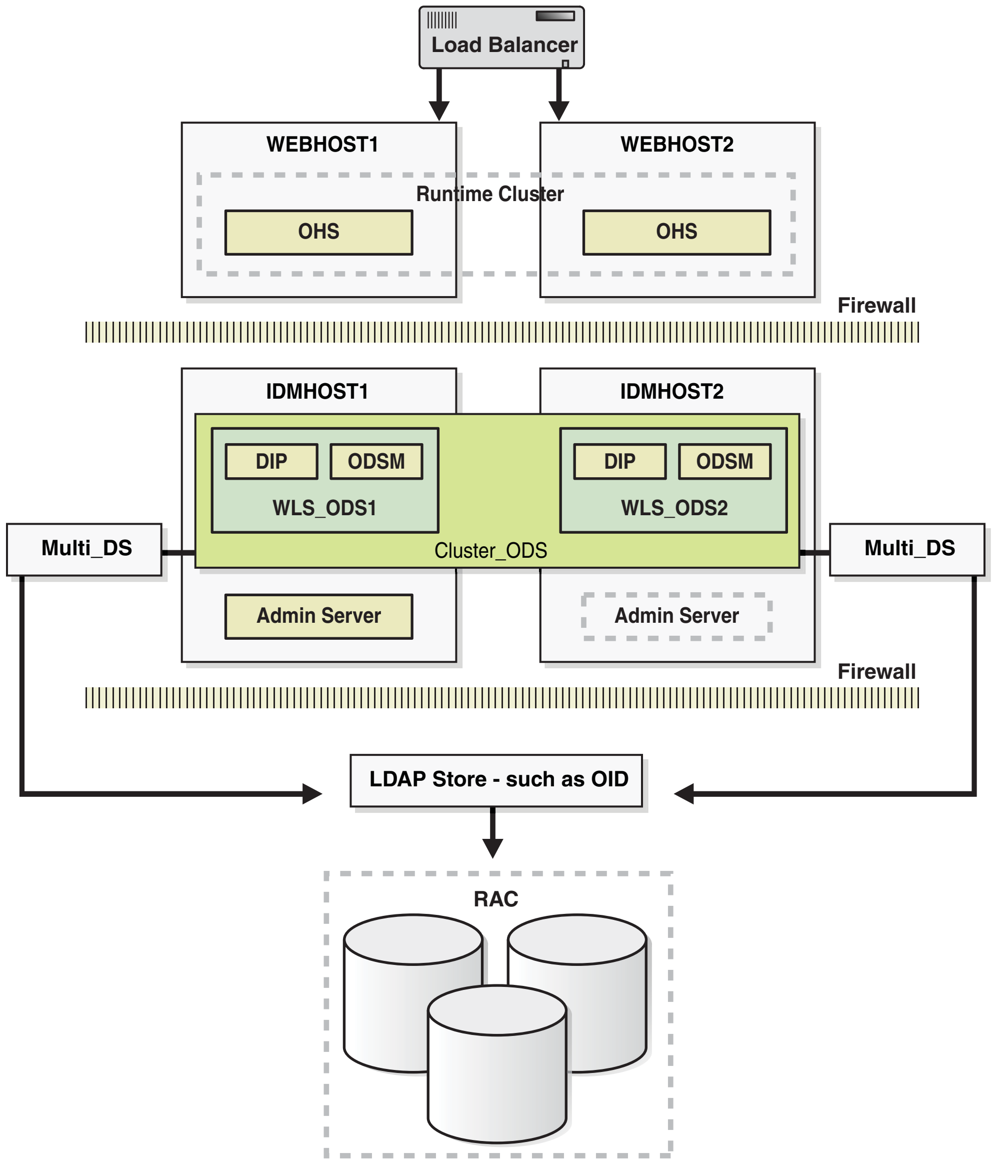
This section provides conceptual information about using Oracle Directory Services Manager in a high availability configuration.

In the Oracle Directory Services Manager high availability configuration described in this section, Oracle Directory Services Manager and Oracle Directory Integration Platform are installed and configured on two hosts in a two-node high availability active-active configuration.

7.6.2.1 Oracle Directory Services Manager High Availability Architecture

Figure 7–9 shows the Oracle Directory Integration Platform and Oracle Directory Services Manager high availability architecture in an active-active configuration.

Figure 7–9 Oracle Directory Services Manager and Oracle Directory Integration Platform in a High Availability Architecture



In Figure 7–9, the application tier includes the IDMHOST1 and IDMHOST2 computers.

On IDMHOST1, the following installations have been performed:

- An Oracle Directory Integration Platform instance and Oracle Directory Services Manager instance have been installed on the WLS_ODS1 Managed Server. The RAC database has been configured in a JDBC multi data source to protect the instances from RAC node failure.
- A WebLogic Administration Server has been installed. Under normal operations, this is the active Administration Server.

On IDMHOST2, the following installations have been performed:

- An Oracle Directory Integration Platform instance and Oracle Directory Services Manager instance have been installed in the WLS_ODS2 Managed Server. The

RAC database has been configured in a JDBC multi data source to protect the instances from RAC node failure.

The instances in the WLS_ODS2 Managed Server on IDMHOST2 and the instances in the WLS_ODS1 Managed Server on IDMHOST1 are configured as the CLUSTER_ODS cluster.

- A WebLogic Administration Server has been installed. Under normal operations, this is the passive Administration Server. You will make this Administration Server active if the Administration Server on IDMHOST1 becomes unavailable.

7.6.2.1.1 Starting and Stopping the Cluster In a high availability architecture, Oracle Directory Integration Platform and Oracle Directory Services Manager are deployed on an Oracle WebLogic Cluster that has at least two servers as a part of the cluster.

By default, the WebLogic Server starts, stops and monitors the applications. By default, both the Oracle Directory Integration Platform and Oracle Directory Services Manager applications leverage the high availability features of the underlying WebLogic Clusters. In case of hardware or other failures, session state is available to other cluster nodes that can resume the work of the failed node.

In a high availability environment, WebLogic Node Manager is configured to monitor the WebLogic servers. In case of failure, Node Manager restarts the WebLogic Server. If Node Manager cannot restart the server, then the front-ending load balancing router detects failure of a WebLogic instance in the Cluster and routes traffic to surviving instances.

7.6.2.2 Protection from Failures and Expected Behaviors

This section discusses protection from different types of failure in an Oracle Directory Services Manager active-active cluster.

7.6.2.2.1 Process Failure In a high availability environment, the Oracle Directory Services Manager applications are deployed on an Oracle WebLogic Server cluster comprised of at least two WebLogic instances.

By default, the Oracle Directory Services Manager applications leverage the high availability features of the underlying WebLogic Clusters. In case of hardware or other failures, session state is available to other cluster nodes that can resume the work of the failed node.

In addition, in a high availability environment, WebLogic Node Manager is configured to monitor the WebLogic servers. In case of failure, Node Manager restarts the WebLogic Server. If Node Manager cannot restart the server, then mod_wl_ohs, which is configured as a part of Oracle HTTP Server, routes the request to the surviving WebLogic instance.

OPMN monitors the Oracle HTTP Server processes and restarts the process in case of failure. If OPMN is unable to restart the HTTP process, the front-ending load balancing router detects the failure of an Oracle HTTP Server instance and routes traffic to surviving instances.

Oracle Directory Services Manager maintains a session state, but in case of failure, the session state information is not carried over to the surviving node.

7.6.2.2.2 Expected Client Application Behavior When Failure Occurs Oracle Directory Services Manager failover is not transparent. You have to reestablish the connection during an Oracle WebLogic Server instance failover using Oracle Directory Services Manager.

7.6.2.2.3 Expected Dependency Failure Oracle Directory Services Manager requires the WebLogic Managed Server to be available during startup. If it is not available, Oracle Directory Services Manager fails to start.

7.6.2.3 Oracle Directory Services Manager Prerequisites

This section describes prerequisites for setting up Oracle Directory Services Manager in the high availability architecture.

Note: Oracle requires that you synchronize the system clocks on the cluster nodes when you are using Oracle Directory Services Manager in a high availability deployment.

The components listed below should be installed and configured before installing Oracle Directory Services Manager. See the sections below for the prerequisites, the installation and configuration steps of each required component:

- Oracle database

  Install and configure an Oracle Real Application Clusters database by following the instructions in these sections:

  – Section 7.2, "Prerequisites for Oracle Identity Management High Availability Configuration"

  – Section 7.2.1, "Database Prerequisites"

  – Section 7.2.2, "Installing and Configuring the Database Repository"

- Oracle Repository Creation Utility

  Install the Oracle Repository Creation Utility to create and load the required schemas by following the instructions in these sections:

  – Section 7.2.3, "Installing the Repository Creation Utility Software"

  – Section 7.2.4, "Configuring the Database for Oracle Fusion Middleware 11g Metadata"

- Load balancers and virtual hosts

  See Section 7.2.4.5, "Configuring Virtual Server Names and Ports for the Load Balancer" for information about configuring the load balancer and creating the required virtual server.

- Oracle Internet Directory

  Install and configure Oracle Internet Directory by following the instructions in these sections:

  – Section 7.3.2.3, "Oracle Internet Directory Prerequisites"

  – Section 7.3.3, "Oracle Internet Directory High Availability Configuration Steps"

7.6.3 Oracle Directory Services Manager High Availability Configuration Steps

See Section 7.5.3, "Oracle Directory Integration Platform and Oracle Directory Services Manager High Availability Configuration Steps" for the steps for installing and configuring the high availability active-active configuration shown in Figure 7–9.

7.6.4 Validating Oracle Directory Services Manager High Availability

This section describes how to validate Oracle Directory Services Manager in a high availability configuration.

7.6.4.1 Performing a WebLogic Server Instance Failover

While you are accessing Oracle Directory Services Manager through Oracle HTTP Server, you can use the steps in this section to fail over a WebLogic Server instance and validate that Oracle Directory Services Manager is still accessible.

The Oracle HTTP Server virtual server name in this example is:

```
http://admin.mycompany.com
```

Perform these steps:

1. Access Oracle Directory Services Manager through the Oracle HTTP Server virtual server name:

   ```
   http://admin.mycompany.com/odsm/faces/odsm.jspx
   ```

2. When the Welcome to Oracle Directory Services Manager screen is displayed, open a web browser and enter the following URL:

   ```
   http://hostname:port/console
   ```

   where `hostname` is the DNS name of the Administration Server and `port` is the address of the port on which the Administration Server is listening for requests (7001 by default).

3. On the login page, enter the user name and the password you used to start the Administration Server and click **Log In**.

4. Shut down one of the Managed Servers where Oracle Directory Services Manager is deployed by following these steps:

   a. In the left pane of the Oracle WebLogic Server Administration Console, expand **Environment** and select **Servers**.

      See the "Starting and Stopping Oracle Fusion Middleware" chapter of the *Oracle Fusion Middleware Administrator's Guide* for information on starting and stopping WebLogic Servers.

   b. Click the name of the Managed Server that you want to shut down. For example, `wls_ods1`.

   c. In the settings for wls_ods1 screen, select the **Control --> Start/Stop** tab.

   d. Select the check box next to the name of the Managed Server (wls_ods1) that you want to shut down and click **Shutdown > When work completes**.

5. Check the status of the Managed Server (`wls_ods1`):

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

   b. The State for the Managed Server (`wls_ods1`) should be **SHUTDOWN**.

6. Access Oracle Directory Services Manager through the Oracle HTTP Server virtual server name:

   ```
   http://admin.mycompany.com/odsm/faces/odsm.jspx
   ```

   The Welcome to Oracle Directory Services Manager screen is displayed.

7.6.4.2 Performing a RAC Database Failover

While you are accessing Oracle Directory Services Manager through the load balancing router, you can follow the steps in this section to fail over one RAC database instance at a time and ensure that Oracle Directory Services Manager is still functional. This example checks Oracle Directory Services Manager as well as Oracle Internet Directory access to the database.

The Oracle HTTP Server virtual server name in this example is:

```
http://admin.mycompany.com
```

The LDAP virtual server name in this example is:

```
oid.mycompany.com:389
```

Perform these steps:

1. Access Oracle Directory Services Manager through the Oracle HTTP Server virtual server name:

   ```
   http://admin.mycompany.com/odsm/faces/odsm.jspx
   ```

   The Welcome to Oracle Directory Services Manager screen is displayed.

2. Verify that you can connect to Oracle Internet Directory using the LDAP virtual server:

   a. Select the **Connect to a directory --> Create A New Connection** link in the upper right hand corner.

   b. In the New Connection screen, fill in the connection information below and click **Connect**:

      * **Directory Type**: `OID`
      * **Name**: `OIDHA`
      * **Server**: `oid.mycompany.com`
      * **Port**: `389`
      * **SSL Enabled**: Leave blank
      * **User Name**: `cn=orcladmin`
      * **Password**: `********`
      * **Start Page**: `Home` (default)

3. Shut down the Oracle Internet Directory Schema database instance on INFRADBHOST1-VIP by running the following `srvctl` commands (in this example, the database name is INFRADB):

   ```
   ORACLE_HOME/bin/srvctl stop instance -d infradb -i infradb1
   ORACLE_HOME/bin/srvctl status database
   ```

4. Refresh the Oracle Directory Services Manager screen or navigate to one of the tabs (Home, Data Browser, Schema, Security, Advanced). You should still be able to access the Oracle Internet Directory information.

5. Restart the Oracle Internet Directory Schema database instance on INFRADBHOST1-VIP by running the following `srvctl` commands:

   ```
   ORACLE_HOME/bin/srvctl start instance -d infradb -i infradb1
   ORACLE_HOME/bin/srvctl status database
   ```

6. Repeat steps 3, 4, and 5 for INFRADBHOST2-VIP.

7.6.5 Oracle Directory Services Manager Failover and Expected Behavior

This section provides steps for using Oracle Directory Services Manager to validate a failover of a Managed Server, to validate a failover of an Oracle Internet Directory instance, and to validate a failover of a RAC database.

7.6.5.1 Using Oracle Directory Services Manager to Validate a Failover of a Managed Server

Follow these steps to use Oracle Directory Services Manager to validate a failover of a Managed Server:

1. In a web browser, launch the Oracle Directory Services Manager page using the Oracle HTTP Server host and port. For example:

   ```
   http://WEBHOST1:PORT/odsm/faces/odsm.jspx
   ```

2. Make a connection to Oracle Internet Directory.
3. Go to the Administration Console and stop the `wls_ods1` Managed Server.
4. Go back to the Oracle Directory Services Manager page and continue your work.
5. The Oracle Directory Services Manager page will be disconnected.
6. Re-launch the Oracle Directory Services Manager page using the Oracle HTTP Server host and port again from the same browser.
7. Reestablish a new connection.

The current behavior is the expected behavior. Oracle Directory Services Manager failover is not transparent. You have to reestablish the connection during a WebLogic Server instance failover using Oracle Directory Services Manager.

7.6.5.2 Using Oracle Directory Services Manager to Validate a Failover of an Oracle Internet Directory Instance

Follow these steps to use Oracle Directory Services Manager to validate a failover of an Oracle Internet Directory instance:

1. In a web browser, launch the Oracle Directory Services Manager page using the Oracle HTTP Server host and port. For example:

   ```
   http://WEBHOST1:PORT/odsm/faces/odsm.jspx
   ```

2. Make a connection to the Oracle Internet Directory hardware load balancer.
3. Shut down one Oracle Internet Directory instance at a time.
4. During the failover, go back to the Oracle Directory Services Manager page and continue your work.

It is expected behavior when Oracle Directory Services Manager displays a pop up window with a message that Oracle Internet Directory is down. Oracle Directory Services Manager will reestablish the connection to Oracle Internet Directory, but the connection may not be persistent during the failover. For additional information, see Section 7.6.6.4, "Oracle Directory Services Manager Displays "LDAP Server is down" Message During Oracle Internet Directory Failover."

While accessing Oracle Directory Services Manager, fail over the Oracle Internet Directory instances one at a time and ensure the LDAP store is still accessible. Oracle

Directory Services Manager might not have a persistent connection to Oracle Internet Directory.

7.6.5.3 Using Oracle Directory Services Manager to Validate a RAC Failover

Follow these steps to use Oracle Directory Services Manager to validate a RAC failover:

1. In a web browser, launch the Oracle Directory Services Manager page using the Oracle HTTP Server host and port. For example:

   ```
   http://WEBHOST1:PORT/odsm/faces/odsm.jspx
   ```

2. Connect to the Oracle Internet Directory from the Oracle Directory Services Manager page.
3. Shut down one RAC database instance at a time.
4. Go back to the Oracle Directory Services Manager page and continue your work from the Oracle Internet Directory connection established in Step 2.

It is expected behavior for Oracle Directory Services Manager to temporarily lose its connection during a RAC failover. See Section 7.6.6.5, "Oracle Directory Services Manager Temporarily Loses Its Connection During RAC Failover" for more information about the error message that displays in Oracle Directory Services Manager during a RAC failover.

While accessing Oracle Directory Services Manager through the hardware load balancer, perform a failover on one RAC database instance at a time and ensure that Oracle Directory Services Manager is still functional. This will check Oracle Directory Services Manager as well as Oracle Internet Directory access to the RAC database.

7.6.6 Troubleshooting Oracle Directory Services Manager

This section describes how to deal with issues involving Oracle Directory Services Manager high availability.

7.6.6.1 Dealing with Error Messages Received After Starting WebLogic Node Manager

If you receive the following error message after starting WebLogic Node Manager, follow the steps that appear below after the error message:

```
<Dec 15, 2008 8:40:05 PM> <Warning> <Uncaught exception in server handler:
javax.net.ssl.SSLKeyException: [Security:090482]BAD_CERTIFICATE alert was
received from stbee21.us.oracle.com - 152.68.64.2155. Check the peer to
determine why it rejected the certificate chain (trusted CA configuration,
hostname verification). SSL debug tracing may be required to determine the
exact reason the certificate was rejected.> javax.net.ssl.SSLKeyException:
[Security:090482]BAD_CERTIFICATE alert was received from stbee21.us.oracle.com -
152.68.64.215. Check the peer to determine why it rejected the certificate chain
(trusted CA configuration, hostname verification). SSL debug tracing may be
required to determine the exact reason the certificate was rejected.
```

1. If you have not already done so, in the Change Center of the Administration Console, click **Lock & Edit**.
2. In the left pane of the Console, expand **Servers** and AdminServer (admin).
3. Select the **Configuration > SSL > Advanced Link**.
4. Select **None** for **Hostname Verification**.

5. Click **Save** to save the setting.
6. To activate these changes, in the Change Center of the Administration Console, click **Activate Changes**.
7. Restart all servers.

If the Managed Server is started in Admin mode, perform these steps:

1. If you have not already done so, in the Change Center of the Administration Console, click **Lock & Edit**.
2. In the left pane of the Console, expand **Servers** and the name of the server that is running in ADMIN mode.
3. Select the **Control > Start/Stop** tab.
4. Select the name of the server.
5. Click **Resume**.
6. Select **Yes** to resume servers.

7.6.6.2 If WebLogic Node Manager Fails to Start

If WebLogic Node Manager fails to start, make sure that you have copied the following domains file from IDMHOST1 to IDMHOST2:

```
MW_HOME/wlserver_10.3/common/nodemanager/nodemanager.domains
```

7.6.6.3 Oracle Directory Services Manager Failover Using Oracle HTTP Server is Not Transparent

When you perform an Oracle Directory Services Manager failover using Oracle HTTP Server, the failover is not transparent. You will see this behavior when you perform the following steps:

1. Oracle Directory Services Manager is deployed in a high availability active-active configuration using Oracle HTTP Server.
2. Display an Oracle Directory Services Manager page using the Oracle HTTP Server name and port number.
3. Make a connection to an LDAP server, for example, an Oracle Internet Directory server or an Oracle Virtual Directory server.
4. Work with the LDAP server using the current Oracle Directory Services Manager Oracle HTTP Server host and port.
5. Shut down one Managed Server at a time using the Oracle WebLogic Server Administration Console.
6. Go back to the Oracle Directory Services Manager page and port, and the connection which was established earlier with Oracle Internet Directory or Oracle Virtual Directory. When you do, a message is displayed advising you to reestablish a new connection to the Oracle Directory Services Manager page.

In this situation, you must do the following:

1. In your web browser, exit the current Oracle Directory Services Manager page.
2. Launch a new web browser page and specify the same Oracle Directory Services Manager Oracle HTTP Server name and port.

3. Reestablish a new connection to the LDAP server you were working with earlier (Oracle Internet Directory or Oracle Virtual Directory).

7.6.6.4 Oracle Directory Services Manager Displays "LDAP Server is down" Message During Oracle Internet Directory Failover

In a high availability configuration where Oracle Directory Services Manager is connected to Oracle Internet Directory through a load balancer, Oracle Directory Services Manager displays the `LDAP Server is down` message during failover from one instance of Oracle Internet Directory to another.

The connection will be reestablished in less than a minute after the Oracle Internet Directory failover is complete, and you will be able to continue without logging in again.

7.6.6.5 Oracle Directory Services Manager Temporarily Loses Its Connection During RAC Failover

Oracle Directory Services Manager temporarily loses its connection to an Oracle Internet Directory instance that is using a RAC database during a RAC failover. Oracle Directory Services Manager may display the message `Failure accessing Oracle database (oracle errcode=errcode)`, where `errcode` is one of the following values: 3113, 3114, 1092, 28, 1041, or 1012.

The connection will be reestablished in less than a minute after the RAC failover is complete, and you will be able to continue without logging in again.

7.6.7 Additional Considerations for Oracle Directory Services Manager High Availability

When using Oracle Directory Services Manager to manage an Oracle Internet Directory cluster, use the load balancer virtual address as the connect string. However, when using Oracle Directory Services Manager to manage an Oracle Virtual Directory cluster, you must specify the host name and port for a specific Oracle Virtual Directory instance.

7.7 Oracle Identity Federation High Availability

This section provides an introduction to Oracle Identity Federation and describes how to design and deploy a high availability environment for Oracle Identity Federation.

7.7.1 Oracle Identity Federation Component Architecture

Oracle Identity Federation is a self-contained, standalone federation server that enables single sign-on and authentication in a multiple-domain identity network and supports the broadest set of federation standards. This allows users to federate in heterogeneous environments and business associations, whether or not they have implemented other Oracle Identity Management products in their solution set.

It can be deployed as a multi-protocol hub acting as both an Identity Provider (IdP) and Service Provider (SP).

Acting as an SP, Oracle Identity Federation enables you to manage your resources while off loading actual authentication of users to an IdP, without having to synchronize users across security domains out of band. Once authenticated at the IdP, the SP can allow or deny access to users for the SP's applications depending upon the local access policies.

If a user no longer has an account with the IdP, the federation is terminated and cross-domain single sign-on for that user is automatically disabled. As an IdP, Oracle Identity Federation allows you to manage and authenticate local users based upon federated requests from trusted providers.

Some key features of Oracle Identity Federation include support for:

- Multiple leading federation protocols such as SAML 1.x, SAML 2.0, WS-Federation, SAML 1.x/2.0 attribute sharing functionality, Liberty ID-FF 1.1/ Liberty ID-FF 1.2.
- Cross-protocol single sign-on and sign-out
- Affiliations, by allowing service providers to share their federation information, reducing the number of federations
- X.509 certificate validation
- Integration with Oracle Access Manager and Oracle Single Sign-On
- Integration with Oracle Internet Directory and support for a range of authentication engines, user data repositories and relational databases
- Implementing cross-site access and authentication in an environment containing both identity providers and service providers
- The ability to configure, enable, and disable external sites
- Accessing applications at destination sites using a single sign-on

Figure 7–10 Oracle Identity Federation Non-High Availability Architecture

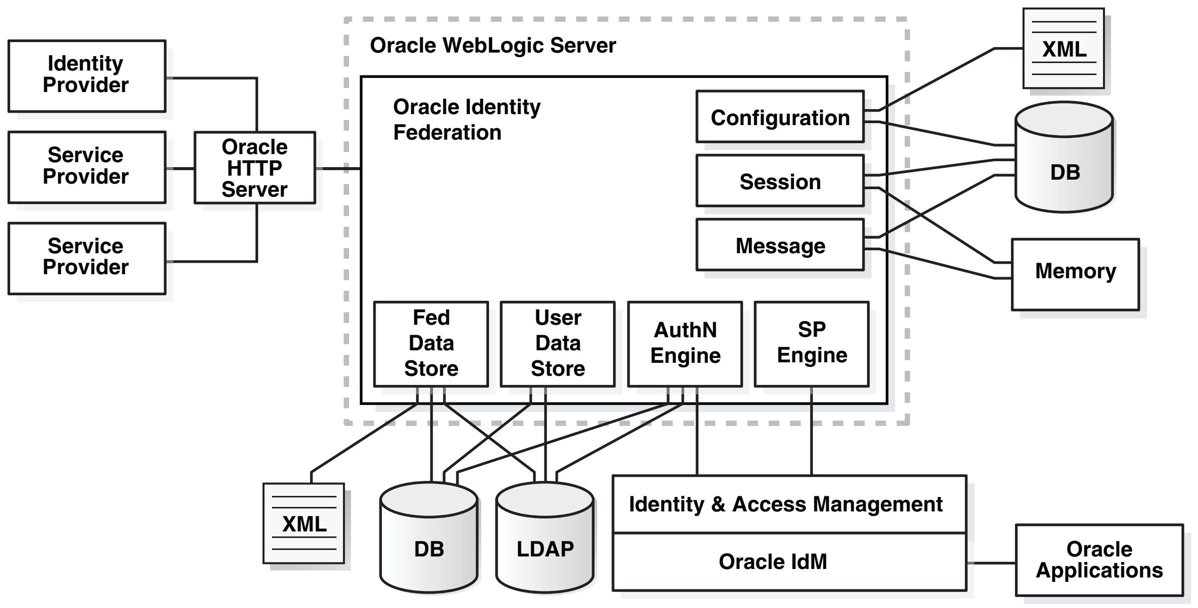


Figure 7–10 shows Oracle Identity Federation deployed on Oracle WebLogic Server in a non-high availability architecture and its relationship to other federation components.

Figure 7–10 shows Oracle Identity Federation as a member of Federation between other identity providers (IdP) and service providers (SP).

The Oracle Identity Federation authentication service can be configured:

- To enable single sign-on access to resources protected by Oracle Single Sign-On (with the Oracle Internet Directory user repository) or Oracle Access Manager (with various repositories). Resources can include Oracle Collaboration Suite, Oracle E-Business Suite, PeopleSoft modules, and so on.
- To interact with Identity and Access Management solutions such as LDAP directories, RDBMS databases, and Oracle Access Manager.

Note: In an environment where Oracle Identity Federation and Oracle Single Sign-On both protect resources, you can configure either component to serve as the authentication mechanism when a user requests access to a protected resource. Likewise, environments containing both Oracle Identity Federation and Oracle Access Manager provide similar functionality.

For more information about Oracle Identity Federation authentication, see *Oracle Fusion Middleware Administrator's Guide for Oracle Identity Federation*.

7.7.1.1 Oracle Identity Federation Component Characteristics

Oracle Identity Federation is Oracle's self-contained, standalone federation server, based on J2EE standards. It enables single sign-on and authentication in a multiple-domain identity network and supports a broad set of federation standards such as SAML 1.x, SAML 2.0, WS-Federation, and SAML 1.x/2.0 attribute sharing functionality.

By default, Oracle Identity Federation is deployed as an externally staged application on Oracle WebLogic Server.

7.7.1.1.1 Runtime Processes Oracle Identity Federation is a J2EE application that is deployed on Oracle WebLogic Server as an externally staged application. The Oracle Identity Federation server is initialized when the WebLogic Server it is deployed on starts up.

The Oracle Identity Federation application accesses the Credential Store Framework, to get the credentials to connect to the back-end servers. Once this is complete, the Oracle Identity Federation server is running and is ready to accept and serve requests.

7.7.1.1.2 Process Lifecycle Oracle Identity Federation is deployed to an Oracle WebLogic Server as an externally managed application. By default, the Oracle WebLogic Server starts, stops, monitors and manages other lifecycle events for the Oracle Identity Federation application.

The application is initialized when the Oracle WebLogic Server it is deployed to starts up and it stops when the WebLogic Server is shut down.

WebLogic Node Manager can be configured to monitor the server process and restart it in case of failure

The Oracle Enterprise Manager Fusion Middleware Control is used to monitor as well as modify the configuration of the application.

Starting and Stopping

Oracle Identity Federation lifecycle events can be managed using one or more of the command line tools and consoles listed below:

- Oracle WebLogic Scripting Tool (WLST)

- Oracle WebLogic Server Administration Console
- Oracle Enterprise Manager Fusion Middleware Control
- WebLogic Node Manager

7.7.1.1.3 Request Flow Users typically access applications in multiple domains through a corporate portal. For example, Alpha Corporation could have a Portal Server in place to manage Alpha's user logins, page personalization, and so on. The portal server might consist of homegrown logic running within an application server, or it might be a commercial product. Its partner, Beta Corporation, may serve its technical database application with a "MyBeta.com" type of portal. In that case, each company would operate its own portal server.

The processing flow is as follows:

1. The user logs into the Alpha portal whose access is being managed by a web access management (WAM) system such as Oracle Access Manager or Oracle Single Sign-On.
2. The user initiates a request by clicking on a resource that is actually hosted by Beta Corporation.
3. The Oracle Identity Federation instance at the Alpha portal, acting as the identity provider (IdP), will send the user information to the WAM system.
4. The WAM system will create a session after mapping a user to an identity in its local identity store.
5. The WAM system will return the successful response and the session token to the Oracle Identity Federation IdP server at the Alpha portal.
6. Using the above information, the IdP at the Alpha portal creates a SAML identity assertion and signs it using its private signing key. This response is sent to the Oracle Identity Federation instance acting as a Service Provider (SP) at Beta Corporation.
7. The Oracle Identity Federation server acting as SP at Beta Corporation verifies the signed response using the IdP's public certificate associated with its signing key.
8. The Oracle Identity Federation service provider at Beta Corporation extracts the assertion, and creates a user session for the assertion after mapping the user session to its local authorization system.
9. The Oracle Identity Federation service provider sends the user's browser a redirect to the requested resource.
10. The user's browser sends a request to the target resource over the user session created by the service provider.

7.7.1.1.4 Configuration Artifacts The configuration of the Oracle Identity Federation server is managed by MBeans. The configuration data for Oracle Identity Federation is stored in three main files:

- config.xml: contains server-wide configuration
- cot.xml: stores provider-specific configuration
- datastore.xml: stores back-end data store configuration

The Oracle Identity Federation application is deployed as an EAR file by the Oracle Identity Management 11*g* Installer. After the application is deployed, the exploded EAR file is stored on each Managed Server in the applications directory. The location is:

```
DOMAIN_HOME/servers/serverName/stage/OIF/11.1.1.1.0/OIF/
```

By default, *serverName* is `wls_oif1`.

Using Oracle Enterprise Manager Fusion Middleware Control to make configuration changes can help prevent inconsistent configuration across different nodes. This is especially true when the application is deployed in a high availability configuration. However, WLST scripts and JMX MBeans can also be used to configure the Oracle Identity Federation server.

7.7.1.1.5 External Dependencies The Oracle Identity Federation server is a J2EE application that is deployed on an Oracle WebLogic Managed Server. The Oracle Identity Federation server interacts with external repositories to store configuration data, runtime transient data (message, user session) and federation data and to authenticate users. The Oracle Identity Federation server requires these repositories to be available during initialization or during runtime. These are the external components required for the Oracle Identity Federation server to function:

- Oracle WebLogic Server
  - Administration Server
  - Managed Server
- Data Repositories
  - Message data store and user session data store (transient data store)
  - Configuration data store
  - User data store
  - Federation data store
- Authentication Engines
- Service Provider Engines

See the following subsections for more information about how these external components function with Oracle Identity Federation.

Oracle WebLogic Server

The Oracle Identity Federation server is a J2EE application that is deployed on an Oracle WebLogic Managed Server. The server is administered and managed using the Oracle Enterprise Manager Fusion Middleware Control.

The application cannot start up if the Managed Server is not running. If the Administration Server is not available, the server will continue to run in its current state, but changes cannot be made to its configuration.

Data Repositories

The Oracle Identity Federation server uses the various external data repositories to store configuration, user session, message and federation data. The server requires these data stores to be available during initialization, runtime or both. Details of the repositories are shown below:

- Message data store and user session data store (transient data store):

  The Oracle Identity Federation server uses the message data store and the user session data store for storing transient data like federation protocol/session state. The message data store together with the user session data store is also referred to as a the transient data store.

Depending on the deployment option for the Oracle Identity Federation server, the transient data can either be stored in memory or in a relational database. Table 7–7 shows the relationship between the deployment option and the required transient data store type:

Table 7–7 Transient Data Store Types for Oracle identity Federation Deployment Options

Deployment Option	Store Type
Non-high Availability/ Standalone	In Memory Store Relational Database Store
High Availability	Relational Database Store

- Configuration data store:

  The Oracle Identity Federation application uses the configuration data store to store its configuration artifacts. Depending on the deployment option of the Oracle Identity Federation server, the configuration store can either be stored in an XML file or in a relational database.

  Table 7–8 shows the relationship between the deployment option and the required configuration data store type:

Table 7–8 Configuration Data Store Types for Oracle Identity Federation Deployment Option

Deployment Option	Store Type
Non-high Availability/ Standalone	XML Relational Database Store
High Availability	Relational Database Store

  The Oracle Identity Federation application connects to the configuration data store to retrieve its configuration data during the start up process. The application will fail to start up if the configuration data store is not available.

- User data store:

  The Oracle Identity Federation server uses the user data store to locate users after local authentication or after processing an incoming SAML Assertion and to retrieve user specific attributes. The user data repository can be either a relational database or a LDAP repository.

  The Oracle Identity Federation server is certified to work with LDAP repositories such as Oracle Internet Directory, Sun Java System Directory Server, and Microsoft Active Directory, as well as with a relational database.

  The role played by the user data repository depends on whether Oracle Identity Federation will be configured as an identity provider (IdP) or a service provider (SP).

  – When configured as an IdP:

    * Oracle Identity Federation uses the repository to verify user identities and to build protocol assertions.

  – When configured as an SP:

  * Oracle Identity Federation uses the repository to map information in received assertions to user identities at the destination, and subsequently to authorize users for access to protected resource.
  * When creating a new federation, Oracle Identity Federation uses the repository to identify the user and link the new federation to that user's account.

  The Oracle Identity Federation application does not require the user data store to be available while starting up. The user data store is required at runtime.

  > **Note:** Oracle Identity Federation is only certified to work with Oracle Database versions 10.2.0.4 and higher or Oracle Database 11.1.0.7 and higher.

- Federation Data Store:

  The federation data store can be XML, RDBMS or LDAP. In high availability mode, use only RDBMS or LDAP so that the federation data store is available to all members of the cluster.

  The Oracle Identity Federation server uses the federation data store to persist user federation records. Identity federation is the linking of two or more accounts that a principal may hold with one or more identity providers or service providers within a given circle of trust. When users federate the otherwise isolated accounts they have with businesses (known as their local identities), they are creating a relationship between two entities, an association comprising any number of service providers and identity providers.

  The Oracle Identity Federation server is certified with industry-standard LDAP repositories including Oracle Internet Directory, Sun Java System Directory Server, and Microsoft Active Directory or a relational database.

  Table 7–9 shows the relationship between the deployment option and the required federation data store type:

Table 7–9 Federation Data Store Types for Oracle Identity Federation Deployment Option

Deployment Option	Store Type
Non-high Availability/ Standalone	None or XML
High Availability	None, LDAP, or Relational Database Store

  The federation data store is required at runtime only if persistent name identifier formats are used during protocol exchanges. The federation data store is optional at runtime if non-opaque name identifier formats are used, such as email address.

  > **Note:** Oracle Identity Federation is only certified to work with Oracle Database versions 10.2.0.4 and higher or Oracle Database 11.1.0.7 and higher.

  For more information about the federation data store for Oracle Identity Federation, see the "Federation Data Store" section in *Oracle Fusion Middleware Administrator's Guide for Oracle Identity Federation*.

- Authentication engines:

  Oracle Identity Federation IdP and SP protocol operations, such as single sign-on, federation creation, federation termination, and NameID registration, require users to be authenticated. The Oracle Identity Federation authentication module handles user authentication. The authentication module can perform two distinct roles in user authentication:

  – When Oracle Identity Federation is deployed as an Identity Provider, the authentication module acts as a local authentication mechanism. The authentication module can delegate authentication or directly interact with a number of repositories and identity management solutions.

  – When Oracle Identity Federation is deployed as a service provider, it uses federation protocols to have the user authenticated at a peer identity provider. Oracle Identity Federation then forwards the user to the authentication module, which propagates and creates an authenticated user session in the deployed identity management solution at the SP. In turn, this enables access to the requested protected resource.

  The Oracle database, Oracle Internet Directory, Sun Java System Directory Server, and Microsoft Active Directory are some examples of supported RDBMS and LDAP repositories.

  Oracle Access Manager and Oracle Single Sign On are examples of supported repositories and Identity Management solutions.

  The authentication module only requires the external authentication engines to be available at runtime.

- Service provider engines:

  A Service Provider Integration Engine (SP Integration Engine) is used to create a user authenticated session at the Identity and Access Management (IAM) server. Oracle Identity Federation is shipped with an SP engine includes several internal plug-ins that allow it to interact with different IAM servers, such as:

  – Oracle Single Sign-On

  – Oracle Access Manager

  – Oracle Identity Federation Test Application

  Oracle Identity Federation also provides a framework to integrate with third party IAM frameworks; the customized SP Integration Module will interact with Oracle Identity Federation using internal J2EE Servlet forwards, and it will communicate with the third party IAM system to create the user authenticated session.

  Any custom SP Engine modules should be deployed on each Managed Server in the cluster.

7.7.1.1.6 Oracle Identity Federation Log File Location Oracle Identity Federation is a J2EE application deployed on Oracle WebLogic Server. All server related logs messages are logged in the server log file and all Oracle Identity Federation specific messages are logged into the diagnostic log file of the Oracle WebLogic Server where the application is deployed.

The default location of the Oracle WebLogic Server log file is:

```
WEBLOGIC_SERVER_HOME/user_projects/domains/domainName/servers/serverName/logs/
serverName-diagnostic.log
```

The Oracle Identity Federation log file is named `serverName-diagnostic.log`. For example, if the Oracle WebLogic Server name is `wls_oif1`, then the log file name is `wls_oif1-diagnostic.log`.

Use the Oracle Enterprise Manager Fusion Middleware Control to view the log files.

7.7.2 Oracle Identity Federation High Availability Concepts

This section provides an introduction to Oracle Identity Federation concepts and describes how to design and deploy a high availability environment for Oracle Identity Federation.

Listed below are a few guidelines to follow when deploying Oracle Identity Federation in a highly availability configuration:

- The transient and configuration data should always be stored in a shared relational database. This is required when:
  - Oracle Identity Federation acts as an IdP and the Artifact SSO profile is used. In this case, the user might interact with IdP #1 where the assertion will be created, and later when the artifact will de-referenced, the SP will connect to IdP #2. If the two Oracle Identity Federation servers are not sharing the same message store, then IdP #2 will not be able to locate the assertion for the artifact created by IdP #1.
  - Oracle Identity Federation acts as an SP and the POST SSO profile is used. In the POST profile, the assertion is being carried by the user's browser, thus opening the possibility of that assertion being re-submitted to the Oracle Identity Federation SP. Because of potential replay attacks, the Oracle Identity Federation SP Server keeps a trace of the assertion it received, so that no assertion can be used twice.
  - Oracle Identity Federation acts as an Authentication Query Authority, where the Oracle Identity Federation Server will answer queries from remote providers by returning assertions representing the existing sessions for a specific user at the Oracle Identity Federation Server. For this reason, the Oracle Identity Federation servers will need to share the transient session stores.
  - Oracle Identity Federation acts as an IdP/Attribute Authority and has the Assertion ID Responder functionalities enabled. This feature allows remote providers to query the IdP to retrieve assertions that were already created during previous SAML flows. For that reason, the Responder needs to have access to a store containing all the assertions created.
- In a high availability environment, it is recommended to enable sticky sessions in the load balancer and disable HTTP session replication for performance reasons. HTTP session replication replicates session information across nodes and is memory intensive. HTTP session replication is not enabled by default.

  Note: HTTP session state is used at runtime by Oracle Identity Federation when processing HTTP requests. The information stored in the HTTP session state is short-lived. The life of the information corresponds to the duration of the federation operation, such as single sign-on.

- Sticky sessions should be enabled in the load balancer so that a request from a user goes to the same Oracle Identity Federation server for that session.

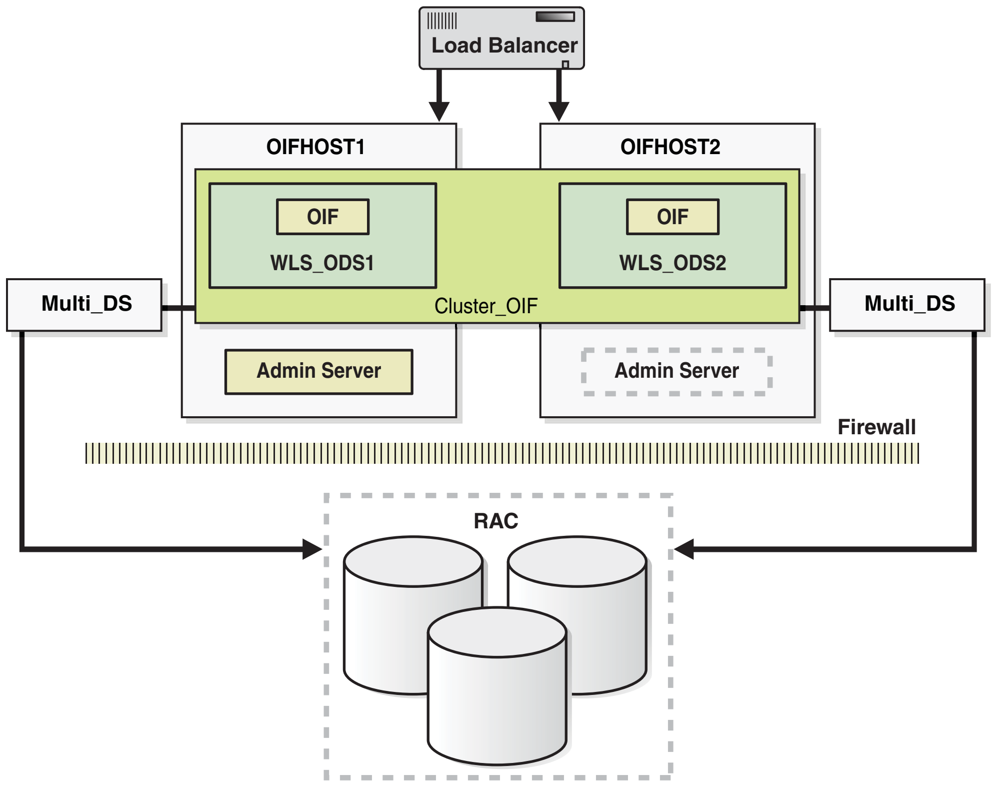
Note: See Section 7.2.4.5.1, "Load Balancers" for a description of the features to enable in the load balancer that is required when you deploy Oracle Identity Management software in a high availability configuration.

- The Oracle Identity Federation server supports two types of HTTP Session replication:
  - In-memory session replication
  - JDBC session replication
- These are some characteristics of in-memory session replication:
  - All Managed Servers running the Oracle Identity Federation application in a cluster have to be up when the session is created.
  - It can slow down performance.
  - If a server receives a request before session replication is completed, the server throws an error. To avoid this, set the following parameters using the Oracle Enterprise Manager Fusion Middleware Control:
    - sessionreplicationenabled: Set this to true.
    - sessionreplicationtimeout: This is set to 2000 by default. Increase this if necessary.
- These are some characteristics of JDBC session replication:
  - Robust in nature.
  - Slower in performance than in-memory session replication due to the additional overhead of database calls.
  - The session persistence state is stored in a shared relational database.
  - To configure Oracle WebLogic Server for JDBC session persistence see "Using a Database for Persistent Storage (JDBC persistence)" in *Oracle Fusion Middleware Developing Web Applications, Servlets, and JSPs for Oracle WebLogic Server*.

7.7.2.1 Oracle Identity Federation High Availability Architecture

Figure 7–11 shows the Oracle Identity Federation high availability architecture in an active-active configuration.

Figure 7–11 Oracle Identity Federation in a High Availability Architecture



In Figure 7–11, the application tier includes the IDMHOST1 and IDMHOST2 computers.

On IDMHOST1, the following installations have been performed:

- An Oracle Identity Federation instance has been installed in the WLS_OIF1 Managed Server. The RAC database has been configured in a JDBC multi data source to protect the instance from RAC node failure.
- A WebLogic Server Administration Server has been installed. Under normal operations, this is the active Administration Server.

On IDMHOST2, the following installations have been performed:

- An Oracle Identity Federation instance has been installed in the WLS_OIF2 Managed Server. The RAC database has been configured in a JDBC multi data source to protect the instance from RAC node failure.

  The instances in the WLS_OIF2 Managed Server on IDMHOST2 and the instances in the WLS_OIF1 Managed Server on IDMHOST1 are configured as the CLUSTER_OIF cluster.
- A WebLogic Server Administration Server has been installed. Under normal operations, this is the passive Administration Server. You make this Administration Server active if the Administration Server on IDMHOST1 becomes unavailable.

7.7.2.1.1 Starting and Stopping the Cluster In a high availability architecture, Oracle Identity Federation server is deployed on an Oracle WebLogic Cluster, which has at least two servers as a part of the cluster.

By default, Oracle WebLogic Server starts stops, monitors and manages the various lifecycle events for the application. The Oracle Identity Federation application leverages the high availability features of the underlying Oracle WebLogic Clusters. In case of hardware or other failures, session state is available to other cluster nodes that can resume the work of the failed node.

In a high availability environment, WebLogic Node Manager is configured to monitor the Oracle WebLogic Servers. In case of failure, Node Manager restarts the WebLogic Server.

In an high availability environment, a hardware load balancer is used to load balance requests between the multiple Oracle Identity Federation instances. If one of the Oracle Identity Federation instances fails, the load balancer detects the failure and routes requests to the surviving instances.

In high availability environments, the state and configuration information is stored a database that is shared by all the members of the cluster.

The surviving Oracle Identity Federation instances will continue to seamlessly process any unfinished transactions started on the failed instance since the state information is in the shared database and is available to all the members in the cluster.

When an Oracle Identity Federation instance fails, its database and LDAP connections are released. The surviving instances in the active-active deployment make their own connections to continue processing unfinished transactions on the failed instance.

You can use one of the following command line tools or consoles to manage the lifecycle events for the Oracle Identity Federation application:

- Oracle WebLogic Server Administration Console
- Oracle Enterprise Manager Fusion Middleware Control
- Oracle WebLogic Scripting Tool (WLST)

7.7.2.1.2 Cluster-Wide Configuration Changes Configuration changes made through one cluster member will be propagated automatically to all others because the configuration is stored in the shared database.

To turn session replication on or off, updates must be made in the following weblogic.xml file on all Managed Servers where Oracle Identity Federation is deployed:

```
DOMAIN_HOME/servers/serverName/stage/OIF/11.1.1.1.0/OIF/web.war/WEB-INF/
weblogic.xml
```

By default, session replication is disabled. If session replication is disabled, sticky sessions must be enabled on the load balancer. For information on setting up sticky sessions, see Section 7.7.3.6.2, "Oracle Identity Federation Configuration."

Follow the appropriate step below to enable or disable session replication:

- To enable session replication, set `persistent-store-type` to `replicated_if_clustered`. Then save the `weblogic.xml` file and restart Oracle WebLogic Server.
- To disable session replication, set `persistent-store-type` to `memory`. Then save the `weblogic.xml` file and restart Oracle WebLogic Server.

Using Oracle Enterprise Manager Fusion Middleware Control to make configuration changes can help prevent inconsistent configuration across different nodes. This is especially true when the application is deployed in a high availability configuration.

However, WLST scripts and JMX MBeans can also be used to configure the Oracle Identity Federation server.

7.7.2.2 High Availability Considerations for Integration with Oracle Access Manager

This section describes the steps to take when you are integrating Oracle Identity Federation with Oracle Access Manager in a high availability topology:

1. In addition to deploying Oracle Identity Federation in high availability mode, Oracle Access Manager should also be deployed in high availability mode.
2. The Oracle Access Server SDK must be installed on every Oracle Identity Federation server in the cluster. Oracle Identity Federation must be configured to reference the directory where the SDK is installed. If the SDK is installed in the Domain Home directory, then you can reference the SDK folder using a relative path from the Domain Home folder. If the SDK is installed elsewhere, Oracle Identity Federation will need to reference the SDK folder using an absolute path.

   When Oracle Identity Federation is used in a high availability environment, it is recommended that the Access Server SDK be installed under the Domain Home folder, using the same directory name on all the computers where Oracle Identity Federation is installed. This is a requirement for Oracle Identity Federation high availability integration with Oracle Access Manager because all the Oracle Identity Federation instances will share the same configuration, specifically the directory where the Access Server SDK is installed. Using a relative path allows the Oracle Identity Federation instances to share the same configuration.
3. Follow the instructions for integrating Oracle Access Manager as an SP integration module in the "Integrate Oracle Access Manager as an SP Integration Module" section in *Oracle Fusion Middleware Administrator's Guide for Oracle Identity Federation* to integrate Oracle Access Manager with the SDK instance on each Oracle Identity Federation server.
4. Copy over the required files to the domain library and update the WebLogic Server startup script for each Oracle Identity Federation server to add the SDK jar file to the classpath and to set the LD_LIBRARY_PATH And LD_ASSUME_KERNEL environment variables, if required. See the "Update the Oracle WebLogic Server Environment" section in *Oracle Fusion Middleware Administrator's Guide for Oracle Identity Federation* for more information.

7.7.2.3 Oracle Identity Federation Prerequisites

Oracle Identity Federation requires the following components:

- Oracle JRockit Version 1.6.0 SDK that is bundled with the installation.
- Oracle WebLogic Server that is bundled with the installation
- User data store. This is typically an LDAP directory, but can optionally be a database store.
- Federation data store. This is a standard LDAP directory such as Oracle Internet Directory, Microsoft Active Directory or Sun Java System Directory Server.

Note: A user federation data store is not absolutely required for Oracle Identity Federation in all cases: it is required for Liberty 1.x and SAML 2.0 opaque persistent identifiers, but is optional for SAML 1.x, WS-Federation, and SAML 2.0 non-opaque identifiers (such as email address, subject DN, and so on).

- Oracle Database version 10.2.0.4.0 and higher or 11.1.0.7 and higher for the RDBMS transient data store.
- Oracle HTTP Server for proxy implementation; this is the only proxy server supported by Oracle Identity Federation, and is bundled with the installation.

Note: Oracle requires that you synchronize the system clocks on the cluster nodes when you are using Oracle Identity Federation in a high availability deployment.

7.7.2.3.1 Using RCU to Create Oracle Identity Federation Schemas in the Repository Before you can install the Oracle Internet Directory instances on OIFHOST1 and OIFHOST2, you must use the Repository Creation Utility (RCU) to create the collection of schemas used by Oracle Identity Federation.

The Repository Creation Utility (RCU) ships on its own CD as part of the Oracle Fusion Middleware 11g kit.

Follow these steps to run RCU and create the Oracle Identity Federation schemas in a RAC database repository:

1. Issue this command:

   prompt> `RCU_HOME/bin/rcu &`

2. On the Welcome screen, click **Next**.
3. On the Create Repository screen, select the **Create** operation to load component schemas into an existing database.

   Click **Next**.

4. On the Database Connection Details screen, enter connection information for the existing database as follows:

   Database Type: `Oracle Database`

   Host Name: Name of the computer on which the database is running. For a RAC database, specify the VIP name or one node name. Example: `INFRADBHOST1-VIP` or `INFRADBHOST2-VIP`

   Port: The port number for the database. Example: `1521`

   Service Name: The service name of the database. Example: `idmedg.mycompany.com`

   Username: `SYS`

   Password: The SYS user password

   Role: `SYSDBA`

   Click **Next**.

5. On the Select Components screen, create a new prefix and select the components to be associated with this deployment:

   Create a New Prefix: `idm`

   Components: Select **Identity Management** (Oracle Identity Federation - OIF). De-select all other schemas.

   Click **Next**.

6. On the Schema Passwords screen, enter the passwords for the mail and additional (auxiliary) schema users.

   Click **Next**.

7. On the Map Tablespaces screen, select the tablespaces for the components.
8. On the Summary screen, click **Create**.
9. On the Completion Summary screen, click **Close**.

7.7.3 Oracle Identity Federation High Availability Configuration Steps

In a high availability environment, it is recommended that Oracle WebLogic Server utilities be used for clustering, load balancing, and failover of Oracle Identity Federation instances.

Make sure that the schema database is running, then follow these steps to install.

This section describes the steps to install and configure Oracle Identity Federation instances on OIFHOST1 and OIFHOST2:

- Section 7.7.3.1, "Installing Oracle WebLogic Server"
- Section 7.7.3.2, "Installing Oracle Identity Federation on OIFHOST1"
- Section 7.7.3.3, "Creating boot.properties for the Administration Server on OIFHOST1"
- Section 7.7.3.4, "Installing Oracle Identity Federation on OIFHOST2"
- Section 7.7.3.5, "Post-Installation Steps for Oracle Identity Federation"
- Section 7.7.3.6, "Configuring the Load Balancer"

7.7.3.1 Installing Oracle WebLogic Server

On OIFHOST1 and OIFHOST2, start the Oracle WebLogic Server installation by running the installer executable file.

Start the Oracle WebLogic Server installer as follows:

- On UNIX (Linux in the following example):

  ```
  ./server103_linux32.bin
  ```

  On Windows:

  ```
  server103_win32.exe
  ```

Then follow these steps in the installer to install Oracle WebLogic Server on the computer:

1. On the Welcome screen, click **Next**.

2. On the Choose Middleware Home Directory screen, choose a directory on your computer into which the Oracle WebLogic software is to be installed. This directory is called the Fusion Middleware Home directory.

   For the **Middleware Home Directory**, specify this value:

   ```
   /u01/app/oracle/product/fmw
   ```

   Click **Next**.

3. On the Choose Install Type screen, the installation program displays a window in which you are prompted to indicate whether you wish to perform a complete or a custom installation.

   Choose **Typical** or **Custom** (**Typical** is chosen in this example)

   Click **Next**.

4. On the Register for Security Updates screen, enter your "My Oracle Support" UserName and Password.

   Click **Next**.

5. On the Choose Install Type screen, specify the type of installation you wish to perform. Select either **Typical** or **Custom**.

   Choose **Typical**.

   Click **Next**.

6. On the Choose Product Installation Directory screen, specify the directory into which the Oracle WebLogic Server binaries should be installed.

   Specify the following values for this field:

   - WebLogic Server Directory:

     ```
     /u01/app/oracle/product/fmw/wlserver_10.3
     ```

   Click **Next**.

7. On the Installation Summary screen, the window contains a list of the components you selected for installation, along with the approximate amount of disk space to be used by the selected components once installation is complete.

   Verify that the choices you made are correct. If they are not, click the **Back** button to make the necessary changes.

   Click **Next**.

8. On the Installation Progress screen, view the installation is in progress.

9. On the Installation Complete screen, deselect the **Run Quickstart** checkbox.

   Click **Done**.

7.7.3.2 Installing Oracle Identity Federation on OIFHOST1

Follow the steps below to install and configure the first instance of Oracle Identity Federation on OIFHOST1.

1. Ensure that the system, patch, kernel and other requirements are met. These are listed in the *Oracle Fusion Middleware Installation Guide for Oracle Identity Management* in the Oracle Fusion Middleware documentation library for the platform and version you are using.

2. Ensure that port number 7499 is not in use by any service on the computer by issuing these commands for the operating system you are using. If a port is not in use, no output is returned from the command.

   On UNIX:

   ```
   netstat -an | grep "7499"
   ```

   On Windows:

   ```
   netstat -an | findstr :7499
   ```

3. If the port is in use (if the command returns output identifying the port), you must free the port.

   On UNIX:

   Remove the entry for port 7499 in the `/etc/services` file and restart the services, or restart the computer.

   On Windows:

   Stop the component that is using the port.

4. Copy the `staticports.ini` file from the `Disk1/stage/Response` directory to a temporary directory.

5. Edit the `staticports.ini` file that you copied to the temporary directory to assign the following custom port (uncomment the line where you specify the port number for Oracle Identity Federation):

   ```
   # The OIF Server Port
   OIF Server Port = 7499
   ```

6. Start the Oracle Identity Management 11*g* Installer as follows:

   On UNIX, issue this command: `runinstaller`

   On Windows, double-click `setup.exe`.

   The `runInstaller` and `setup.exe` files are in the `../install/`*`platform`* directory, where *platform* is a platform such as Linux or Win32.

   This displays the Specify Oracle Inventory screen.

7. On the Specify Inventory Directory screen, enter values for the Oracle Inventory Directory and the Operating System Group Name. For example:

   Specify the Inventory Directory: `/u01/app/oraInventory`

   Operating System Group Name: `oinstall`

   A dialog box appears with the following message:

   "Certain actions need to be performed with root privileges before the install can continue. Please execute the script /u01/app/oraInventory/createCentralInventory.sh now from another window and then press "Ok" to continue the install. If you do not have the root privileges and wish to continue the install select the "Continue installation with local inventory" option"

   Log in as root and run the "/u01/app/oraInventory/createCentralInventory.sh"

   This sets the required permissions for the Oracle Inventory Directory and then brings up the Welcome screen.

Note: The Oracle Inventory screen is not shown if an Oracle product was previously installed on the host. If the Oracle Inventory screen is not displayed for this installation, make sure to check and see:

1. If the `/etc/oraInst.loc` file exists
2. If the file exists, the Inventory directory listed is valid
3. The user performing the installation has write permissions for the Inventory directory

---

8. On the Welcome screen, click **Next**.
9. On the Select Installation Type screen, select **Install and Configure** and then click **Next**.
10. On the Prerequisite Checks screen, the installer completes the prerequisite check. If any checks fail, please fix them and restart your installation.

    Click **Next**.
11. On the Select Domain screen, select **Create a New Domain** and specify these values:

    HostName: OIFHOST1.MYCOMPANY.COM

    Port: 7001

    UserName: weblogic

    User Password: <password for weblogic user>

    Click **Next**.
12. On the Specify Installation Location screen, specify the following values:

    - **Oracle Middleware Home Location**:

      ```
      /u01/app/oracle/product/fmw
      ```
    - **Oracle Home Directory**: `oif`
    - **WebLogic Server Directory**:

      ```
      /u01/app/oracle/product/fmw/wlserver_10.3
      ```
    - **Oracle Instance Location**:

      ```
      /u01/app/oracle/admin/oif_inst1
      ```
    - **Instance Name**: `oif_inst1`

    Note: Ensure that the Oracle Home Location directory path for OIFHOST1 is the same as the Oracle Home Location path for OIFHOST2. For example, if the Oracle Home Location directory path for OIFHOST1 is: `/u01/app/oracle/product/fmw/oif`, then the Oracle Home Location directory path for OIFHOST2 must also be `/u01/app/oracle/product/fmw/oif`.

    Click **Next**.

13. On the Specify Oracle Configuration Manager Details screen, specify the values shown in the example below:

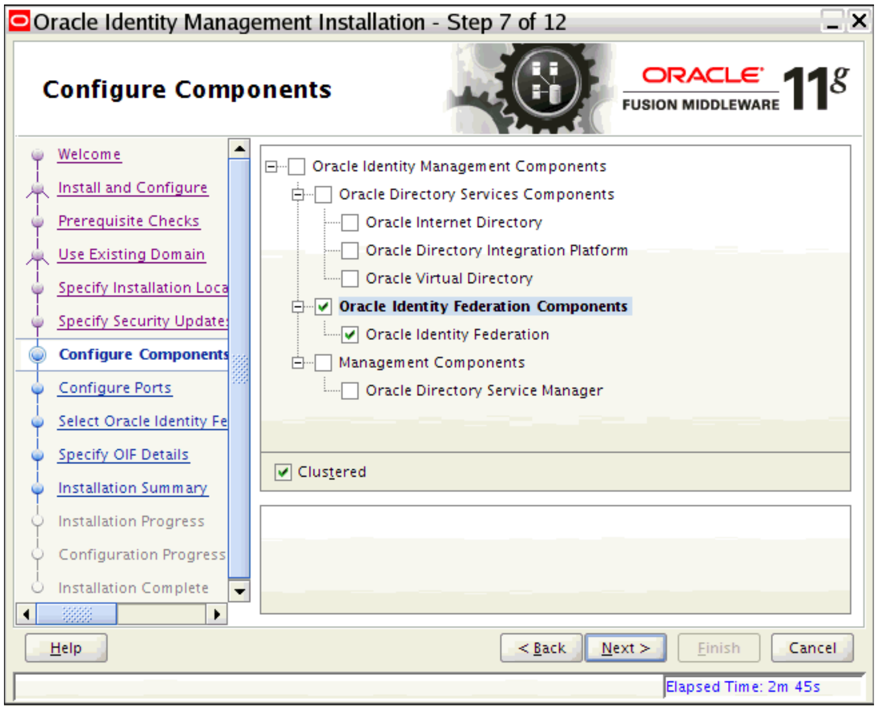
    - **Email Address**: Provide the email address for your My Oracle Support account.
    - **Oracle Support Password**: Provide the password for your My Oracle Support account.
    - Check the checkbox next to the **I wish to receive security updates via My Oracle Support** field.

    Click **Next**.

14. On the Configure Components screen, de-select all the components except **Oracle Identity Federation components**. The Oracle Identity Federation components include Oracle Identity Federation and Oracle HTTP Server. Select the **Clustered** checkbox.

    Click **Next**.

    Note: The default Oracle WebLogic Server clustering mode set by the installer is unicast (not multicast).



- [ ] Oracle Identity Management Components
  - [ ] Oracle Directory Services Components
    - [ ] Oracle Internet Directory
    - [ ] Oracle Directory Integration Platform
    - [ ] Oracle Virtual Directory
  - [✓] Oracle Identity Federation Components
    - [✓] Oracle Identity Federation
  - [ ] Management Components
    - [ ] Oracle Directory Service Manager

[✓] Clustered

15. On the Configure Ports screen, select **Specify Ports using Configuration File**. Provide the path to the `staticports.ini` file that you copied to the temporary directory.

    Click **Next**.

16. On the Specify OIF Details screen, specify these values:

    - **PKCS12 Password**: `<password>`
    - **Confirm Password**: `<confirm the password entered above>`

- **Server Id**: `oif_OIFDomain`

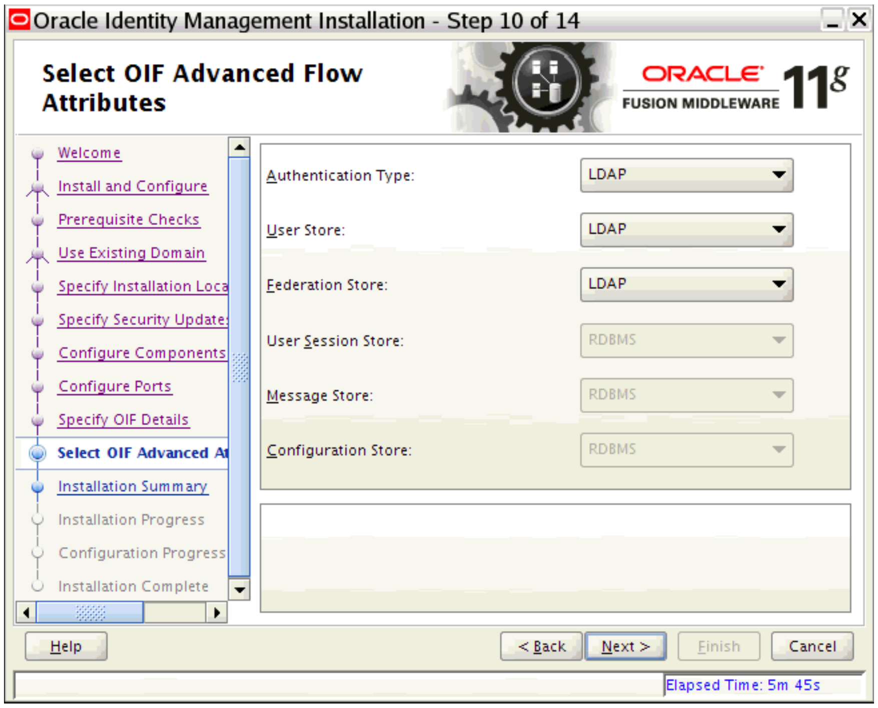
Click **Next**.

17. On the Select OIF Advanced Flow Attributes screen, specify these values:

    - **Authentication Type**: `LDAP`
    - **User Store**: `LDAP`
    - **Federation Store**: `LDAP`
    - **User Session Store**: `RDBMS` (default selection, which cannot be changed for a cluster)
    - **Message Store**: `RDBMS` (default selection, which cannot be changed for a cluster.
    - **Configuration Store**: `RDBMS` (default selection, which cannot be changed for a cluster.

Note: When you choose `RDBMS` for the session, message, and configuration data stores during an Advanced installation, the installer creates one data source for all three data stores.

If you want to have separate databases for each of these stores, you must configure this after the installation.



Authentication Type: LDAP
User Store: LDAP
Federation Store: LDAP
User Session Store: RDBMS
Message Store: RDBMS
Configuration Store: RDBMS

18. On the Authentication LDAP Details screen, specify the values shown in the example below:

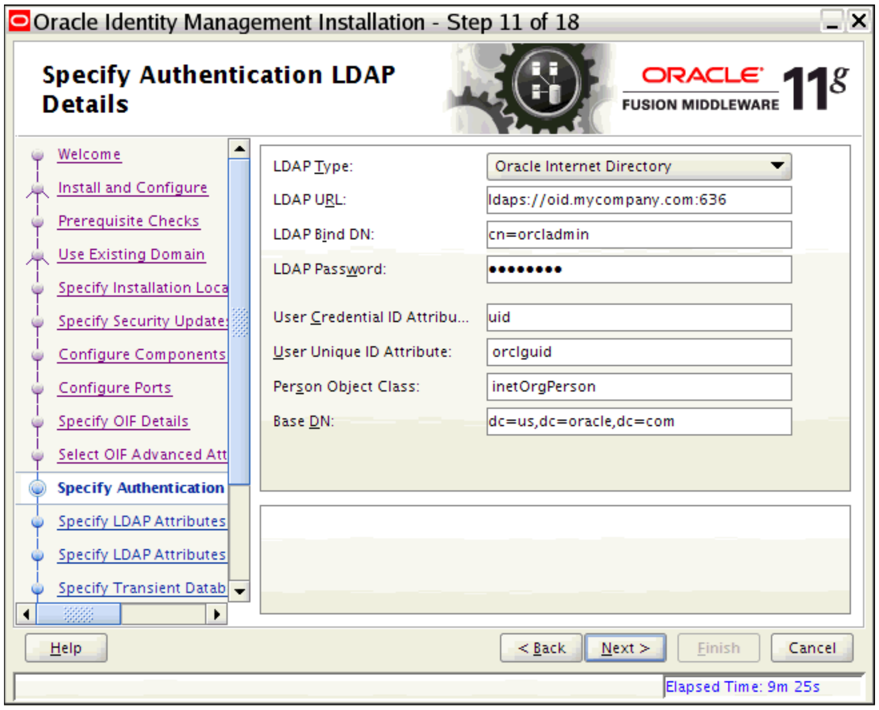
    - **LDAP Type**: Select `Oracle Internet Directory` from the drop down.

- **LDAP URL**: Provide the LDAP URL to connect to your LDAP store in the following format: `ldap://host:port` or `ldaps://host:port`. For example:

  ```
  ldaps://oid.mycompany.com:636
  ```

- **LDAP Bind DN**: `cn=orcladmin`
- **LDAP Password**: `<password for orcladmin>`
- **User Credential ID Attribute**: `uid`
- **User Unique ID Attribute**: `orclguid`
- **Person Object Class**: `inetOrgPerson`

Click **Next**.


Oracle Identity Management Installation - Step 11 of 18

Specify Authentication LDAP Details

LDAP Type:	Oracle Internet Directory
LDAP URL:	ldaps://oid.mycompany.com:636
LDAP Bind DN:	cn=orcladmin
LDAP Password:	••••••••
User Credential ID Attribu...	uid
User Unique ID Attribute:	orclguid
Person Object Class:	inetOrgPerson
Base DN:	dc=us,dc=oracle,dc=com

Elapsed Time: 9m 25s

19. On the LDAP Attributes for User Data Store screen, specify the values shown in the example below:

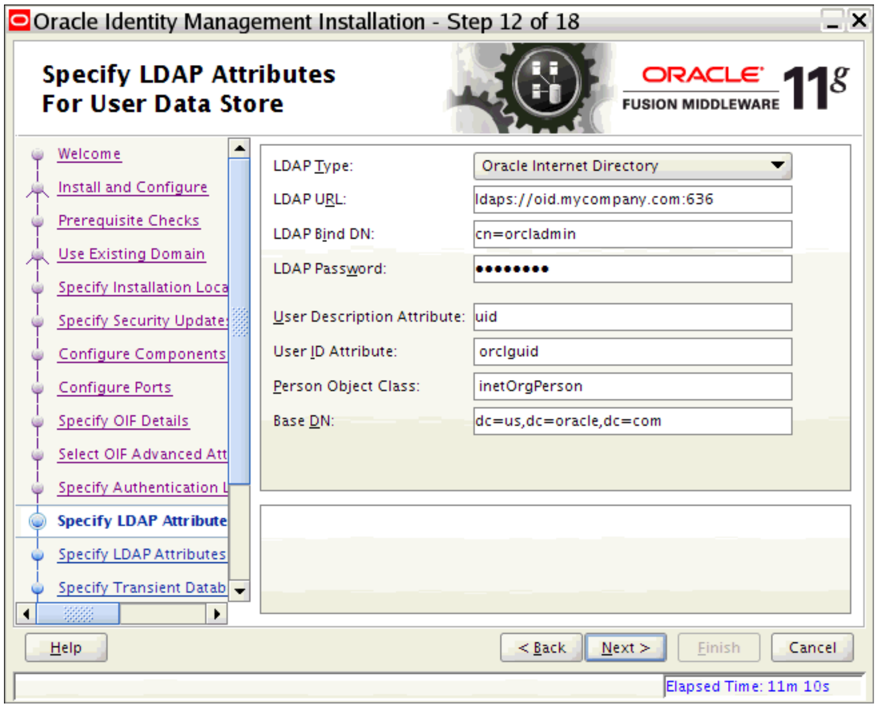
    - **LDAP Type**: Select Oracle Internet Directory from the drop down.
    - **LDAP URL**: Provide the LDAP URL to connect to your LDAP store in the following format: `ldap://host:port` or `ldaps://host:port`. For example:

      ```
      ldaps://oid.mycompany.com:636
      ```

    - **LDAP Bind DN**: `cn=orcladmin`
    - **LDAP Password**: `<password for orcladmin>`
    - **User Description Attribute**: `uid`
    - **User ID Attribute**: `orclguid`
    - **Person Object Class**: `inetOrgPerson`

- **Base DN**: `dc=us,dc=mycompany,dc=com`

Click **Next**.



LDAP Type: Oracle Internet Directory
LDAP URL: ldaps://oid.mycompany.com:636
LDAP Bind DN: cn=orcladmin
LDAP Password: ••••••••
User Description Attribute: uid
User ID Attribute: orclguid
Person Object Class: inetOrgPerson
Base DN: dc=us,dc=oracle,dc=com

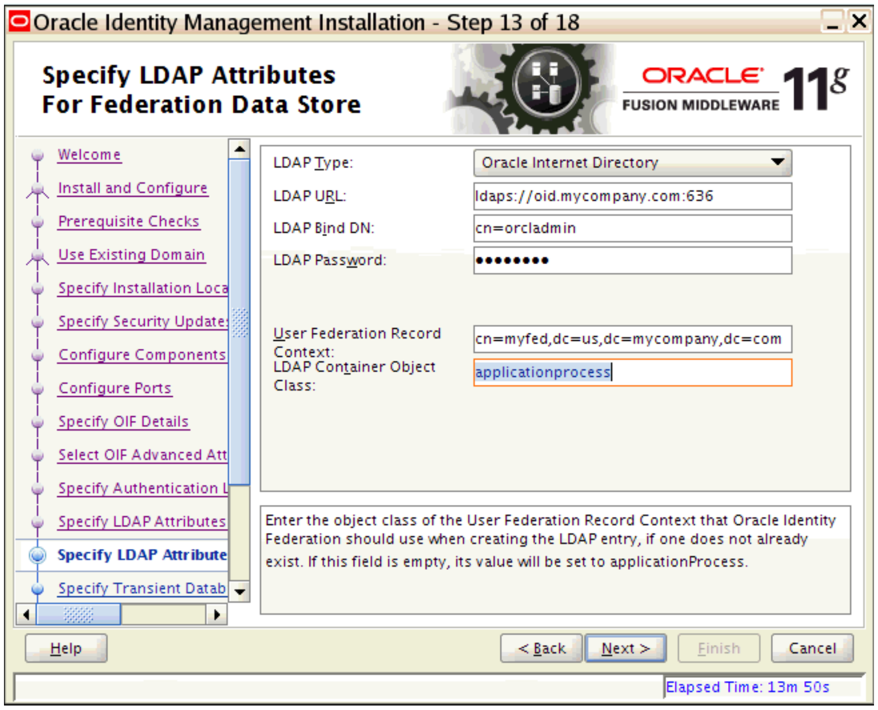
20. On the LDAP Attributes for Federation Data Store screen, specify the values shown in the example below:

    - **LDAP Type**: Select Oracle Internet Directory from the drop down.
    - **LDAP URL**: Provide the LDAP URL to connect to your LDAP store in the following format: `ldap://host:port` or `ldaps://host:port`. For example:

      ```
      ldaps://oid.mycompany.com:636
      ```

    - **LDAP Bind DN**: cn=orcladmin
    - **LDAP Password**: <password for orcladmin>
    - **User Federation Record Context**: cn=myfed,dc=us,dc=mycompany,dc=com
    - **Container Object Class**: This is the type of User Federation Record Context that Oracle Identity Federation should use when creating the LDAP container, if it does not exist already. If that field is empty, its value will be set to `applicationprocess`. For Microsoft Active Directory this field has to be set to `container`.

    Click **Next**.



LDAP Type: Oracle Internet Directory
LDAP URL: ldaps://oid.mycompany.com:636
LDAP Bind DN: cn=orcladmin
LDAP Password: ••••••••
User Federation Record Context: cn=myfed,dc=us,dc=mycompany,dc=com
LDAP Container Object Class: applicationprocess

Enter the object class of the User Federation Record Context that Oracle Identity Federation should use when creating the LDAP entry, if one does not already exist. If this field is empty, its value will be set to applicationProcess.

21. On the Transient Store Database Details screen, specify the values shown in the example below:

   - **Connect String**: Provide the connect string to your database. For example:

     ```
     infradbhost1-vip.mycompany.com:1521:idmdb1^infradbhost2-vip.mycompany.com:
     1521:idmdb2@idmedg.mycompany.com
     ```

     Note: The RAC database connect string information needs to be provided in the format *host1:port1:instance1^host2:port2:instance2@servicename*.

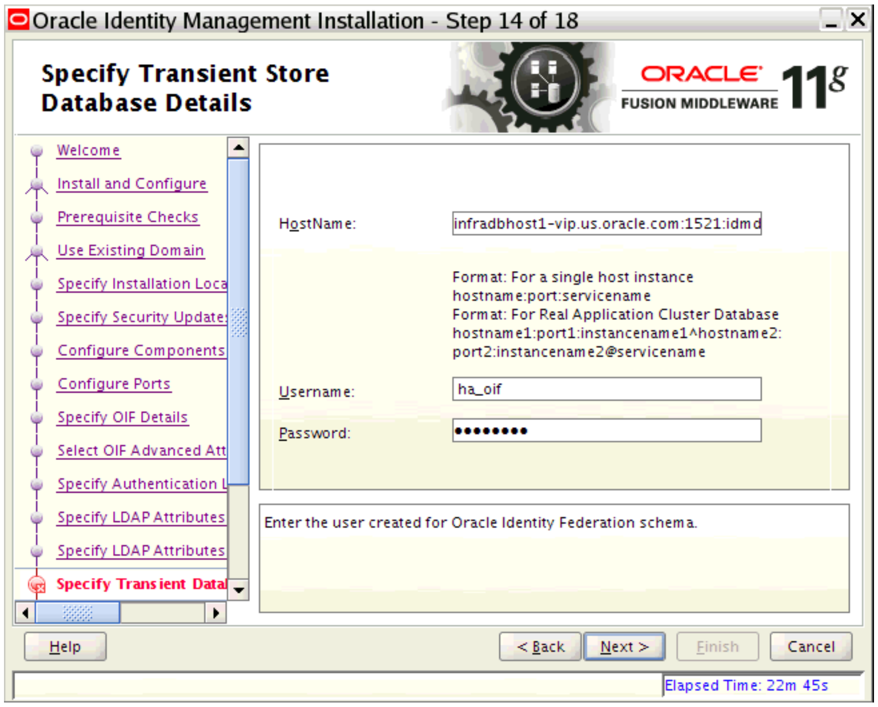
     During this installation, it is not required for all the RAC instances to be up. If one RAC instance is up, the installation can proceed.

     It is required that the information provided above is complete and accurate. Specifically, the correct host, port, and instance name must be provided for each RAC instance, and the service name provided must be configured for all the specified RAC instances.

     Any incorrect information entered in the RAC database connect string has to be corrected manually after the installation.

   - **UserName**: Enter the username for the OIF Schema. For example: `ha_oif`
   - **Password**: *`password for the oif user`*

   Click **Next**.



HostName: infradbhost1-vip.us.oracle.com:1521:idmd

Format: For a single host instance
hostname:port:servicename
Format: For Real Application Cluster Database
hostname1:port1:instancename1^hostname2:
port2:instancename2@servicename

Username: ha_oif

Password: ••••••••

Enter the user created for Oracle Identity Federation schema.

22. On the Installation Summary screen, review the selections to ensure that they are correct. If they are not correct, click **Back** to modify selections on previous screens. Then click **Install**.

23. On the Installation Progress screen, view the progress of the installation.

    Once the installation is done, the oracleRoot.sh confirmation dialog box displays. This dialog box advises you that a configuration script needs to be run as root before installation can proceed. Leaving this dialog box open, open another shell window, log in as root, and run the following script:

    ```
    /u01/app/oracle/product/fmw/oif/oracleRoot.sh
    ```

24. On the Configuration Progress screen, view the progress of the configuration.

25. On the Installation Complete screen, click **Finish** to confirm your choice to exit.

7.7.3.3 Creating boot.properties for the Administration Server on OIFHOST1

This section describes how to create a `boot.properties` file for the Administration Server on OIFHOST1. The `boot.properties` file enables the Administration Server to start without prompting for the `administrator` username and password.

Follow these steps to create the `boot.properties` file:

1. On OIFHOST1, go the following directory:

   ```
   MW_HOME/user_projects/domains/domainName/servers/AdminServer/security
   ```

   For example:

   ```
   cd /u01/app/oracle/product/fmw/user_
   projects/domains/IDMDomain/servers/AdminServer/security
   ```

2. Use a text editor to create a file called `boot.properties` under the `security` directory. Enter the following lines in the file:

```
username=adminUser
password=adminUserPassword
```

Note: When you start the Administration Server, the username and password entries in the file get encrypted.

For security reasons, minimize the time the entries in the file are left unencrypted. After you edit the file, you should start the server as soon as possible so that the entries get encrypted.

---

3. Stop the Administration Server if it is running.

   See the "Starting and Stopping Oracle Fusion Middleware" chapter of the *Oracle Fusion Middleware Administrator's Guide* for information on starting and stopping WebLogic Servers.

4. Start the Administration Server on OIFHOST1 using the `startWebLogic.sh` script located under the `*MW_HOME*/user_projects/domains/*domainName*/bin` directory.

5. Validate that the changes were successful by opening a web browser and accessing the following pages:

   - WebLogic Server Administration Console at:

     ```
     http://oidhost1.mycompany.com:7001/console
     ```

   - Oracle Enterprise Manager Fusion Middleware Control at:

     ```
     http://oidhost1.mycompany.com:7001/em
     ```

   Log into these consoles using the `weblogic` user credentials.

7.7.3.4 Installing Oracle Identity Federation on OIFHOST2

Follow the steps below to install and configure the second instance of Oracle Identity Federation on OIFHOST2.

1. Ensure that the system, patch, kernel and other requirements are met. These are listed in the *Oracle Fusion Middleware Installation Guide for Oracle Identity Management* in the Oracle Fusion Middleware documentation library for the platform and version you are using.

2. On OIFHOST1, port 7499 was used for Oracle Identity Federation. The same port should be used for the Oracle Identity Federation instance on OIFHOST2. Therefore, ensure that port 7499 is not in use by any service on OIFHOST2 by issuing these commands for the operating system you are using. If a port is not in use, no output is returned from the command.

   On UNIX:

   ```
   netstat -an | grep "7499"
   ```

   On Windows:

   ```
   netstat -an | findstr :7499
   ```

3. If the port is in use (if the command returns output identifying the port), you must free the port.

   On UNIX:

Remove the entry for ports 7499 in the `/etc/services` file and restart the services, or restart the computer.

On Windows:

Stop the component that is using the port.

4. Copy the `staticports.ini` file from the `Disk1/stage/Response` directory to a temporary directory.
5. Edit the `staticports.ini` file that you copied to the temporary directory to assign the following custom port (uncomment the line where you specify the port number for Oracle Identity Federation):

```
# The OIF Server Port
OIF Server Port = 7499
```

6. Start the Oracle Identity Management 11*g* Installer as follows:

   On UNIX, issue this command: `runinstaller`

   On Windows, double-click `setup.exe`.

   The `runInstaller` and `setup.exe` files are in the `../install/`*`platform`* directory, where *platform* is a platform such as Linux or Win32.

   This displays the Specify Oracle Inventory screen.
7. On the Specify Inventory Directory screen, enter values for the Oracle Inventory Directory and the Operating System Group Name. For example:

   Specify the Inventory Directory: `/u01/app/oraInventory`

   Operating System Group Name: `oinstall`

   A dialog box appears with the following message:

   "Certain actions need to be performed with root privileges before the install can continue. Please execute the script /u01/app/oraInventory/createCentralInventory.sh now from another window and then press "Ok" to continue the install. If you do not have the root privileges and wish to continue the install select the "Continue installation with local inventory" option"

   Log in as root and run the "/u01/app/oraInventory/createCentralInventory.sh"

   This sets the required permissions for the Oracle Inventory Directory and then brings up the Welcome screen.

Note: The Oracle Inventory screen is not shown if an Oracle product was previously installed on the host. If the Oracle Inventory screen is not displayed for this installation, make sure to check and see:

1. If the `/etc/oraInst.loc` file exists
2. If the file exists, the Inventory directory listed is valid
3. The user performing the installation has write permissions for the Inventory directory

8. On the Welcome screen, click **Next**.
9. On the Select Installation Type screen, select **Install and Configure** and then click **Next**.

10. On the Prerequisite Checks screen, the installer completes the prerequisite check. If any checks fail, please fix them and restart your installation.

    Click **Next**.

11. On the Select Domain screen, select the **Expand Cluster** option and specify these values:

    HostName: `OIFHOST1.MYCOMPANY.COM`

    Port: `7001`

    UserName: `weblogic`

    User Password: `<password for weblogic user>`

    Click **Next**.

12. On the Specify Installation Location screen, specify the following values:

    - **Oracle Middleware Home Location**:

      `/u01/app/oracle/product/fmw`

    - **Oracle Home Directory**: `oif`
    - **WebLogic Server Directory**:

      `/u01/app/oracle/product/fmw/wlserver_10.3`

    - **Oracle Instance Location**:

      `/u01/app/oracle/admin/oif_inst2`

    - **Instance Name**: `oif_inst2`

    Note: Ensure that the Oracle Home Location directory path for OIFHOST1 is the same as the Oracle Home Location path for OIFHOST2. For example, if the Oracle Home Location directory path for OIFHOST1 is: `/u01/app/oracle/product/fmw/oif`, then the Oracle Home Location directory path for OIFHOST2 must also be `/u01/app/oracle/product/fmw/oif`.

    Click **Next**.

13. On the Specify Oracle Configuration Manager Details screen, specify the values shown in the example below:

    - **Email Address**: Provide the email address for your My Oracle Support account.
    - **Oracle Support Password**: Provide the password for your My Oracle Support account.
    - Check the checkbox next to the **I wish to receive security updates via My Oracle Support** field.

    Click **Next**.

14. On the Configure Components screen, de-select all the components except **Oracle Identity Federation components**. The Oracle Identity Federation components include Oracle Identity Federation and Oracle HTTP Server.

    Click **Next**.

15. On the Installation Summary screen, review the selections to ensure that they are correct. If they are not correct, click **Back** to modify selections on previous screens. Then click **Install**.
16. On the Installation Progress screen, view the progress of the installation.

    Once the installation is done, the oracleRoot.sh confirmation dialog box displays. This dialog box advises you that a configuration script needs to be run as root before installation can proceed. Leaving this dialog box open, open another shell window, log in as root, and run the following script:

    ```
    /u01/app/oracle/product/fmw/oif/oracleRoot.sh
    ```
17. On the Configuration Progress screen, view the progress of the configuration.
18. On the Installation Complete screen, click **Finish** to confirm your choice to exit.

7.7.3.5 Post-Installation Steps for Oracle Identity Federation

In the previous section, the installer created a second Managed Server, wls_oif2 on OIFHOST2. However, the Oracle Identity Federation application is not deployed on OIFHOST2 and the newly created Managed Server is not automatically started. Also, the WebLogic Administration Console shows the state of the wls_oif2 Managed Server on OIFHOST2 as UNKNOWN.

Follow the post-installation steps in this section to complete the installation and configuration of the Oracle Identity Federation application on OIFHOST2.

7.7.3.5.1 Copy the OIF Directory from OIFHOST1 to OIFHOST2 The Oracle Identity Federation application is deployed on OIFHOST1 as an externally staged application. The application must be copied from OIFHOST1 to OIFHOST2:

Copy the OIF directory on OIFHOST1:

```
MW_HOME/user_projects/domains/OIFDomain/servers/wls_oif1/stage/OIF/11.1.1.1.0/OIF
```

to the following location on OIFHOST2:

```
MW_HOME/user_projects/domains/OIFDomain/servers/wls_oif2/stage/OIF/11.1.1.1.0/
```

For example, from OIFHOST1, execute this command:

```
scp -rp MW_HOME/user_projects/domains/OIFDomain/servers/wls_oif1/stage/OIF/
11.1.1.1.0/OIF user@OIFHOST2://MW_HOME/user_projects/domains/OIFDomain/servers/
wls_oif2/stage/OIF/11.1.1.1.0
```

7.7.3.5.2 Start the Managed Server on OIFHOST2 in a Cluster Follow these steps to start the newly created wls_oif2 Managed Server in a cluster on OIFHOST2:

1. In the left pane of the Oracle WebLogic Server Administration Console, expand **Environment** and select **Clusters**.

   See the "Starting and Stopping Oracle Fusion Middleware" chapter of the *Oracle Fusion Middleware Administrator's Guide* for information on starting and stopping WebLogic Servers.
2. Click on the link for the cluster (`cluster_oif`) containing the Managed Server (`wls_oif2`) you want to stop.
3. Select **Control**.

4. Under **Managed Server Instances in this Cluster**, select the check box next to the Managed Server (`wls_oif2`) you want to start and click **Start**.
5. On the Cluster Life Cycle Assistant page, click **Yes** to confirm.

WebLogic Node Manager starts the server on the target machine. When the Node Manager finishes its start sequence, the server's state is indicated in the **State** column in the Server Status table.

7.7.3.5.3 Configure Oracle HTTP Server

Oracle HTTP Server is installed on OIFHOST1 and OIFHOST2 along with the Oracle Identity Federation server. Configure the Oracle HTTP Server by following these steps:

1. On OIFHOST1, edit the `oif.conf` file located under the *`INSTANCE_HOME`*`/config/OHS/`*`ohsName`*`/moduleconf` directory.
2. If the Identity Management installation is in standalone mode, uncomment and set the WebLogicHost and WebLogicPort variables to reference the WebLogic Server Managed Server where Oracle Identity Federation is running (for example: `oifhost1.mycompany.com` and `7499`).
3. If the Identity Management installation is in clustered mode, uncomment and set the WebLogicCluster variable to reference the WebLogic Server Managed Servers where Oracle Identity Federation is running (for example: `oifhost1.mycompany.com:7499,oifhost2.mycompany.com:7499`).
4. Save and exit the `oif.conf` file.
5. Restart Oracle HTTP Server.

7.7.3.6 Configuring the Load Balancer

Oracle Identity Federation uses an external load balancer to balance requests between the Oracle Identity Federation nodes deployed in a high availability configuration.

7.7.3.6.1 Load Balancer Virtual Server Name Setup

When Oracle Identity Federation is deployed in a high availability configuration, it is recommended to use an external load balancer to front-end the Oracle Identity Federation server instances and load balance the HTTP requests between the various Oracle Identity Federation instances.

Refer to Section 7.2.4.5, "Configuring Virtual Server Names and Ports for the Load Balancer" for details.

7.7.3.6.2 Oracle Identity Federation Configuration

In a high availability environment, it is recommended to enable sticky sessions in the load balancer and disable HTTP Session replication. HTTP session replication replicates session information across nodes and is memory intensive and can cause slower performance.

HTTP session replication is not enabled by default and should be turned off in a high availability environment. Follow the steps below to turn off HTTP session replication on all the Managed Servers running the Oracle Identity Federation application:

1. Open the following file in a text editor:

   ```
   DOMAIN_
   HOME/servers/serverName/stage/OIF/11.1.1.1.0/OIF/web.war/WEB-INF/weblogic.xml
   ```

2. Change the value of the `persistent-store-type` parameter from `replicated_if_clustered` to `memory`.
3. Save the `weblogic.xml` file.

4. Restart the WebLogic Server Managed Server.
5. Repeat the previous steps for each Managed Server where Oracle Identity Federation is deployed.
6. In the Oracle Enterprise Manager Fusion Middleware Control, navigate to **Administration > Server Properties** and change the host name and port to reflect the load balancer host and port.
7. In the Oracle Enterprise Manager Fusion Middleware Control, navigate to **Administration > Identity Provider** and change URL to http://*LoadBalancerHost*:*LoadBalancerPort*.
8. In the Oracle Enterprise Manager Fusion Middleware Control, navigate to **Administration > Service Provider** and change URL to http://*LoadBalancerHost*:*LoadBalancerPort*.
9. Repeat these steps for each Managed Server where Oracle Identity Federation is deployed.

7.7.3.7 Validating Oracle Identity Federation High Availability

This section describes how to validate Oracle Identity Federation in a high availability configuration.

1. In a web browser, you will be able to access the following URLs if the configuration is correct:

```
http://<LoadBalancerHost>:<LoadBalancerPort>/fed/sp/metadata

http://<LoadBalancerHost>:<LoadBalancerPort>/fed/idp/metadata
```

2. Follow the instructions in the "Obtain Server Metadata" and "Add Trusted Providers" sections of *Oracle Fusion Middleware Administrator's Guide for Oracle Identity Federation* to import metadata from the SP into the IdP and the IDP metadata into the SP.
3. Go to the following URL and do a Single Sign-On operation:

```
http://<SP_Host>:<SP_port>/fed/user/testspsso
```

7.7.3.8 Enabling Oracle Identity Federation Integration with Highly Available LDAP Servers

By default, Oracle Identity Federation is not configured to be integrated with LDAP Servers deployed in a high availability configuration. To integrate Oracle Identity Federation with highly available LDAP Servers to serve as user data store, federation data store, or authentication engine, Oracle Identity Federation needs to be configured based on the LDAP server's function.

Enter the WLST script environment for Oracle Identity Federation, then set the following properties as needed:

- To integrate the user data store with a highly available LDAP Server, set the `userldaphaenabled` boolean property from the datastore group to `true`; otherwise set it to `false`:

```
setConfigProperty('datastore','userldaphaenabled', 'true', 'boolean')
```

- To integrate the federation data store with a highly available LDAP Server, set the `fedldaphaenabled` boolean property from the datastore group to `true`; otherwise set it to `false`:

```
setConfigProperty('datastore', 'fedldaphaenabled','true', 'boolean')
```

- To integrate the LDAP authentication engine with a highly available LDAP Server, set the `ldaphaenabled` boolean property from the authnengines group to `true`; otherwise set it to `false`:

```
setConfigProperty('authnengines','ldaphaenabled', 'true', 'boolean')
```

7.7.4 Oracle Identity Federation Failover and Expected Behavior

This section describes steps for performing various failover operations on Oracle Identity Federation instances deployed in a high availability environment and their expected behavior. Follow the steps in this section to perform:

- Oracle Identity Federation instance failover
- Oracle Real Application Clusters failover

7.7.4.1 Performing an Oracle Identity Federation Failover

Follow these steps to perform a a test of a failover of an Oracle Identity Federation instance and to check the status of Oracle Identity Federation:

Note: The testspsso URL referred to in the steps below is the Test SP SSO service that is bundled with Oracle Identity Federation 11g.

The testing service enabled by default, but can be disabled by the administrator. In a production environment, the Test SP SSO Service may be disabled.

if the Test SP SSO Service is disabled, you can use whatever service you have integrated to start the Federation SSO Flow from the SP.

1. Set up Oracle Identity Federation to be able to perform a federation single sign-on operation.
2. Start Single Sign-On operation from Oracle Identity Federation, acting as a Service Provider. One possible way to do this is to use the `http://<SPhost>:<SPport>/fed/user/testspsso` URL choosing Artifact profile.
3. On the IdP login page, shut down wls_oif1 through the Managed Server page and enter the username and password.
4. The Single Sign-On operation should succeed.

7.7.4.2 Performing a RAC Failover

Follow these steps to perform a RAC failover:

1. On one of the database hosts (infradbhost1-vip) where the Oracle Identity Federation schema is installed, use the `srvctl` command to stop a database instance:

   ```
   srvctl stop instance -d db_unique_name -i inst_name_list
   ```

2. Use the `srvctl` command to check the status of the database:

   ```
   srvctl status database -d db_unique_name -v
   ```

3. Perform an operation on Oracle Identity Federation:

   ```
   ORACLE_INSTANCE/bin/opmnctl status ias-component=oif1
   ```

4. Use the `srvctl` command to start the database instance:

   ```
   srvctl start instance -d db_unique_name -i inst_name_list
   ```

7.7.5 Troubleshooting Oracle Identity Federation High Availability

The following information may help you troubleshoot Oracle Identity Federation issues:

- Oracle Identity Federation logs its messages in the Oracle WebLogic Server Managed Server log files, for example:

  ```
  DOMAIN_HOME/servers/wls_oif1/logs/wls_oif1-diagnostic.log
  ```

  It is recommended that you use the Oracle Enterprise Manager Fusion Control to view log files by choosing **Identity And Access > OIF > Logs > View Log Messages**.

- Make sure that the database that you use for Oracle Identity Federation does not have old configuration data. If so, newer data may be overwritten. Delete old data in the Oracle Identity Federation configuration database tables or recreate the schema before you point Oracle Identity Federation to the database.

- The host name and port in the Oracle Identity Federation server configuration should be set to the load balancer host and port - otherwise there will be errors during Single Sign-On operation.

- If the clocks on the computers on which the IDP and SP are running have different times, you will see errors during Single Sign-On. Fix this by setting the system clocks to have the same time or by adjusting the server drift using the Server Properties page in the Oracle Enterprise Manager Fusion Middleware Control.

- The ProviderId is a string that uniquely identifies the IDP/SP. The ProviderId for all servers in a cluster must be the same. The ProviderId is defaulted to: `host:port/fed/<sp|idp>` at installation time. If necessary change or set this value after installation. Do not change it again during operation.

For more information on troubleshooting Oracle Identity Federation, see *Oracle Fusion Middleware Administrator's Guide for Oracle Identity Federation*.

7.8 Collocated Architecture High Availability

This section describes how to design and deploy Oracle Internet Directory, Oracle Directory Integration Platform, Oracle Virtual Directory, and Oracle Directory Services Manager in collocated high availability environments. The introduction to these components is provided in previous sections of this manual.

This section describes the collocated architecture for Oracle Internet Directory, Oracle Directory Integration Platform, Oracle Virtual Directory and Oracle Directory Services Manager when deployed in a high availability configuration.

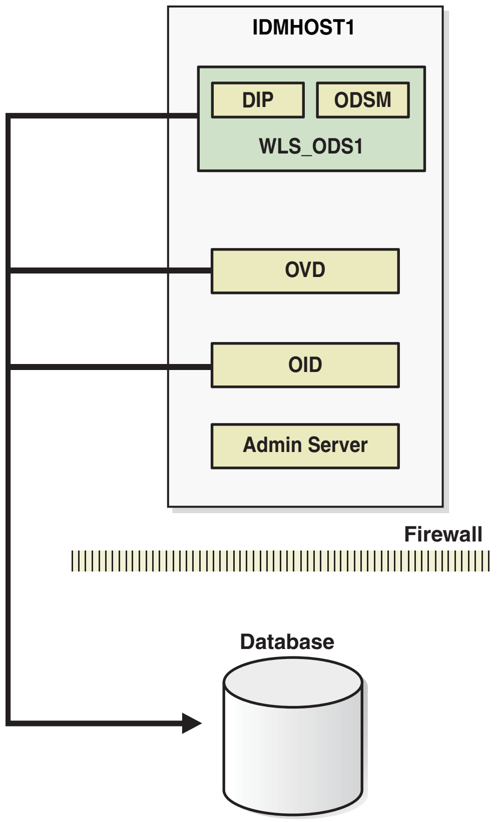
7.8.1 Collocated Architecture Overview

See the sections below for an architecture overview of each component in the collocated architectures described in this section:

- Oracle Internet Directory: Section 7.3.1, "Oracle Internet Directory Component Architecture"
- Oracle Virtual Directory: Section 7.4.1, "Oracle Virtual Directory Component Architecture"
- Oracle Directory Integration Platform: Section 7.5.1, "Oracle Directory Integration Platform Component Architecture"
- Oracle Directory Services Manager: Section 7.6.1, "Oracle Directory Services Manager Component Architecture"

Figure 7–12 shows Oracle Internet Directory, Oracle Directory Integration Platform, Oracle Virtual Directory, and Oracle Directory Services Manager collocated on a single host and deployed in a non-high availability architecture.

Figure 7–12 Collocated Components Architecture

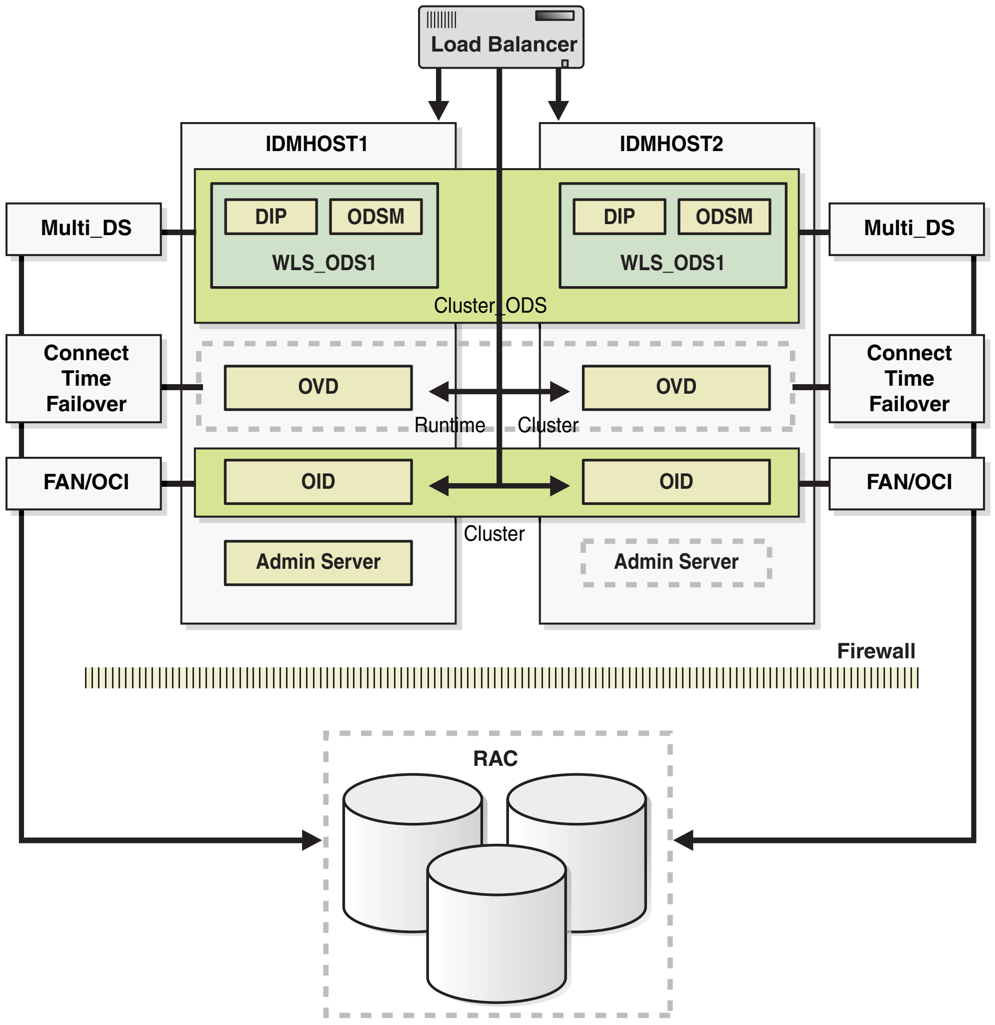


All the components in Figure 7–12 are deployed on the same host, but have separate Oracle homes and Oracle instances. Oracle Internet Directory uses a standalone Oracle database as the security metadata repository.

7.8.2 Collocated Architecture High Availability Deployment

Figure 7–13 shows Oracle Internet Directory, Oracle Virtual Directory, Oracle Directory Integration Platform, and Oracle Directory Services Manager collocated on IDMHOST1 and IDMHOST2 and deployed in a high availability architecture.

Figure 7–13 Collocated Components in a High Availability Architecture



7.8.2.1 Collocated Architecture Prerequisites

See the sections below for the prerequisites of each component in the collocated architectures described in this section:

- Oracle Internet Directory: Section 7.3.2.3, "Oracle Internet Directory Prerequisites"
- Oracle Virtual Directory: Section 7.4.2.2, "Oracle Virtual Directory Prerequisites"
- Oracle Directory Integration Platform: Section 7.5.2.3, "Oracle Directory Integration Platform Prerequisites"
- Oracle Directory Services Manager: Section 7.6.2.3, "Oracle Directory Services Manager Prerequisites"
- Oracle Identity Federation: Section 7.7.2.3, "Oracle Identity Federation Prerequisites"

7.8.2.2 Configuring Collocated Components for High Availability

This section provides the steps to install and configure Oracle Internet Directory, Oracle Directory Integration Platform, Oracle Virtual Directory and Oracle Directory Services Manager on IDMHOST1 and IDMHOST2 in a high availability architecture:

Note: In a collocated environment, the Oracle Identity Management components should be installed in separate Oracle Homes. They can share the same MW_HOME.

For each component, make sure that the Oracle Home Location directory path for the first instance is the same as the Oracle Home Location directory path for the second instance.

For example, if the Oracle Home Location directory path for the first Oracle Internet Directory instance on OIDHOST1 is `/u01/app/oracle/product/fmw/idm`, then the Oracle Home Location directory path for the second Oracle Internet Directory instance on OIDHOST2 must also be `/u01/app/oracle/product/fmw/idm`.

---

1. Install the database. For more information, see Section 7.2.2, "Installing and Configuring the Database Repository."
2. Install RCU. For more information, see Section 7.2.3, "Installing the Repository Creation Utility Software."
3. Configure the database. For more information, see Section 7.2.4, "Configuring the Database for Oracle Fusion Middleware 11g Metadata."
4. Run RCU to install the required schemas for Oracle Internet Directory and Oracle Identity Federation. For more information, see Section 7.3.2.3.2, "Using RCU to Create Oracle Internet Directory Schemas in the Repository" and Section 7.7.2.3.1, "Using RCU to Create Oracle Identity Federation Schemas in the Repository."
5. Install and configure Oracle Internet Directory on the first host. For more information, see Section 7.3.3.1.1, "Installing Oracle Internet Directory on OIDHOST1" or Section 7.3.3.2.2, "Installing Oracle Internet Directory on OIDHOST1."
6. Install and configure Oracle Internet Directory on the second host. For more information, see Section 7.3.3.1.3, "Installing Oracle Internet Directory on OIDHOST2" or Section 7.3.3.2.4, "Installing Oracle Internet Directory on OIDHOST2."
7. Install and configure Oracle Virtual Directory on the first host. For more information, see Section 7.4.3.1.1, "Installing Oracle Virtual Directory on OVDHOST1" or Section 7.4.3.2.2, "Installing the First Oracle Virtual Directory."
8. Install and configure Oracle Virtual Directory on the second host. For more information, see Section 7.4.3.1.2, "Installing Oracle Virtual Directory on OVDHOST2" or Section 7.4.3.2.4, "Installing an Additional Oracle Virtual Directory."
9. Install and configure Oracle Directory Integration Platform and Oracle Directory Services Manager on the first host. For more information, see Section 7.5.3.2, "Installing and Configuring Oracle Directory Integration Platform and Oracle Directory Services Manager on IDMHOST1."
10. Install and configure Oracle Directory Integration Platform and Oracle Directory Services Manager on the second host. For more information, see Section 7.5.3.4, "Installing and Configuring Oracle Directory Integration Platform and Oracle Directory Services Manager on IDMHOST2."

7.8.3 Validating the Collocated Components High Availability

See the following sections for information about validating components in the collocated high availability architectures and for information about how to failover the components and RAC.

7.8.3.1 Validation Tests

See the sections below for information on validating the following components in the collocated high availability architectures:

- Oracle Internet Directory: Section 7.3.4, "Validating Oracle Internet Directory High Availability"
- Oracle Virtual Directory: Section 7.4.4, "Validating Oracle Virtual Directory High Availability"
- Oracle Directory Integration Platform: Section 7.6.4, "Validating Oracle Directory Services Manager High Availability"
- Oracle Directory Services Manager: Section 7.6.4, "Validating Oracle Directory Services Manager High Availability"
- Oracle Identity Federation: Section 7.7.3.7, "Validating Oracle Identity Federation High Availability"

7.8.3.2 Failures and Expected Behaviors

See the sections below for information on failures and expected behaviors for the following components in the collocated high availability architectures:

- Oracle Internet Directory: Section 7.3.5, "Oracle Internet Directory Failover and Expected Behavior"
- Oracle Virtual Directory: Section 7.4.5, "Oracle Virtual Directory Failover and Expected Behavior"
- Oracle Directory Integration Platform: Section 7.5.4, "Oracle Directory Integration Platform Failover and Expected Behavior"
- Oracle Directory Services Manager: Section 7.6.5, "Oracle Directory Services Manager Failover and Expected Behavior"
- Oracle Identity Federation: Section 7.7.4, "Oracle Identity Federation Failover and Expected Behavior"

7.8.4 Troubleshooting Collocated Components Manager High Availability

See the sections below for information on troubleshooting the following components in the collocated high availability architectures:

- Oracle Internet Directory: Section 7.3.6, "Troubleshooting Oracle Internet Directory High Availability"
- Oracle Virtual Directory: Section 7.4.6, "Troubleshooting Oracle Virtual Directory High Availability"
- Oracle Directory Integration Platform: Section 7.5.5, "Troubleshooting Oracle Directory Integration Platform High Availability"
- Oracle Directory Services Manager: Section 7.6.6, "Troubleshooting Oracle Directory Services Manager"

- Oracle Identity Federation: Section 7.7.5, "Troubleshooting Oracle Identity Federation High Availability"

7.8.5 Additional Considerations for Collocated Components High Availability

See the sections below for information on additional considerations for the following components in the collocated high availability architectures:

- Oracle Internet Directory: Section 7.3.7, "Additional Oracle Internet Directory High Availability Issues"
- Oracle Directory Services Manager: Section 7.6.7, "Additional Considerations for Oracle Directory Services Manager High Availability"

8 Configuring Identity Management for Maximum High Availability

This chapter provides high-level instructions for setting up a maximum high availability deployment for Oracle Identity Management. This deployment includes two sites in different geographic locations. This is an active-active deployment where both sites are active at the same time when the deployment is functioning normally. If one site fails, the surviving site continues to function.

Each site includes a two-node Oracle Internet Directory cluster configuration, which provides high availability for Oracle Internet Directory. The Oracle Internet Directory cluster configuration at each site uses a RAC database as the security store, which provides high availability for the database. Chapter 7, "Configuring High Availability for Identity Management Components" provides an introduction to the high availability Oracle Internet Directory cluster configurations.

Multimaster replication is used to replicate data from the master site to the replica site.

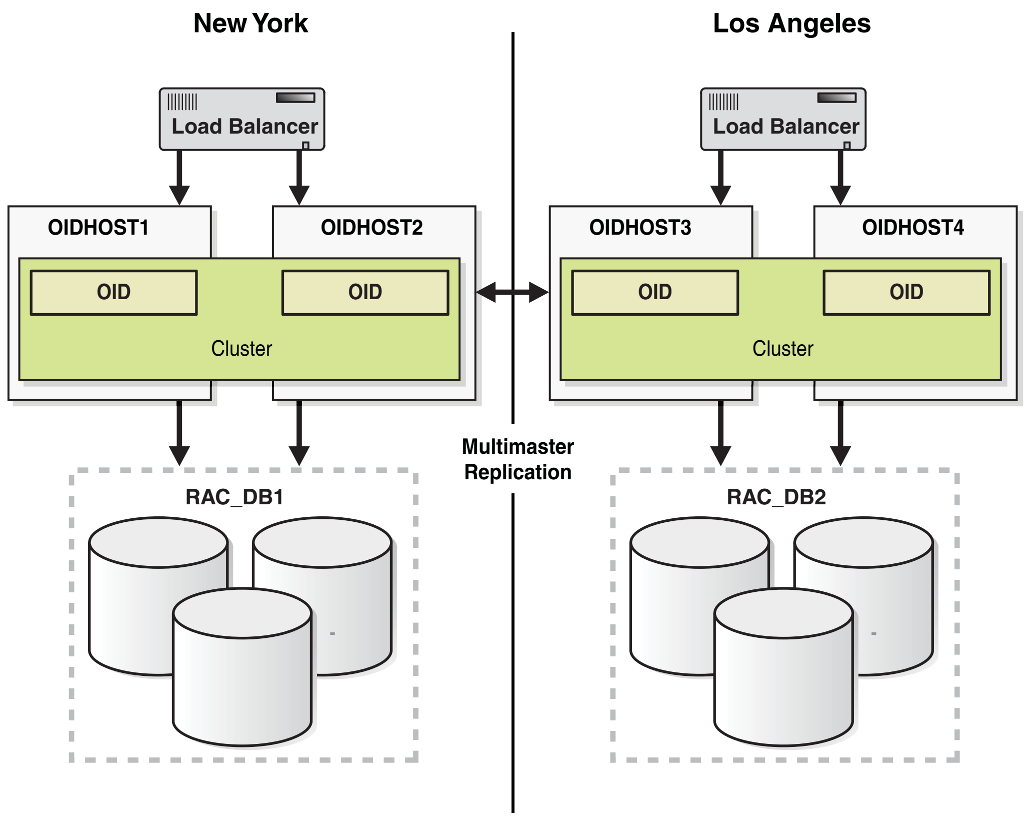
This chapter includes the following topics:

- Section 8.1, "Introduction to the Maximum High Availability Identity Management Deployment"
- Section 8.2, "Overview of Replication"
- Section 8.3, "Setting up Multimaster Replication"

8.1 Introduction to the Maximum High Availability Identity Management Deployment

Figure 8–1 shows the maximum high availability deployment for Oracle Identity Management.

Figure 8–1 Maximum High Availability Multimaster Replication Deployment



The master site is located in New York and the replica site is located in Los Angeles.

Each site includes a highly available two-node Oracle Internet Directory cluster configuration that uses a RAC database as a highly available security store. Each two-node cluster has a load balancer. See Section 7.3.3, "Oracle Internet Directory High Availability Configuration Steps" for information on setting up a two-node Oracle Internet Directory cluster.

The master site in New York consists of:

- OIDHOST1 and OIDHOST2

  These are the two clustered hosts on which Oracle Internet Directory is installed.

- RAC_DB1

  This is the RAC database which serves as the security store for the Oracle Internet Directory instances on OIDHOST1 and OIDHOST2. Multimaster replication is used to replicate data between RAC_DB1 in New York and RAC_DB2 in Los Angeles.

The replica site in Los Angeles consists of:

- OIDHOST3 and OIDHOST4

  These are the two clustered hosts on which Oracle Internet Directory is installed.

- RAC_DB2

  This is the RAC database which serves as the security store for the Oracle Internet Directory instances on OIDHOST3 and OIDHOST4. Multimaster replication is used to replicate data between RAC_DB1 in New York and RAC_DB2 in Los Angeles.

8.2 Overview of Replication

The following types of replication are available for Oracle Internet Directory:

- LDAP multimaster replication

  Uses the industry-standard Lightweight Directory Access Protocol Version 3 as the replication transport mechanism. This is the recommended protocol to use for replication.

- Oracle Advanced Database multimaster replication

  Uses the replication feature of Oracle Database. This is also called Advanced Replication.

- Two-way fan-out replication

  With this replication method, the replicated data is read/write at both the master site and replica site. Fan-out uses LDAP as its transport mechanism.

- One-way fan-out replication

  With this replication method, the replicated data is read-only at the replica site. Fan-out uses LDAP as its transport mechanism.

For more information about the replication types for Oracle Internet Directory, refer to *Oracle Fusion Middleware Administrator's Guide for Oracle Internet Directory*.

For the maximum availability deployment shown in Figure 8–1, either LDAP or Oracle Advanced Database multimaster replication can be set up.

8.3 Setting up Multimaster Replication

This section describes how to set up LDAP multimaster replication or Oracle Advanced Database multimaster replication for the maximum high availability Oracle Internet Directory deployment shown in Figure 8–1.

Note: See Section 7.3.3, "Oracle Internet Directory High Availability Configuration Steps" for information on installing the Oracle Internet Directory two-node clusters for the New York and Los Angeles multimaster topology shown in Figure 8–1.

It is recommended that you use LDAP multimaster replication for the maximum availability Oracle Internet Directory deployment.

Note: New Oracle Fusion Middleware 11*g* customers who want to install and configure 10.1.4.3 or later Oracle Single Sign-On and Oracle Delegated Administration Services against 11*g* Oracle Internet Directory and to set up multimaster replication should refer to these steps:

1. To configure 10.1.4.3 Oracle Single Sign-On and Oracle Delegated Administration Services to run against 11*g* Oracle Internet Directory, follow the steps in the "Installing Oracle Single Sign-On and Oracle Delegated Administration Services against Oracle Internet Directory" section of *Oracle Fusion Middleware Installation Guide for Oracle Identity Management*.
2. Perform the steps in the following sections of the "Deploying Identity Management with Multimaster Replication" chapter in the *Oracle Fusion Middleware High Availability Guide* for release 10.1.4.0.1 (part number B28186-01) to install and configure 10.1.4.3 Oracle Single Sign-On and Oracle Delegated Administration Services for multimaster replication:
   - Section 10.1.4 "Installing OracleAS Single Sign-On and Oracle Delegated Administration Services on the Master Node"
   - Section 10.1.5 "Synchronizing the OracleAS Single Sign-On Schema Password"
   - Section 10.1.6 "Installing OracleAS Single Sign-On and Oracle Delegated Administration Services on the Replica Node"
   - Section 10.1.7 "If You Are Running in SSL Mode"

8.3.1 Setting Up LDAP Multimaster Replication

Follow these steps in the Oracle Enterprise Manager Fusion Middleware Control to set up LDAP multimaster replication:

1. From the Oracle Internet Directory menu on the Oracle Internet Directory instance home page, choose **Administration**, and then **Replication Management**.
2. You are prompted to log into the replication DN account. Provide the host, port of one of the Oracle Internet Directory servers at the master site (the New York cluster in Figure 8–1), replication DN, and replication DN password. If anonymous binds are enabled on this Oracle Internet Directory component, the replication DN field will fill in automatically when you enter the host and port.
3. Click the Create icon.
4. On the Type screen, select the replication type: **Multimaster Replication**.
5. Click **Next**. The Replicas screen displays the replication type you selected.
6. Provide an agreement name. The agreement name must be unique across all the nodes.
7. For multimaster replication, enter the host, port, user name (replication DN), and replication password for the primary node and all the secondary nodes.

Note: Enter the host/port of any of the Oracle Internet Directory instances in the cluster.

---

8. Click **Next** to go the Settings page.
9. In the **LDAP Connection** field, select **Keep Alive** if you want the replication server to use same connection for performing multiple LDAP operations. Select **Always Use New Connection** if you want the server to open a new connection for each LDAP operation.
10. Enter the **Replication Frequency**.
11. Enter the **Human Intervention Queue Schedule**. This is the interval, in minutes, at which the directory replication server repeats the change application process.
12. The Replication Server Start Details section has options to start the replication server and enable bootstrap. Choose **Start Server** to start the appropriate server instance. You can also enable bootstrap by choosing **Enable Bootstrap**. You must select the **Instance Name** and **Component Name** from the dropdown lists to start the server.
13. Click **Next** to go to the Scope page. The default primary naming context will be filled in. Keep the default settings.
14. Click **Next**. The Summary page displays a summary of the replication agreement you are about to create. To make any changes to information on the Summary page, click **Back**.
15. Click **Finish** to create the replication agreement.

For detailed instructions on setting up LDAP multimaster replication, read the Oracle Enterprise Manager tool tips or refer to *Oracle Fusion Middleware Administrator's Guide for Oracle Internet Directory*.

8.3.1.1 Adding a Node in LDAP Multimaster Replication

Follow these steps in the Oracle Enterprise Manager Fusion Middleware Control to add a node in an LDAP multimaster replication deployment:

1. From the Oracle Internet Directory menu on the Oracle Internet Directory instance home page, select **Administration**, and then **Replication Management**.
2. You will be prompted to log into the replication DN account. Provide the host, port and replication DN password of any of the replicas in the multimaster replication deployment.
3. In the upper half of the screen, click on the appropriate multimaster replication agreement row to enable editing.
4. Click **Edit** on the Replication Agreements page.
5. In the lower half of the screen, click the Replicas tab.
6. To add a new replica to the multimaster replication deployment, click the **Create** icon.
7. In the popup window, provide the host, port and replication DN password details for the new node. Click **Add**.
8. Click **Apply**. The replica will be added to the existing multimaster directory replication group (DRG).

For more detailed information on adding a node in an LDAP multimaster replication deployment, see *Oracle Fusion Middleware Administrator's Guide for Oracle Internet Directory*.

8.3.1.2 Deleting a Node in LDAP Multimaster Replication

Follow these steps in the Oracle Enterprise Manager Fusion Middleware Control to delete a node in an LDAP multimaster replication deployment:

1. From the Oracle Internet Directory menu on the Oracle Internet Directory instance home page, select **Administration**, and then **Replication Management**.
2. You will be prompted to log into the replication DN account. Provide the host, port and replication DN password of any of the replicas in the multimaster replication deployment.
3. In the upper half of the screen, click on the appropriate multimaster replication agreement row to enable editing.
4. Click **Edit** on Replication Agreement screen.
5. In the lower half of the screen, click the Replicas tab.
6. Click the replica you want to delete from the multimaster replication deployment. The **Delete** icon becomes enabled.
7. Click the **Delete** icon.
8. Click the **Apply** button to remove the replica from the LDAP multimaster directory replication group (DRG).

For more detailed information on deleting a node in an LDAP multimaster replication deployment, see *Oracle Fusion Middleware Administrator's Guide for Oracle Internet Directory*.

8.3.2 Setting Up Oracle Advanced Database Multimaster Replication

The detailed steps for setting up Oracle Advanced Database multimaster replication are available in the "Installing and Setting Up an Oracle Database Advanced Replication-Based Multimaster Replication Group" section in *Oracle Fusion Middleware Administrator's Guide for Oracle Internet Directory*.

Table 8–1 shows the subsections of the "Installing and Setting Up an Oracle Database Advanced Replication-Based Multimaster Replication Group" section in *Oracle Fusion Middleware Administrator's Guide for Oracle Internet Directory*. It also describes the instructions to perform in each subsection to set up Oracle Advanced Database multimaster replication for the maximum high availability Oracle Internet Directory deployment shown in Figure 8–1.

Table 8–1 Steps for Setting Up Oracle Database Advanced Multimaster Replication

Subsection	Instructions
Task 1: Install Oracle Internet Directory on the Master Definition Site (MDS)	This task should already have been performed based on the Note in Section 8.3, "Setting up Multimaster Replication."
Task 2: Install the Oracle Internet Directory on the Remote Master Sites (RMS)	This task should already have been performed based on the Note in Section 8.3, "Setting up Multimaster Replication."
If an Existing Master is Used as a Remote Master Site	Set up one site as the master definition site (MDS), for example, the New York site. Then set up one Oracle Internet Directory node in the cluster at the master site (for example, OIDHOST1) to be the master host. OIDHOST1 will be the host in the master site cluster where the replication server will be configured and will run. When the setup steps require a reference to a replication server, process, or port for the MDS, specify the correct value for OIDHOST1. Set up another site as the remote master site (RMS), for example, the Los Angeles site. Then set up one Oracle Internet Directory node in the cluster at the remote site (for example, OIDHOST3) to be the replica host. The replica host is referred to as the "new node" in the "If an Existing Master is Used as a Remote Master Site" section). OIDHOST3 will be the host in the Los Angeles cluster where the replication server will be configured and will run. When the setup steps require a reference to a replication server, process, or port for the RMS, specify the correct value for OIDHOST3.
Task 3: Set Up Advanced Replication for a Directory Replication Group	Follow the instructions for this task.
On All Nodes, Prepare the Oracle Net Services Environment for Replication	Perform the steps in this subsection in all the database Oracle homes and in all the Oracle Internet Directory Oracle homes in the New York site and the Los Angeles site.
From the MDS, Configure Advanced Replication For Directory Replication	Perform the steps in this subsection in all of the Oracle Internet Directory Oracle homes in New York and Los Angeles, with one exception: When you configure Advanced Replication using the Replication Environment Management Tool, execute the command on only the master host at the MDS site (for example, on OIDHOST1 in New York). The replication must be started on only one Oracle Internet Directory host.
Task 4 (Optional): Load Data into the Directory	If you choose to use the bulkload utility, stop all the Oracle Internet Directory instances in all the Oracle Internet Directory homes, and use only one of the Oracle Internet Directory instances to perform the bulkload operation.
Task 5: Ensure that Oracle Directory Server Instances are Started on All the Nodes	Perform this task in all the Oracle Internet Directory Oracle home directories.
Task 6: Start the Replication Servers on All Nodes in the DRG	Start the replication server on only OIDHOST1 in New York and OIDHOST3 in Los Angeles.

8.3.2.1 Adding a Node in Oracle Advanced Database Multimaster Replication

The detailed steps for adding a node in an Oracle Advanced Database multimaster replication deployment are in the "Adding a Node for Oracle Database Advanced Replication-Based Multimaster Replication Group" section in *Oracle Fusion Middleware Administrator's Guide for Oracle Internet Directory*.

Table 8–2 shows the subsections of the "Adding a Node for Oracle Database Advanced Replication-Based Multimaster Replication Group" section in *Oracle Fusion Middleware Administrator's Guide for Oracle Internet Directory*. It also describes the instructions to perform in each subsection to add a node to the maximum high availability Oracle Internet Directory deployment shown in Figure 8–1.

Table 8–2 Steps for Adding a Node in Oracle Advanced Database Replication

Subsection	Instructions
Prepare the Oracle Net Services Environment	Perform the steps in this subsection in all the database Oracle homes and in all the Oracle Internet Directory Oracle homes in the master definition site (New York site) and the remote master definition site (Los Angeles site). Also, perform these steps in the database Oracle homes and in all the Oracle Internet Directory Oracle homes for the new cluster that is being added.
Task 1: Stop the Directory Replication Server on All Nodes	Stop the replication server on OIDHOST1 in New York and on OIDHOST3 in Los Angeles.
Task 2: Identify a Sponsor Node and Install Oracle Internet Directory on the Remote Site	OIDHOST1 in the New York cluster will be the sponsor node and the MDS.
Task 3: Switch the Sponsor Node to Read-Only Mode	Perform this task in the Oracle home for Oracle Internet Directory on OIDHOST1 and OIDHOST2.
Task 4: Back up the Sponsor Node by Using ldifwrite	Perform this task in the Oracle home for Oracle Internet Directory on OIDHOST1.
Task 5: Perform Advanced Replication Add Node Setup	Perform this task in the Oracle home for Oracle Internet Directory on OIDHOST1.
Task 6: Switch the Sponsor Node to Updatable Mode	Perform this task in the Oracle home for Oracle Internet Directory on OIDHOST1 and OIDHOST2.
Task 7: Start the Directory Replication Server on All Nodes Except the New Node	Perform this task on OIDHOST1 in the New York cluster and on OIDHOST3 in the Los Angeles cluster.
Task 8: Load Data into the New Node by Using bulkload	Perform this task on one of the Oracle Internet Directory Oracle homes in the new cluster that is being added. Stop all the Oracle Internet Directory processes on the new node before using bulkload.
Task 9: Start the Directory Server on the New Node	Perform this task on all of the Oracle Internet Directory nodes in the new cluster that is being added.
Task 10: Start the Directory Replication Server on the New Node	Perform this task on one of the Oracle Internet Directory Oracle homes in the new cluster that is being added.

8.3.2.2 Deleting a Node in Oracle Advanced Database Multimaster Replication

The detailed steps for deleting a node in an Oracle Advanced Database multimaster replication deployment are in the "Deleting a Node from a Multimaster Replication Group" section in *Oracle Fusion Middleware Administrator's Guide for Oracle Internet Directory*.

Table 8–3 shows the subsections of the "Deleting a Node from a Multimaster Replication Group" section in *Oracle Fusion Middleware Administrator's Guide for Oracle Internet Directory*. It also describes the instructions to perform in each subsection to delete a node in the maximum high availability Oracle Internet Directory deployment shown in Figure 8–1. In the instructions below, the MDS is assumed to be the New York site.

Table 8–3 Steps for Deleting a Node in Oracle Advanced Database Replication

Subsection	**Instructions**
Task 1: Stop the Directory Replication Server on All Nodes	Perform this task on each node in the Directory Replication Group (DRG).
Task 2: Stop All Oracle Internet Directory Processes in the Node to be Deleted	Perform this task in the node to be deleted.
Task 3: Delete the Node from the Master Definition Site	Perform this task in the Oracle home for Oracle Internet Directory on OIDHOST1.
Task 4: Start the Directory Replication Server on All Nodes	Perform this task on all the remaining nodes in the DRG.

9 Configuring High Availability for Web Tier Components

The Web tier receives each incoming HTTP request and invokes the requested business logic operation in the application. Based on the results of the operation and state of the model, the next view is selected to display. The selected view is transmitted to the client for presentation.

This chapter describes high availability concepts and configuration procedures for Oracle HTTP Server and Oracle Web Cache. This chapter includes the following topics:

- Chapter 9.1, "About the Web Tier"
- Chapter 9.2, "Oracle HTTP Server and High Availability Concepts"
- Chapter 9.3, "Oracle Web Cache and High Availability Concepts"

9.1 About the Web Tier

The Web tier of Java EE application server is responsible for interacting with the end user such as web browsers primarily in the forms of HTTP requests and responses. It is the outermost tier in the application server, closest to the end user. At the highest level, the Web tier does four basic tasks:

- Interprets client requests
- Dispatches those requests to an object (for example, an enterprise Java bean) that encapsulates business logic
- Selects the next view for display
- Generates and delivers the next view

The Web tier receives each incoming HTTP request and invokes the requested business logic operation in the application. Based on the results of the operation and state of the model, the next view is selected to display. The selected view is transmitted to the client for presentation.

Oracle Web Cache is a content-aware server accelerator, or reverse proxy, for the Web tier that improves the performance, scalability, and availability of Web sites that run on Oracle HTTP Server.

Oracle Web Cache is the primary caching mechanism provided with Oracle Fusion Middleware. Caching improves the performance, scalability, and availability of Web sites that run on Oracle Application Server by storing frequently accessed URLs in memory.

By storing frequently accessed URLs in memory, Oracle Web Cache eliminates the need to repeatedly process requests for those URLs on the application Web server and database tiers. Unlike legacy proxies that handle only static objects, Oracle Web Cache caches both static and dynamically generated content from one or more application Web servers. Because Oracle Web Cache is able to cache more content than legacy proxies, it provides optimal performance by greatly reducing the load on application Web server and database tiers. As an external cache, Oracle Web Cache is also an order of magnitude faster than object caches that run within the application tier.

Because Web Cache is fully compliant with HTTP 1.0 and 1.1 specifications, it can accelerate Web sites that are hosted by any standard web servers, such as Apache and IIS. In Oracle Fusion Middleware, Oracle Web Cache resides in front of one or more instances of Oracle HTTP Server. Responses to browser based HTTP requests are directed to the Oracle HTTP Server instance and transmitted through Oracle Web Cache. The Oracle Web Cache instance can handle any Web content transmitted with standard HTTP protocol.

9.2 Oracle HTTP Server and High Availability Concepts

Oracle HTTP Server is the Web server component for Oracle Fusion Middleware. It provides a listener for Oracle WebLogic Server and the framework for hosting static pages, dynamic pages, and applications over the Web.

9.2.1 Oracle HTTP Server Single-Instance Characteristics

Oracle HTTP Server is based on Apache 2.2.10 infrastructure, and includes modules developed specifically by Oracle. The features of single sign-on, clustered deployment, and high availability enhance the operation of the Oracle HTTP Server. Oracle HTTP Server has the following components to handle client requests:

- HTTP listener, to handle incoming requests and route them to the appropriate processing utility.
- Modules (mods), to implement and extend the basic functionality of Oracle HTTP Server. Many of the standard Apache modules are included with Oracle HTTP Server. Oracle also includes several modules that are specific to Oracle Fusion Middleware to support integration between Oracle HTTP Server and other Oracle Fusion Middleware components.
- Perl interpreter, a persistent Perl runtime environment embedded in Oracle HTTP Server through mod_perl.

Oracle HTTP Server enables developers to program their site in a variety of languages and technologies, such as the following:

- Perl (through mod_perl and CGI)
- C (through CGI and FastCGI)
- C++ (through FastCGI)
- PHP (through mod_php)
- Oracle PL/SQL

Oracle HTTP Server can also be a proxy server, both forward and reverse. A reverse proxy enables content served by different servers to appear as if coming from one server.

The Oracle HTTP Server is essentially a stateless application. Session State can be maintained using browser-based cookies.

Figure 9–1 Oracle HTTP Server Architecture

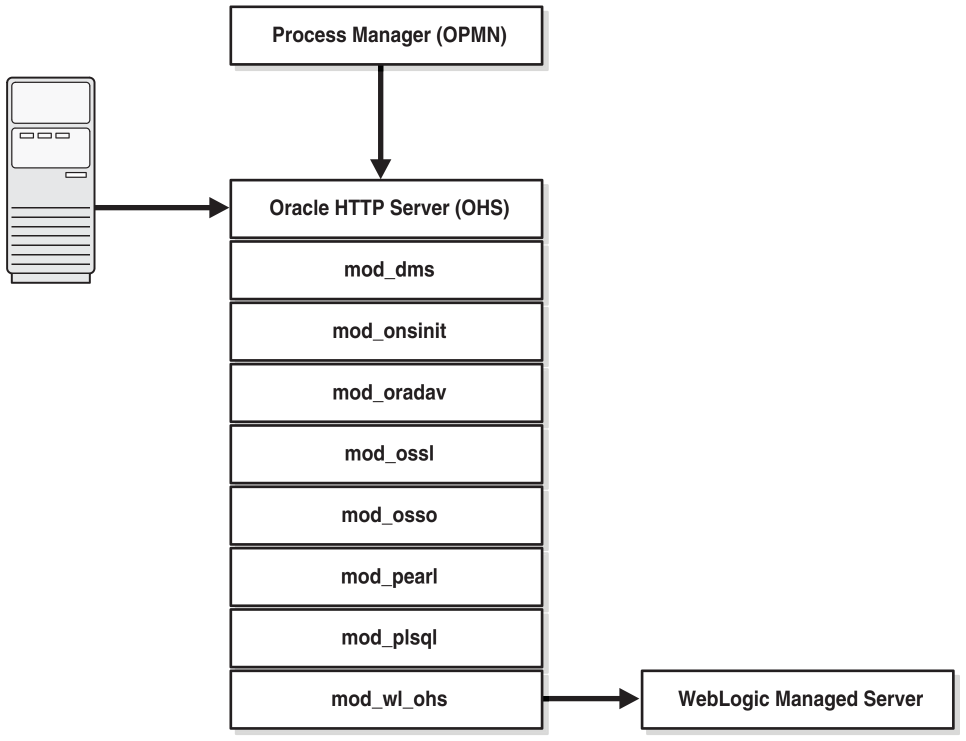


Figure 9–1 illustrates the following Oracle HTTP Server components:

- The Oracle Process Manager and Notification Service (OPMN) is used to start, stop, and monitor Oracle HTTP Server. OPMN periodically polls Oracle HTTP server to ensure that it is still functioning. If Oracle HTTP Server is not functioning, OPMN takes appropriate action to restart the failed component to ensure that service is maintained.
- Oracle HTTP server is based on the industry leading Apache Web Server. The Oracle HTTP Server's core functionality can be extended using the following *modules*:
  - The **mod_dms** module enables performance monitoring of site components using Oracle Dynamic Monitoring Service (DMS).
  - The **mod_onsint** module provides integration support with Oracle Notification Service (ONS) and OPMN.
  - The **mod_oradav** module is an Oracle Call Interface (OCI) application written in C, that extends the implementation of **mod_dav**. The mod_oradav directive can read and write to local files, or to an Oracle database.
  - The **mod_ossl** module enables strong cryptography for Oracle HTTP Server. This Oracle module is a plug-in to Oracle HTTP Server that enables the server to use SSL.
  - The **mod_osso** module enables Oracle Single Sign-on.
  - The **mod_perl** module embeds the Perl interpreter into Oracle HTTP Server. This eliminates start-up overhead and enables Perl applications to be accessed through Oracle HTTP Server.
  - The **mod_plsql** module connects Oracle HTTP Server to an Oracle database, enabling Web application creation using Oracle stored procedures.

– The **mod_wl_ohs** module allows requests to be proxied from Oracle HTTP Server to Oracle WebLogic Server.

9.2.1.1 Oracle HTTP Server and Oracle WebLogic Server

Oracle HTTP Server does not require an Oracle WebLogic domain. However, Oracle HTTP Server is typically used in conjunction with an Oracle WebLogic Domain.

The link to Oracle WebLogic managed servers is handled through the module mod_wl_ohs. This module is configured by routing requests of a particular type, for example, JSPs, or by routing requests destined to a URL, to specific WebLogic Managed Servers. In a non high availability deployment, this destination is defined by specifying the host name and port on which the WebLogic manage server resides.

In a high availability deployment, the WebLogic managed servers are typically clustered together. In such a deployment, a special mod_wl_ohs directive, WebLogicCluster, is used to specify a comma-separated list of cluster members. This list is not necessarily a complete list of cluster members. When a request requiring a WebLogic managed server is received, mod_wl_ohs send that request to one of the WebLogic cluster members listed in the directive. When the WebLogic managed server receives the request, it not only processes it, but it also sends a complete list of cluster members back to mod_wl_ohs. When mod_wl_ohs receives this updated list, it dynamically adds any previously unknown servers to the list of known servers, allowing all future requests to be load balanced across the full cluster member list. This process has the advantage of allowing new managed servers to be added to the cluster without updating mod_wl_ohs, or adding the Oracle HTTP Server.

In an example scenario, assume that the WebLogic Cluster consists of WLS1, WLS2, and WLS3. Also assume that WLS1 and WLS2 are known to mod_wl_ohs.

When mod_wl_ohs first receives a request, it attempts to send that request to either WLS1 or WLS2, if it is successful then the list is updated to include WLS3 for future requests. If, however, WLS1 and WLS2 are unavailable, but WLS3 is available, the request is rejected because mod_wl_ohs has no way of knowing WLS3 exists.

When using the Oracle HTTP server with an Oracle WebLogic domain, Oracle recommends associating Oracle HTTP Server with the WebLogic domain. This allows the Oracle HTTP Server to be incorporated into the Oracle Fusion Middleware Console for centralized management and monitoring.

For more information Oracle WebLogic clusters, see *Oracle Fusion Middleware Using Clusters for Oracle WebLogic Server*.

9.2.1.2 Oracle HTTP Server External Dependencies

Oracle HTTP Server is dependent on the Oracle Process Management and Notification Server to start the HTTP server itself.

In order to access other resources, such as Oracle WebLogic, or Oracle Single Sign-On, Oracle HTTP Server is also dependent on the appropriate loading of Apache module.

9.2.2 Oracle HTTP Server Startup and Shutdown Lifecycle

The Oracle HTTP Server process is invoked indirectly through OPMN. When OPMN receives a request to start the Oracle HTTP Server, it starts the Oracle HTTP Server process (httpd).

After Oracle HTTP Server is started, it is ready to listen for and respond to HTTP(S) requests. The request processing model on Microsoft Windows systems differs from that on UNIX systems.

- For Microsoft Windows, there is a single parent process and a single child process. The child process creates threads that are responsible for handling client requests. The number of created threads is static and can be configured for performance.
- For UNIX, there is a single parent process that manages multiple child processes. The child processes are responsible for handling requests. The parent process brings up additional child processes as necessary, based on configuration. Although the server has the ability to dynamically bring up additional child processes, it is best to configure the server to start enough child processes initially so that requests can be handled without a need for more child processes.

9.2.3 Starting and Stopping Oracle HTTP Server

You can use Fusion Middleware Control or the `opmnctl` command to start and stop Oracle HTTP Server.

Note: Do not use the `apachectl` utility to manage Oracle HTTP Server.

You can determine the status of Oracle HTTP Server using `opmnctl`:

```
opmnctl status

Processes in Instance: instance1
---------------------------------+--------------------+---------+---------
ias-component                    | process-type       |     pid | status
---------------------------------+--------------------+---------+---------
webcache1                        | WebCache-admin     |   19556 | Alive
webcache1                        | WebCache           |   19555 | Alive
ohs1                             | OHS                |    7249 | Alive
```

9.2.3.1 Understanding the PID File

When Oracle HTTP Server starts up, it writes the process ID (PID) of the parent `httpd` process to the `httpd.pid` file located, by default, in the following directory:

```
ORACLE_INSTANCE/diagnostics/logs/OHSComponent/ohs1/
```

The process ID can be used by the administrator when restarting and terminating the daemon. If a process stops abnormally, it is necessary to stop the httpd child processes using the kill command.

The `PidFile` directive in `httpd.conf` specifies the location of the PID file.

Note: For UNIX platforms, if you edit the `PidFile` directive, you also have to edit the `ORACLE_HOME/ohs/bin/apachectl` file to specify the new location of the PID file.

See Also: PidFile directive in the Apache Server documentation at:

`http://httpd.apache.org/docs/`

9.2.3.2 Starting and Stopping Oracle HTTP Server Using Oracle Fusion Middleware Control

To start Oracle HTTP Server using Fusion Middleware Control:

1. Navigate to the Oracle HTTP Server home page.
2. Select **Control** from the Oracle HTTP Server menu.
3. Select **Start Up** from the Control menu.

To stop Oracle HTTP Server using Fusion Middleware Control:

1. Navigate to the Oracle HTTP Server home page.
2. Select **Control** from the Oracle HTTP Server menu.
3. Select **Shut Down** from the Control menu.

9.2.3.3 Starting and Stopping Oracle HTTP Server Using opmnctl

To start all Oracle HTTP Server components in an Oracle instance using `opmnctl`:

```
opmnctl startproc process-type=OHS
```

To start a specific Oracle HTTP Server component, such as `ohs1`, using `opmnctl`:

```
opmnctl startproc ias-component=ohs1
```

To stop all Oracle HTTP Server components in an Oracle instance using `opmnctl`:

```
opmnctl stopproc process-type=OHS
```

To stop a specific Oracle HTTP Server component, such as `ohs1`, using `opmnctl`:

```
opmnctl stopproc ias-component=ohs1
```

9.2.4 Oracle HTTP Server Configuration Artifacts

Oracle HTTP Server is configured using several operating system files. When Oracle HTTP Server is deployed, its configuration information is placed into the following directory structure:

- *ORACLE_INSTANC*E/config/OHS/*OHS_NAME*/
- *ORACLE_INSTANCE*/config/OHS/*OHS_NAME*/moduleconf

The primary configuration file is httpd.conf, located in the *ORACLE_INSTANCE*/config/OHS/*OHS_NAME*. All other configuration files are invoked through this main control file.

Note: The http.conf file has a general include directive. As a result, all files with a `.conf` extension in the directory moduleconf are automatically included in the configuration. Many of the Oracle applications, such as Oracle Portal and Oracle Forms, place their Oracle HTTP directives into files located in this directory.

9.2.5 Oracle HTTP Server Log File Locations

There are two types of log files for Oracle HTTP Server:

- Error logs, which record server problems.

- Access logs, which record which components and applications are being accessed, and by whom.

You can view Oracle Fusion Middleware log files using either Oracle Fusion Middleware Control or a text editor. The log files for Oracle HTTP Server are located in the following directory:

ORACLE_INSTANCE/diagnostics/logs/OHS/*OHS_NAME*.

The default name of a log file is *ohs_name*.log.

For more information about Oracle HTTP Server log files, see the *Oracle Fusion Middleware Administrator's Guide for Oracle HTTP Server*.

9.2.6 Oracle HTTP Server High Availability Architecture and Failover Considerations

Figure 9–2 shows two Oracle HTTP Servers, which have been placed behind a load balancer. The load balancer receives requests from users and forwards them on to the connected Oracle HTTP Servers. In the example, the Load Balancer receives the requests on the standard HTTP/HTTPS ports (80/443), however, it then passes them on to the Oracle HTTP Servers using completely different ports. This has the following advantages:

- Actual ports are hidden from users.
- Users do not have to add the port numbers to the URL.

On Unix-based systems, it is not mandatory to start Oracle HTTP Server with root privileges. Only root can start a process which uses a port less than 1024.

The load balancer routes requests to the functioning Oracle HTTP Servers.

Figure 9–2 Oracle HTTP Server High Availability Architecture

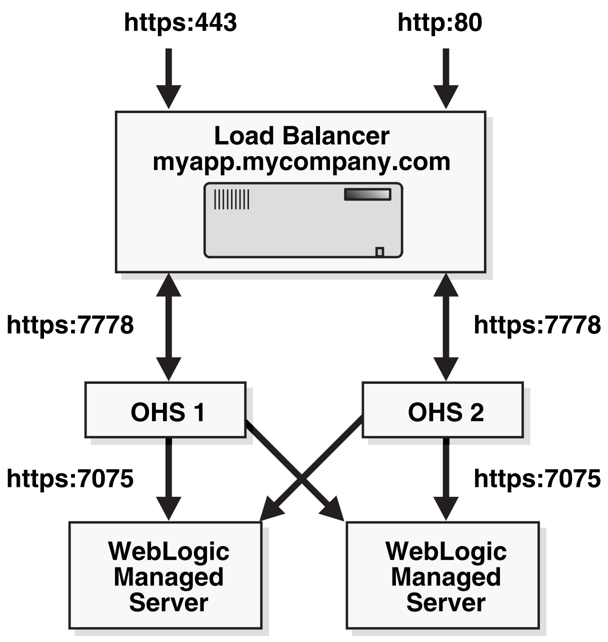


Figure 9–2 also shows how Oracle HTTP Server distribute requests to Web Logic Managed Servers. For High Availability, it is assumed that each pair of components (Oracle HTTP Server and Web Logic Managed Servers) exist on different hosts.

This architecture could simply be separated across two servers. Alternatively, in more complex implementations, each component could be located on a completely separate server.

9.2.7 Oracle HTTP Server Protection from Failures and Expected Behaviors

Oracle HTTP Servers failures can be divided into two categories: process failures and node failures. Individual operating system process may fail, where as node failures are involve the entire host upon which the Oracle HTTP Server failing.

Process Failure

The Oracle HTTP Server processes are protected by the Oracle Process Manager and Notification system (OPMN). If an Oracle HTTP Server process fails, OPMN automatically restarts the process.

Node Failure

If an entire node fails, the load balancer or Oracle Web Cache, in front of Oracle HTTP Server, sends a request to another Oracle HTTP Server if the first one does not respond, or is determined to be failed through URL pings.

WebLogic Managed Server Failure

In a high Availability deployment, Oracle WebLogic managed servers are part of a cluster. If one of the managed servers fails, mod_wl_ohs automatically redirects requests to one of the active cluster members. If the application stores state, state replication is enabled within the cluster, which allows redirected requests access to the same state information.

Database Failure

Database failures are only likely to be an issue where mod_oradav or mod_plsql is used. If this is an Oracle Real Application Clusters (RAC) database, the failure characteristics are determined by the defined RAC connection.

If client connection failover is configured, any in-flight transactions are rolled back, and a database reconnection is required.

If Transparent Application Failover (TAF) is configured, any in-flight database write is rolled back, but an automatic database reconnection takes place, and select statements are automatically recovered. In this scenario, TAF only fails over select statements, and package variables are lost.

9.2.8 Oracle HTTP Server Cluster-Wide Configuration Changes

Oracle HTTP Servers are not deployed in a clustered framework. The Oracle HTTP server configuration is file-based, so changes made to one Oracle HTTP Server must be manually copied to other Oracle HTTP Servers in the configuration. This also applies to static HTML files stored in the htdocs directory.

9.2.9 Configuring Oracle HTTP Server for High Availability

This section describes the prerequisites and procedures for configuring an example high availability deployment of Oracle HTTP Server.

9.2.9.1 Prerequisites

Consider the following prerequisites before configuring a high availability Oracle HTTP Server deployment.

9.2.9.1.1 Load Balancer In Order to distribute requests against Oracle HTTP Servers a load balancer is required. This can either be an external load balancer or Oracle Web Cache. If using an external load balancer, it should have the following features:

- Virtual server name and port configuration
- Process failure detection
- Monitoring of ports (HTTP, HTTPS) for Oracle HTTP and HTTPS
- SSL Protocol Conversion (if required)

Configuring Virtual Server Names and Ports for the Load Balancer

For each application, such as myapp.mycompany.com, configure the load balancer with a virtual server name and associated ports. In an Oracle HTTP Server installation, these virtual servers are configured for HTTP connections, which are distributed across the HTTP servers.

If your site is serving requests for HTTP and HTTPS connections, Oracle recommends that HTTPS requests are terminated at the load balancer and passed through as HTTP requests. To do this the load balancer should be able to perform the protocol conversion, and must be configured for persistent HTTP sessions.

For this example configuration, it is assumed that a the load balancer has been configured as:

- Virtual Host: Myapp.mycompany.com
- Virtual Port: 7778
- Server Pool: Map
- Server: WEBHOST1, Port 7778, WEBHOST2, Port 7778

Managing Port Numbers

Many Oracle Fusion Middleware components and services use ports. As an administrator, it is important to know the port numbers used by these services, and to ensure that the same port number is not used by two services on your host.

Most port numbers are assigned during installation. It is important that any traffic going from the Oracle HTTP Servers to the Oracle WebLogic Servers has access through any firewalls.

9.2.9.1.2 Associating Oracle HTTP Server with a WebLogic Domain If registering Oracle HTTP Server with a WebLogic Administration Server, that server must be up and running and configured to accept JMX requests.

9.2.9.2 Install Oracle HTTP Server on Webhost1

Start the Oracle Universal Installer for Oracle Fusion Middleware Web Tier and Plug-ins 11g (11.1.1.1.0) DVD installation as follows:

For UNIX, run the following command: `runInstaller`

For Windows, double-click **setup.exe**

1. In the Welcome screen, click **Next**.

2. In the Select Install Type screen, select **Install and Configure**, and click **Next**.
3. In the Specify Installation Location screen, enter the following installation location:

   /u01/app/oracle/product/FMW/Web1
4. In the Prerequisite Checks screen, click **Next**.
5. In the Configure Components screen, select **Oracle HTTP Server** and click **Next**

   Note: If this installation is to be associated with a WebLogic Domain, select the **Associate Selected Components with WebLogic Domain** check box.

   If this installation is not to be associated with a WebLogic domain deselect the **Associate Selected Components with WebLogic Domain** check box. If required, this association can be created after the installation.

6. In the Specify WebLogic Domain Details screen (optional), enter the following values:
   - Domain Host Name: The name of the server hosting the WebLogic administration server, for example, **wladmin.mycompany.com**.
   - Domain Port Number: The WebLogic Administration server Port, for example, **7001**.
   - Username: The WebLogic Administration Server user, for example **WebLogic**.
   - Password: Administration Server password
7. Click **Next**.
8. In the Specify Component Details screen, enter the following values:
   - Instance Home Location: **/u01/app/oracle/admin/web1**
   - AS Instance Name: **web1**
   - OHS Component Name: **ohs1**
   - WebCache Component Name: **webcache1**
9. Click **Next**.
10. In the Web Cache Administrator Password screen, specify the password for your Web Cache administrator.

    Valid passwords are 5 to 30 characters long, must begin with an alphabetic character, use only alphanumeric, underscore (_), dollar ($) or pound (#) characters and include at least one number.
11. Click **Next**.
12. In the Specify Web Tier Port Details screen, create a file with the following entries:
    - [OHS]#: The http_main port for the OHS component.
    - OHS Port = 7778

    Save the file and specify the name of the file in the **File name** field after highlighting **Specify Ports** using the **Configuration** file option.

Note: It is not mandatory to have the same port numbers on all Web tier instances, however, it is recommended. For this configuration example, it is assumed.

---

13. Click **Next**.
14. In the Configuration Summary screen, review the selections to ensure that they are correct. If they are not correct, click **Back** to modify selections on previous screens.
15. Click **Install**.
16. In the Configuration screen, multiple configuration assistants are launched in succession. When this process completes, click **Next**.

    The Installation Complete screen appears.
17. Click **Finish**.

Validate the Installation

Validate the installation using the following URL to access the Oracle Http Server home page:

http://webhost1:7778/

9.2.9.2.1 Configure Virtual Host(s) For each virtual host or site name used add an entry to the Oracle HTTP server configuration. Create a file called virtual_hosts.conf file located in the ORACLE_INSTANCE/config/OHS/ohs_component_name /moduleconf directory as follows:

```
NameVirtualHost *:7778
<VirtualHost *:7778>
    ServerName http://myapp.mycompany.com:80
    RewriteEngine On
    RewriteOptions inherit
    UseCanonicalName On
</VirtualHost>
```

If you are using SSL/SSL Termination (*):

```
NameVirtualHost *:7778
<VirtualHost *:7778>
    ServerName https://myapp.mycompany.com:443
    RewriteEngine On
    RewriteOptions inherit
    UseCanonicalName On
</VirtualHost>
```

(*) SSL Termination refers to when the client sends requests to the load balancer in SSL. The load balancer strips out the SSL and passes on the requests to Web Cache or Oracle HTTP Server without SSL. In order for SSL termination to function, a special module must be added to Oracle HTTP Server. The examples in this section show what to include for Unix and Windows systems. Choose the appropriate option depending on the target platform.

9.2.9.2.2 Configure mod_wl_ohs After installing and configuring Oracle HTTP Server, link it to any definedWebLogic managed servers by editing the mod_wl_ohs.conf file located in *ORACLE_INSTANCE*/config/OHS/*ohs_name* directory.

The following is an example of mod_wl_ohs.conf entries:

```
LoadModule weblogic_module ORACLE_HOME/ohs/modules/mod_wl_22.so

<IfModule mod_weblogic.c>
      WebLogicCluster apphost1.mycompany.com:7050, apphost2.mycompany.com:7050
      Debug ON
      WLLogFile /u01/app/oracle/admin/WL1/logs/mod_wl_ohs.log
      MatchExpression *.jsp
 </IfModule>

<Location /weblogic>
  SetHandler weblogic-handler
  WebLogicCluster apphost1.mycompany.com:7050,apphost2.com:7050
  Debug ERR
  WLLogFile /u01/app/oracle/admin/WL1/logs/mod_wl_ohs.log
  DefaultFileName index.jsp
</Location>
```

These examples show two different ways of routing requests to Oracle WebLogic managed servers:

- The `<ifModule>` block sends any requests ending in *.jsp to the WebLogic managed server cluster located on Apphost1 and Apphost2.
- The `<Location>` block sends any requests with URLs prefixed by /weblogic to the WebLogic managed server cluster located on Apphost1 and Apphost2.

9.2.9.2.3 Restart Oracle HTTP Server Restart the Oracle HTTP Server using the following commands:

```
opmnctl restartproc process-type=OHS
```

9.2.9.2.4 Validate the Oracle HTTP Server Configuration Validate the configuration using the following URLs:

http://myapp.mycompany.com:7778/weblogic

http://myapp.mycompany.com:7778/weblogic

9.2.9.3 Install Oracle HTTP Server on Webhost2

In WebHost2, perform the steps from Section 9.2.9.2, "Install Oracle HTTP Server on Webhost1."

Validate the Installation

Validate the installation using the following URL to access the Oracle HTTP Server home page:

http://webhost2:7778/

9.2.9.3.1 Configure Virtual Host(s) Add an entry for each virtual host or site name to the Oracle HTTP Server configuration. Copy the file virtual_hosts.conf from Webhost1 to Webhost2. This is located in the ORACLE_INSTANCE/config/OHS/ohs_component_name/moduleconf directory.

9.2.9.3.2 Configure mod_wl_ohs After installing and configuring the Oracle HTTP Server, link it to any defined WebLogic Server instances by copying the file mod_wl_ohs.conf file located in the ORACLE_INSTANCE/config/OHS/ohs_component_name directory from Webhost1.

9.2.9.3.3 Restart Oracle HTTP Server Restart the HTTP Server using the following commands:

```
opmnctl restartproc process-type=OHS
```

9.2.9.3.4 Validate the Oracle HTTP Server Configuration Validate the configuration using the following URLs:

http://myapp.mycompany.com/

https://myapp.mycompany.com (if using SSL/SSL termination)

http://myapp.mycompany.com:7778/weblogic

9.3 Oracle Web Cache and High Availability Concepts

Oracle Web Cache is a content-aware server accelerator, or *reverse proxy*, for the Web tier that improves the performance, scalability, and availability of Web sites that run on Oracle HTTP Server.

Oracle Web Cache is the primary caching mechanism provided with Oracle Fusion Middleware. Caching improves the performance, scalability, and availability of Web sites that run on Oracle Application Server by storing frequently accessed URLs in memory.

By storing frequently accessed URLs in memory, Oracle Web Cache eliminates the need to repeatedly process requests for those URLs on the application Web server and database tiers. Unlike legacy proxies that handle only static objects, Oracle Web Cache caches both static and dynamically generated content from one or more application Web servers. Because Oracle Web Cache is able to cache more content than legacy proxies, it provides optimal performance by greatly reducing the load on application Web server and database tiers. As an external cache, Oracle Web Cache is also an order of magnitude faster than object caches that run within the application tier.

9.3.1 Oracle Web Cache Single-Node Characteristics

Oracle Web Cache is fully compliant with HTTP 1.0 and 1.1 specifications. Therefore, it can accelerate Web sites that are hosted by any standard web servers, such as Apache and IIS. In Oracle Fusion Middleware, Oracle Web Cache resides in front of one or more instances of Oracle HTTP Server. Responses to browser based HTTP requests are directed to the Oracle HTTP Server instance and transmitted through Oracle Web Cache. The Oracle Web Cache instance can handle any Web content transmitted with standard HTTP protocol.

A reverse proxy appears to be the content server to clients but internally retrieves its objects from other backend origin servers as a proxy. A reverse proxy acts as a gateway to the origin servers. It relays requests from outside the firewall to origin servers behind the firewall, and delivers retrieved content back to the client.

Figure 9–3 shows an overview of how reverse proxy Web caching works. Oracle Web Cache has an IP address of `144.25.190.241` and the application Web server has an IP address of `144.25.190.242`.

The steps for browser interaction with Oracle Web Cache are as follows:

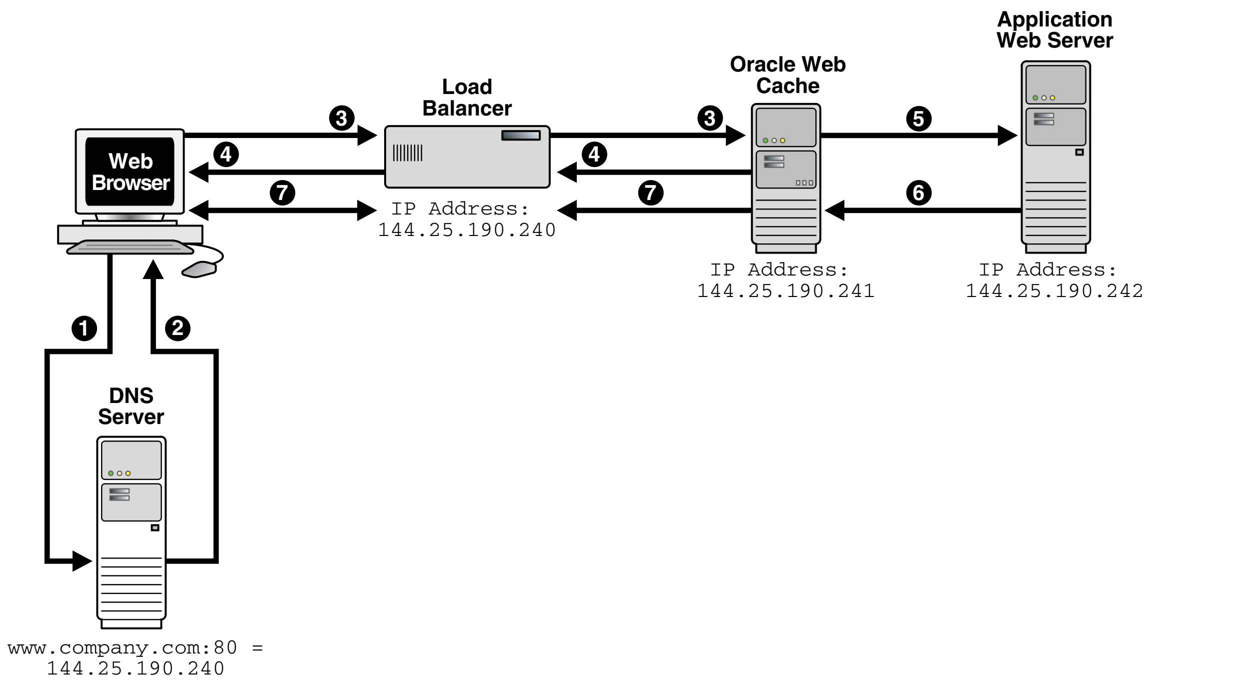
1. A browser sends a request to a Web site named `www.company.com:80`.

   This request in turn generates a request to Domain Name System (DNS) for the IP address of the Web site.

2. DNS returns the IP address of the load balancer for the site, that is, `144.25.190.240`.
3. The browser sends the request for a Web page to the load balancer. In turn, the load balancer sends the request to Oracle Web Cache server `144.25.190.241`.
4. If the requested content is in its cache, then Oracle Web Cache sends the content directly to the browser. This is called a cache hit.
5. If Oracle Web Cache does not have the requested content or the content is stale or invalid, it hands the request off to application Web server `144.25.190.242`. This is called a cache miss.
6. The application Web server sends the content to Oracle Web Cache.
7. Oracle Web Cache sends the content to the client and stores a copy of the page in cache.

   A page stored in the cache is removed when it becomes invalid or outdated.

Figure 9–3 Web Server Acceleration


Application Web Server
Oracle Web Cache
Load Balancer
Web Browser
IP Address: 144.25.190.240
IP Address: 144.25.190.241
IP Address: 144.25.190.242
DNS Server
www.company.com:80 = 144.25.190.240
1
2
3
4
5
6
7

9.3.1.1 Oracle Web Cache Component Characteristics

An Oracle Web Cache system component consists of two performance-oriented native processes, the `cache` server process and the `admin` server process. The `cache` server process handles client requests serving content back to the client. The `admin` server process provides administration, configuration, and monitoring capabilities.

9.3.1.2 Oracle Web Cache Process Monitoring

You manage the `cache` and `admin` server processes withOracle Process Manager and Notification Server (OPMN) using the Fusion Middleware Control, Oracle Web Cache Manager, or the `opmnctl` utility, as described in the *Oracle Fusion Middleware Administrator's Guide for Oracle Web Cache*.

9.3.1.3 Oracle Web Cache Startup and Shutdown Lifecycle

OPMN is responsible for the direct start, stop, restart, and monitoring of the `cache` server and `admin` server processes. Anytime the Oracle Web Cache configuration is statically modified, you must stop and restart Oracle Web Cache processes:

The executable for the `cache` process is `webcached`, and the executable for the `admin` server process is `webcachea`. These executables reside in the following directories:

```
(UNIX) ORACLE_HOME/webache/bin
(Windows) ORACLE_HOME\bin
```

When you stop Oracle Web Cache, all objects are cleared from the cache. In addition, all statistics are cleared.

After you configure Oracle Web Cache, restart Oracle Web Cache. In addition, if you change the `administrator` password, restart the `admin` server.

To restart Oracle Web Cache, use one of the following tools:

- Use Fusion Middleware Control or the `opmnctl` command-line utility to restart the `cache` or `admin` server processes.
- Use Oracle Web Cache Manager to restart the `cache` server process.

You must restart *both* the `cache` server and `admin` server processes if you modified one of the following configuration settings:

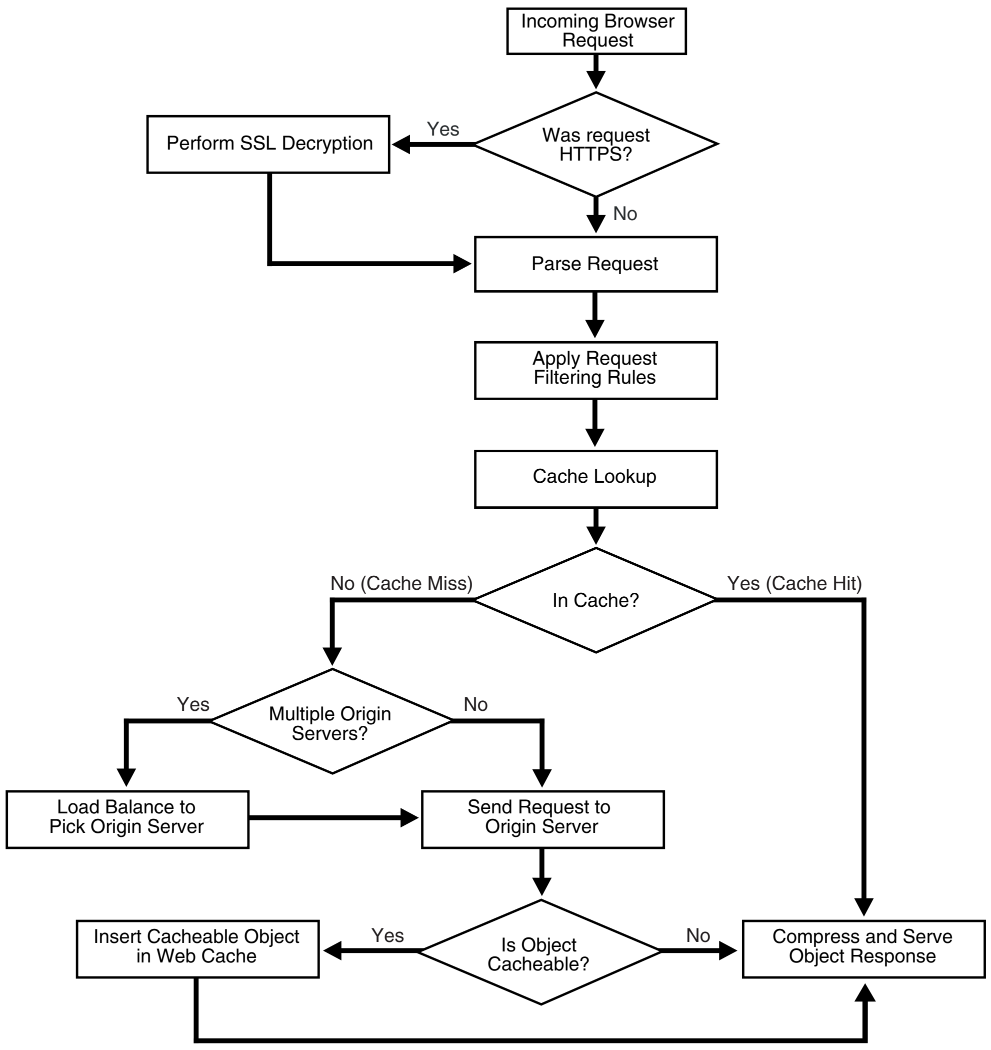
- Administration port properties
- Trusted subnets
- User and group ID information

See the *Oracle Fusion Middleware Administrator's Guide for Oracle Web Cache* for further information about starting and stopping Oracle Web Cache.

9.3.1.4 Oracle Web Cache Request Flow

Figure 9–4 shows further details of the request flow within the Oracle Web Cache tier.

Figure 9–4 Request Flow to Oracle Web Cache within the Web Tier



As shown in Figure 9–4, the following occurs within the Oracle Web Cache tier:

1. The incoming browser request is analyzed for the correct HTTP format.
2. The browser request is then further analyzed to determine if it is in HTTPS format:
   a. If the browser request is in HTTPS format, Oracle Web Cache decrypts SSL.
   b. If the browser request is not in HTTPS format, Oracle Web Cache parses the request.
3. After the request is understood, it is filtered by a set of prescribed filtering rules.
4. A cache lookup is performed to see if the HTTP request was sent previously and is present in the cache.

   If the request is present in the cache, a cache hit, the request is compressed and the content is sent directly to the browser.

   If the request is not present in the cache, a cache miss, then either:

   a. The request is sent directly to a single origin server.
   b. The request is sent to load-balanced origin servers.

Each load balanced origin server pings each Oracle Web Cache server on a periodic basis to check the status of the cache. The load balancer distributes any incoming requests among cache cluster members. If Oracle Web Cache does not have the requested content or the content is stale or invalid, it hands the request off to the application Web server. The application Web server sends the content to Oracle Web Cache. Oracle Web Cache sends the content to the client and stores a copy of the page in cache.

The proxy server is placed in a less secure zone, the Demilitarized Zone (DMZ), instead of the origin server.

Caching rules determine which objects are cached. When you establish a caching rule for a particular URL, those objects contained within the URL are not cached until there is a client request for them. When a client first requests an object, Oracle Web Cache sends the request to the origin server. This request is a cache miss. Because this URL has an associated caching rule, Oracle Web Cache caches the object for subsequent requests. When Oracle Web Cache receives a second request for the same object, Oracle Web Cache serves the object from its cache to the client. This request is a cache hit.

When you stop Oracle Web Cache, the cache clears all objects. In addition, Oracle Web Cache clears and resets statistics.

9.3.1.5 Oracle Web Cache Configuration Artifacts

Oracle stores configuration for Oracle Web Cache in the `webcache.xml` file, located in the following directories:

```
(UNIX) ORACLE_INSTANCE/instance_name/config/WebCache/webcache_name
(Windows) ORACLE_INSTANCE\instance_name\config\WebCache\webcache_name
```

Oracle offers two tools for managing the configuration files.

- Fusion Middleware Control.
- Oracle Web Cache Manager.

Use these tools rather than edit the `webcache.xml` configuration file, to perform all administrative tasks unless a specific procedure requires you to edit a file. Editing a file may cause the settings to be inconsistent and generate problems.

For further information about these tools, see the *Oracle Fusion Middleware Administrator's Guide for Oracle Web Cache*.

9.3.1.6 Log File Locations

Oracle Web Cache records event and error information in event logs. An event log entry can help you determine what objects have been inserted in the cache and alert you to any cache-related issues. Oracle Web Cache stores every request internally and then writes them in bulk at the end of the request. Request-based logging groups all the requests together.

By default, the event log has a file name of `event_log` for the Oracle Web Cache and Oracle Diagnostic Logging (ODL) text formats and `log.xml` for the ODL XML format.

Oracle Web Cache records information about the received HTTP requests in access logs. By default, the access log has a file name of `access_log`.

By default, Oracle Web Cache stores logs files in the following directories:

```
(UNIX) ORACLE_INSTANCE/diagnostics/logs/WebCache/<webcache_name>
(Windows) ORACLE_INSTANCE\diagnostics\logs\WebCache\<webcache_name>
```

For further information about these tools, see the *Oracle Fusion Middleware Administrator's Guide for Oracle Web Cache*.

9.3.2 Oracle Web Cache High Availability Considerations

The following sections describe the high availability solutions available with Oracle Web Cache:

- Section 9.3.2.1, "Oracle Web Cache Stateless Load Balancing"
- Section 9.3.2.2, "Oracle Web Cache Backend Failover"
- Section 9.3.2.3, "Oracle Web Cache Session Binding"
- Section 9.3.2.4, "Oracle Web Cache Cluster-Wide Configuration Changes"
- Section 9.3.2.5, "Oracle Web Cache as a Software Load Balancer"

9.3.2.1 Oracle Web Cache Stateless Load Balancing

Most Web sites are served by multiple origin servers running on multiple computers that share the load of HTTP and HTTPS requests. All requests that Oracle Web Cache cannot serve are passed to the origin servers. Oracle Web Cache balances the load among origin servers by determining the percentage of the available capacity, the weighted available capacity of each origin server. Oracle Web Cache sends a request to the origin server with the most weighted available capacity. The weighted available capacity is determined by the following formula:

```
(Capacity - Load) / Capacity
```

where:

- `Capacity` is the maximum number of concurrent connections that the origin server can accept
- `Load` is the number of connections currently in use

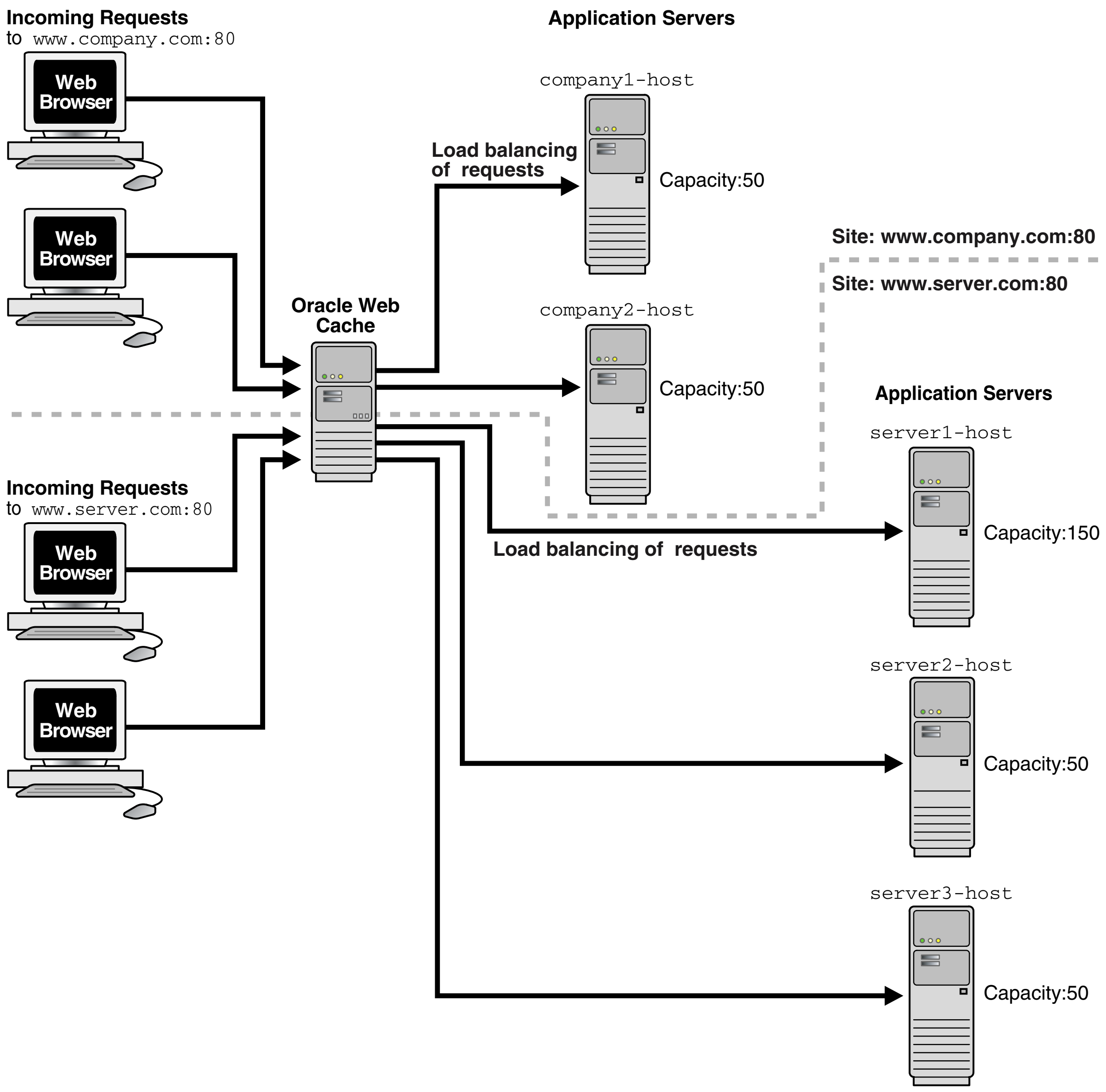
If the weighted available capacity is equal for multiple origin servers, Oracle Web Cache sends requests to the origin servers using *round robin*. With round robin, the first origin server in the list of configured servers receives the request, then the second origin server receives the second request. If the weighted available capacity is not equal, Oracle Web Cache sends the request to the origin server with the most available capacity.

If the load of origin servers is equivalent, Oracle Web Cache continues to use round robin, even when capacity is not equal for origin servers. Therefore, it is possible to see an even distribution of requests to origin server when the capacities are not configured to be the same.

To configure load balancing for a site, set the capacity of each origin server, and create *one* site-to-server mapping that maps *all* the applicable origin servers to the site.

Figure 9–5 shows two sites, `www.company.com:80` and `www.server.com:80`. The site `www.company.com:80` is supported by application Web servers `company-host1` and `company-host2` with capacities of 50 each. The site `www.server.com:80` is supported by application Web servers `server-host1`, `server-host2`, and `server-host3` with capacities of 150, 50, and 50, respectively.

Figure 9–5 Load Balancing



Assuming all application Web servers have an initial load of 0, the requests to `www.company.com:80` and `www.server.com:80` will be distributed in the following manner:

- The requests to `www.company.com:80` are distributed between the two origin servers using round robin.

  The requests to `company-host1` and `company-host2` will be distributed between the two origin servers so that they maintain an equal load. The first request is sent to `company-host1`. The second request is sent to `company-host2` if `company-host1` is still processing the first request. The third and subsequent requests are sent to the origin server that has the highest weighted available capacity.

  When the capacities are equal, Oracle Web Cache uses round robin to distribute requests.

- The requests to `www.server.com:80` are distributed between three origin servers using the weighted available capacity percentage.

  The first request to `www.server.com:80` is sent to `server-host1`, because it is the first in the configured list. The second request is sent to `server2-host`, because `server-host1` is still processing the first request and has a weighted available capacity of 99.3 percent and `server-host2` has a weighted available capacity of 100 percent. The third request is sent to `server-host3` because `server2-host` is still processing a request and has a weighted available capacity of 98 percent and `server3-host` has a weighted available capacity of 100 percent. The fourth request is sent to `server-host1` because `server-host2` and `server3-host` are still processing requests and have weighted available capacities of 98 percent. The fifth request is sent to `server-host1` because its weighted available capacity is 98.6 percent, which is still greater than `server-host2` and `server-host3`, respectively.

  When the capacities and loads are not equal, Oracle Web Cache uses the weighted available capacity to distribute requests. If requests were processed before new requests came in, then it is possible for all three origin servers to have loads of 0. In this case, Oracle Web Cache uses round robin.

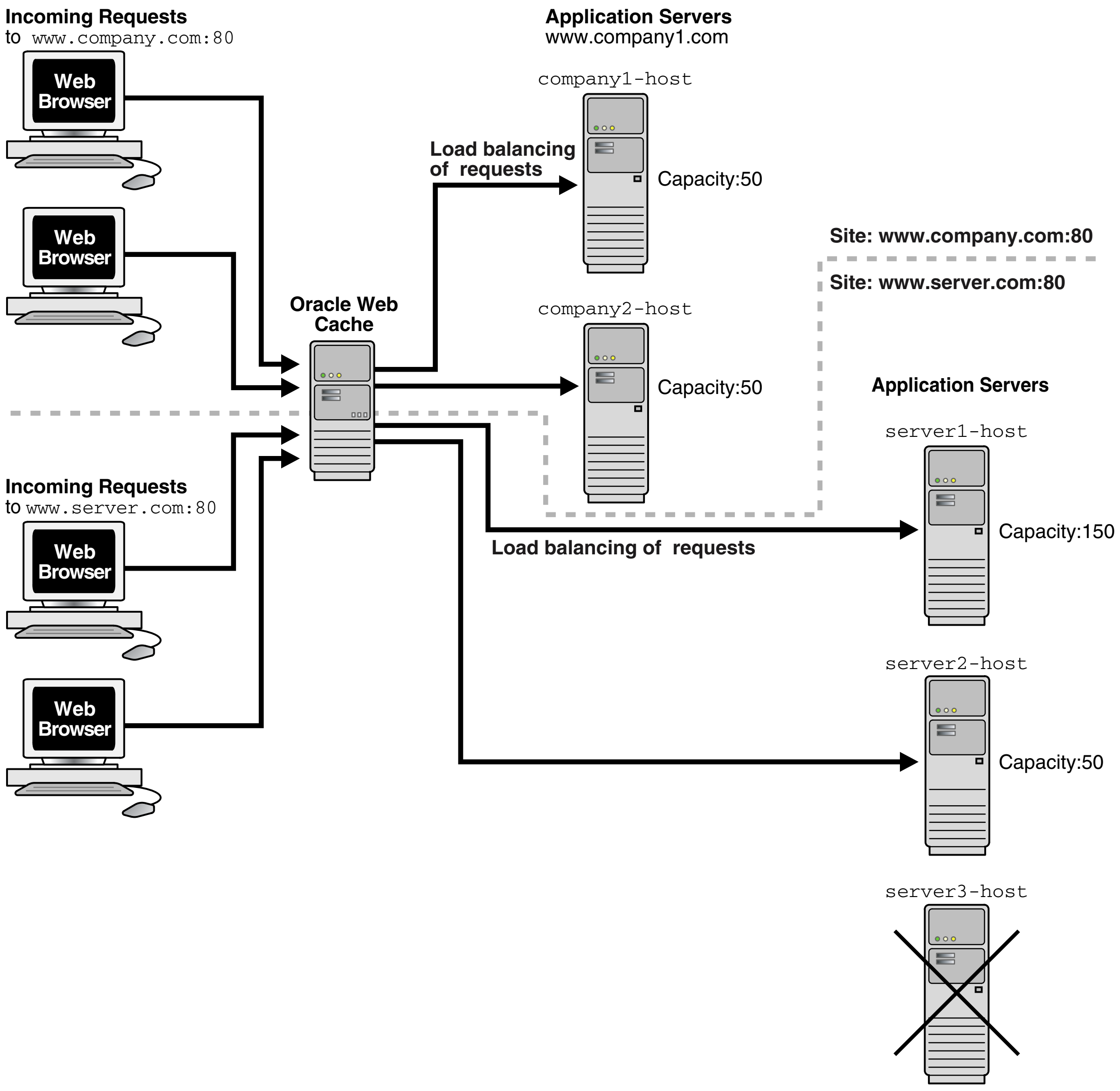
See the *Oracle Fusion Middleware Administrator's Guide for Oracle Web Cache* for instructions on specifying capacity and creating site-to-server mappings.

9.3.2.2 Oracle Web Cache Backend Failover

After a specified number of continuous request failures, Oracle Web Cache considers an origin server as failed. When an origin server fails, Oracle Web Cache automatically distributes the load over the remaining origin servers and polls the failed origin server for its current up or down status until it is back online. Existing requests to the failed origin server result in errors. However, new requests are directed to the other origin servers. When the failed server returns to operation, Oracle Web Cache includes it in its weighted available capacity to load balance requests.

The failure feature is shown in Figure 9–6. An outage of `server-host3`, which had a capacity of 50, results in 75 percent of requests being distributed to `server-host1` and 25 percent request being distributed to `server-host2`.

Figure 9–6 Failover



See the *Oracle Fusion Middleware Administrator's Guide for Oracle Web Cache* for instructions on specifying the failover threshold.

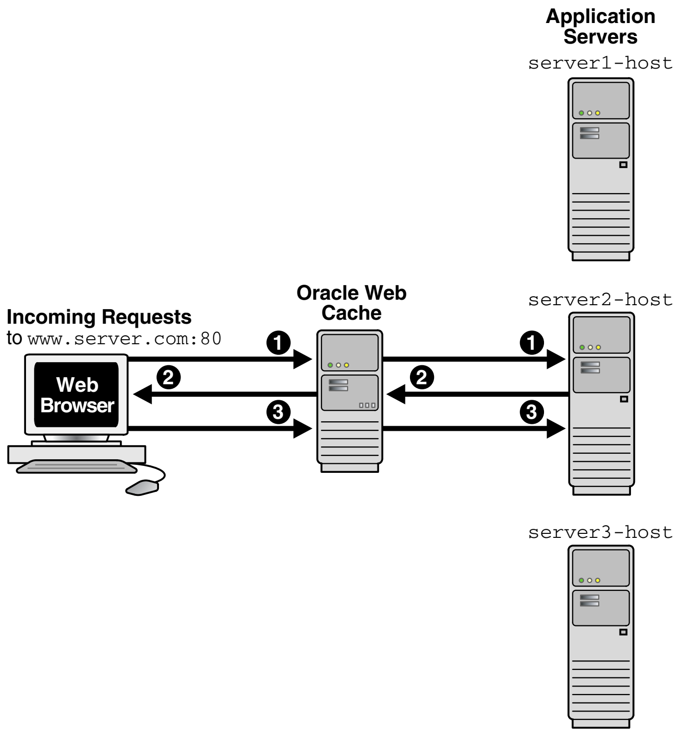
9.3.2.3 Oracle Web Cache Session Binding

You can configure Oracle Web Cache to support *session binding*, whereby a user session for a particular site is bound to an origin server in order to maintain state for a period of time. To utilize this feature, the origin server itself must maintain state; that is, it must be stateful.

If a request is forwarded to an origin server for an object requiring session binding, the origin server creates the user session by including the session information to clients through Oracle Web Cache in the form of a *session cookie*, or an embedded URL parameter. Oracle Web Cache does not process the value of the parameter or cookie; it simply passes the information back to the client browser. When a client includes the session information in a subsequent request, Oracle Web Cache forwards the request to the origin server that originally created the user session. Oracle Web Cache binds the user session to that particular origin server.

Figure 9–7 shows how Oracle Web Cache supports objects that use session binding.

Figure 9–7 Session Binding



The steps for how session binding works for requests are as follows:

1. When a request first comes in, Oracle Web Cache uses load balancing to determine to which origin server the request is forwarded. In this example, application Web server `www.server2.com` is selected.
2. If the requested object requires session binding, the origin server sends the session information back to the client through Oracle Web Cache in the form of a cookie or an embedded URL parameter.
3. Oracle Web Cache sends subsequent requests for the session to the origin server that established the session, bypassing load balancing. In this example, application Web server `www.server2.com` handles the subsequent requests.

If you configure a cache cluster, when you configure session binding, do not select the **Internal-Tracking** mechanism option, as it does not work for cache clusters. The other mechanisms work for cache clusters.

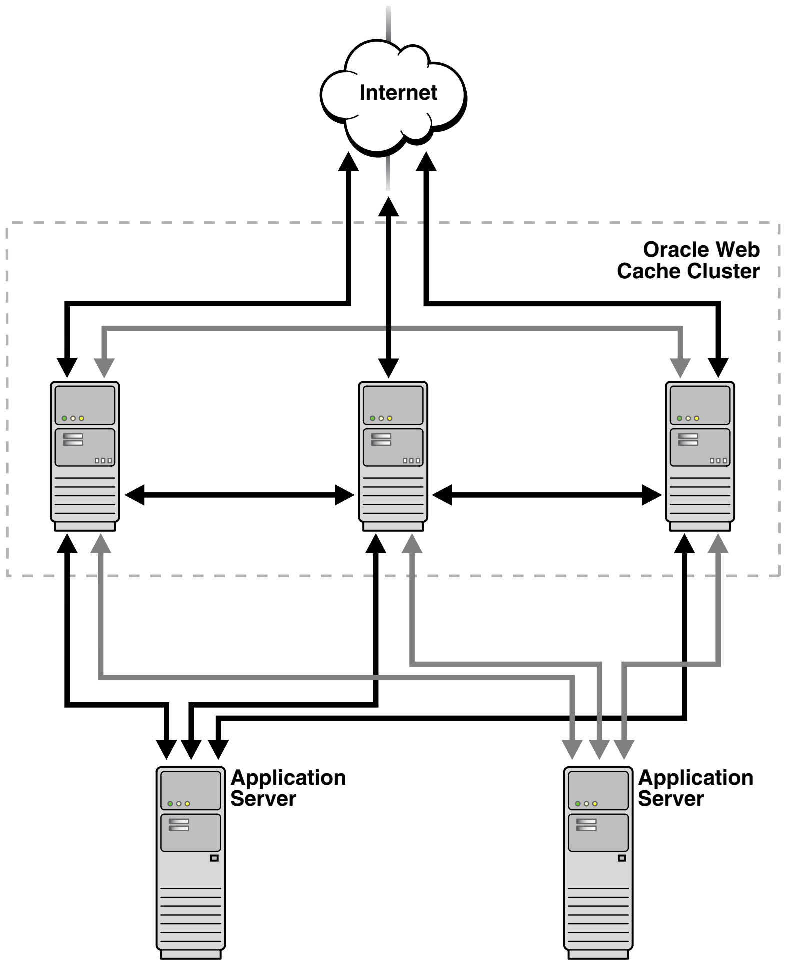
See Section 9.3.2.3 for instructions on configuring session binding.

9.3.2.4 Oracle Web Cache Cluster-Wide Configuration Changes

Oracle Web Cache provides cluster-wide capabilities through *cache clusters*. In a cache cluster, multiple system components of Oracle Web Cache operate as one logical cache. This one logical cache is referred to as the *cache cluster member*. The cache cluster members communicate with one another to request cacheable content that is cached by another cache cluster member and to detect when a cache cluster member fails.

Figure 9–8 shows an Oracle Web Cache cluster that contains three cache cluster members. As the figure shows, the cluster members communicate with one another as well as with the application Web servers and with the clients.

Figure 9–8 Oracle Web Cache Cluster Architecture



Oracle Web Cache uses the relative capacity of each cache instance to distribute the cached content among the cache cluster members. In effect, it assigns a cache cluster member to be the owner of a particular object. This content is called *owned content*.

In addition to the owned content, Oracle Web Cache stores popular objects in the cache of each cluster member. These objects are known as *on-demand content*. By storing the on-demand content, Oracle Web Cache responds to requests for those objects quickly and decreases the number of cache misses. Fewer requests are sent to the application Web server. The result is improved performance.

A cache cluster uses one configuration that is synchronize to all cluster members. The configuration contains general information, such as security, session information, and caching rules, which is the same for all cluster members. It also contains cache-specific information, such as capacity, administration and other ports, resource limits, and log files, for each cluster member.

Each member must be authenticated before it is added to the cache cluster. The authentication requires that the administration username and password of the Oracle Web Cache instance to be added be the same as the administration username and password of the cluster.

When you add a cache to the cluster, the cache-specific information of the new cluster member is added to the configuration of the cache cluster. Then, Oracle Web Cache synchronizes the configuration to all members of the cluster. Because adding a new member changes the relative capacity of each Web cache, Oracle Web Cache uses the information about capacity to recalculate which cluster member owns which content.

When cache cluster members detect the failure of another cluster member, the remaining cache cluster members automatically take over ownership of the content of the failing member. When the cache cluster member is reachable again, Oracle Web Cache again reassigns the ownership of the content.

When you remove a Web cache from a cache cluster, the remaining cache cluster members take over ownership of the content of the removed member. In addition, the configuration information about the removed member is deleted from the configuration and the revised configuration is synchronized with the remaining cache cluster members.

See Section 9.3.3.2 for instructions on configuring a cache cluster.

9.3.2.5 Oracle Web Cache as a Software Load Balancer

You can configure a special mode of Oracle Web Cache that enables you to use Oracle Web Cache solely as a software load balancer of HTTP traffic. This configuration mode is useful to:

- Manage low-volume, departmental, or test Web sites
- Manage traffic in the DMZ between a hardware load balancer and an application using Oracle Portal or OracleAS Single Sign-On Server

This mode does not cache *any* content or provide support for the following features:

- Compression: Oracle Web Cache ignores all compression settings.
- Request filtering: Oracle Web Cache ignores any configure request filters and rules.
- ESI: Oracle Web Cache does not assemble ESI content.
- Cache hierarchies: If you plan to configure two caches in a cache hierarchy, then you should not configure one of the caches as a load balancer.

You can deploy a single Oracle Web Cache server as a load balancer. However, this deployment makes the Oracle Web Cache server a single point of failure for your application. You can instead configure a cache cluster with multiple Oracle Web Cache servers in conjunction with operating system load balancing capabilities. Take note of the capacity changes mentioned earlier in this section.

In this mode, you can configure Oracle Web Cache to load balance HTTP traffic in front of an application using ESI or in front of another Oracle Web Cache. The Oracle Web Cache load balancer does not process ESI content or participate in hierarchical caching. For example, a typical Oracle Portal deployment has a built-in Oracle Web Cache used for ESI assembly. For these configurations, do not configure the Oracle Web Cache used for ESI assembly as a load balancer.

If you require other Oracle Web Cache features, such as caching or compression support, do not configure this mode. Instead, configure a hardware load balancer or operating system load balancing support, and use the load balancing feature to manage requests to origin servers.

See Section 9.3.3.3 for instructions on configuring this mode.

9.3.3 Configuring Oracle Web Cache High Availability Solutions

The following sections describe how to configure the following high availability solutions:

- Section 9.3.3.1, "Configure Oracle Web Cache Session Binding"
- Section 9.3.3.2, "Configuring a Cache Cluster"
- Section 9.3.3.3, "Configure Oracle Web Cache as a Software Load Balancer"

9.3.3.1 Configure Oracle Web Cache Session Binding

To configure session binding, you specify a set of session binding rules, and then apply them to the sites. By default, sites are configured to use a default rule. You can use the default rule or select another rule customized for the site.

If you decide to change the value of the default session binding rule, ensure all named sites currently configured with this rule require session binding. If some sites do not require session binding, leave the value of default rule, and instead specify session binding settings for the site.

To configure session binding:

1. Navigate to the Web Cache Home page in the Fusion Middleware Control.
2. From the Web Cache menu, select **Administration** and then select **Session Configuration**.

   The Session Configuration page displays.
3. From the **Site** list, select the site to create customized session-bindings.

   Select **Global** to specify default settings for sites and for requests which do not match any defined site. If you do not specify customized session-binding settings for a site, then you can click the **Use global settings** option to apply the settings you specify for **Global**. The default selection for the Global selection is the **Disable session binding**. You change the default setting by selecting **Global** from the **Site** list and selecting on of the other session-binding selections.
4. Create a session definition in the **Session Definition** table.

   - **Use global settings:** Select this option if you want to apply the session-binding settings you configured for the **Global** selection from the **Site** list.

   By default, Oracle Web Cache disables session binding for all requests. The default selection for **Global** is the **Disable session binding** option. When you first create a site, it is set by default to use the global session binding settings

   - **Disable session binding:** Select this option to disable session binding. This selection is the default for the Global site. You change the default setting by selecting **Global** from the **Site** list and selecting on of the other session-binding selections.

   - **Cookie based session binding with any Set-Cookie:** Select this option if the client supports cookies and your application server uses one or more cookies for session state. Oracle Web Cache uses its own cookie to track a session. This cookie, which contains routing information, is maintained between Oracle Web Cache and the client browser. The client to application server binding will be initiated by the first cookie that the application server sends to the client.

   - **Bind using a session:** Select this option to enable binding for a specific session, and then perform the following steps:

  a. From the **Session Name** list, select a session to enable binding for a specific session.

  b. From the Session Binding Mechanism list, select one of the following binding mechanisms for the selected session definition:

  c. Select one of the following binding mechanisms for the selected session definition:

     - **Cookie Based:** Select if the client supports cookies. Oracle Web Cache uses its own cookie to track a session. This cookie, which contains routing information, is maintained between Oracle Web Cache and the client browser.

     - **Session Binding IAS:** Select if the application is based on OC4J. Oracle Web Cache forwards routing information with each request to OC4J through Oracle HTTP Server.

     - **Internal-Tracking:** Select if the client does not support cookies and the application is not based on Oracle HTTP Server and OC4J. This option is intended for backward compatibility with earlier releases of Oracle Web Cache. Oracle Web Cache maintains an in-memory routing table, of which each entry maps a session ID to an origin server. The routing table is not shared among cluster nodes. If you select this option and you have a cache cluster configuration, then you must also bind at the load balancer layer.

5. Click **Apply** to submit changes.
6. Restart Oracle Web Cache.

9.3.3.2 Configuring a Cache Cluster

To configure a cache cluster, you configure two or more Oracle Web Cache instances as cache cluster members, and specify properties for the cluster.

A cache cluster uses one configuration that is synchronized from the current cache (the cache to which your client browser is connected) to all cluster members. The configuration contains settings that are the same for all cluster members as well as cache-specific settings for each cluster member.

This section contains the following topics to aid you in configuring a cache cluster in a environment in which all the caches are associated with the same Oracle WebLogic Server. These instruction explain how to configure a cluster using Fusion Middleware Control, which requires all the cache members to use the same Oracle WebLogic Server:

- Section 9.3.3.2.1, "Configuration Prerequisites"
- Section 9.3.3.2.2, "Understanding Failover Threshold and Capacity Settings"
- Section 9.3.3.2.3, "Task 1: Add Caches to the Cluster and Configure Properties"
- Section 9.3.3.2.4, "Task 2: Enable Tracking of Session Binding"
- Section 9.3.3.2.5, "Task 3: Synchronize Configuration to Cluster Members"

In addition, see the following information about configuring clusters:

- Section 9.3.3.2.6, "Removing a Cache Member from a Cluster"
- Section 9.3.3.2.7, "Configuring Administration and Invalidation-Only Clusters"

If you have want to configure a cache cluster for a configuration in which the caches are associated with different Oracle WebLogic Servers, follow the instructions in *Oracle Fusion Middleware Administrator's Guide for Oracle Web Cache*.

9.3.3.2.1 Configuration Prerequisites Because a cache cluster contains two or more instances of Oracle Web Cache, you must have two or more instances of Oracle Web Cache installed on one or more nodes before you configure a cache cluster. The instances must be the same version of Oracle Web Cache. In addition, the respective passwords for the Oracle Web Cache administrator, and the invalidator user, `invalidator`, must be the same across the cluster members. If they are different, you must connect to the cache's `admin` server and modify the administration password.

9.3.3.2.2 Understanding Failover Threshold and Capacity Settings To ease with configuration, take the time to understand the following key configuration settings for a cache cluster and its members:

- Failover Threshold for the Cache Cluster
- Capacity for Cache Cluster Members

Failover Threshold for the Cache Cluster

You set the failover threshold when you configure cache cluster properties. This setting reflects the number of allowed consecutive request failures before Oracle Web Cache considers another cache cluster member to have failed. In other words, Oracle Web Cache consecutively retries a failed member for certain number of times, before considering the cache-member as down. The number of times Oracle Web Cache is allowed to retry is termed as failover threshold.

Oracle Web Cache considers a request to another cache cluster member to have failed if:

- There are any network errors
- The HTTP response status code is either less than 100, or is one of the following: 500 Internal Server Error, 502 Bad Gateway, 503 Service Unavailable, or 504 Gateway Timeout.

For each failed request, Oracle Web Cache increments the failure counter for that cluster member. This counter is kept separately by each cluster member. When a request is successfully processed by a cluster member, Oracle Web Cache resets the failure counter.

When the failover threshold is met, Oracle Web Cache considers the cache cluster member to have failed. Oracle Web Cache recalculates the relative capacity of the remaining cache cluster members. It then reassigns ownership of cache content.

When a cache cluster member is down, Oracle Web Cache starts polling the cache cluster member. It does this by sending requests to the ping URL you specify. When Oracle Web Cache receives a success response from the cache cluster member, it considers that cache cluster member to be up again. It recalculates the relative capacity of the cache cluster members and it reassigns ownership of cache content.

Capacity for Cache Cluster Members

When you configure a cache cluster member, you specify capacity for that member.

Oracle Web Cache uses capacity in two different ways:

- As the absolute capacity for the number of concurrent incoming connections to this cache cluster member from all other cache cluster members.

  The connections are used to receive requests for owned content from other cache cluster members. The number of connections are divided among the other cluster members. For example, in a three-cache cluster, if the capacity of Cache_A is 50, Cache_B can open 25 connections to Cache_A and Cache_C can open 25 connections to Cache_A.

More connections are used when another cache cluster member contains little or no data in its cache, such as when it is initially started, when it recovers from a failure, or after invalidation. During this time, the cluster member sends many of the requests to its peers, the owners of the content. In most cases, these requests are served more quickly than requests to the origin server. Having a higher number of connections increases performance during this time and shortens the time it takes to fully load the cache. After a cache is fully loaded, fewer of the connections are used. There is no overhead for unused connections.

- As the relative capacity of the cache cluster member.

  The capacity of a cache cluster member is weighted against the total capacity of all active cache cluster members. When you set the capacity, Oracle Web Cache assigns a percentage of the ownership array to the cluster member, indicating how much of the cached content will be owned by the cluster member. The percentage is calculated using the following formula:

  ```
  cluster_member_capacity / total_capacity_of_all_active_cluster_members
  ```

  For example, if cache cluster member Cache_A has a capacity of 100 and cache cluster member Cache_B has a capacity of 300, for a total capacity of 400, Cache_A is assigned 25 percent of the ownership array and Cache_B is assigned 75 percent of the ownership array. That means that Cache_A owns 25 percent of the cached content.

  Note that in calculating the relative capacity, Oracle Web Cache considers the capacity of active cluster members; it does not consider the capacity of cluster members that it has determined to have failed.

Set the initial capacity for each cache cluster member to 10 percent of the **Maximum Incoming Connections** setting.

Once you have a better idea of an application's capacity needs and hit rates, fine tune the capacity. If these two assumptions apply to your cache cluster, then apply the following formula to determine the capacity for each cluster member:

1. Incoming traffic will be distributed equally to all the cache cluster members.
2. Ownership of content will be distributed equally among all the cache cluster members.

In the following formula, pick the highest value between the default value or the *max_incoming_connections* formula:

```
max(default_value, (max_incoming_connections * (cacheable_misses%/100) * (number_
of_caches - 1) / number_of_caches))
```

In the formula:

- *default_value* is one of the following:
  - 100 for production environments
  - 30 for test environments
  - 0 for invalidation-only clusters

  When the capacity increases, the number of file descriptors needed by Oracle Web Cache also increases.

- `max_incoming_connections` is the **Maximum Incoming Connections** setting from the Resource Limits page of Fusion Middleware Control.
- `cacheable_misses%` is the percentage of requests for cacheable objects that were not served directly by Oracle Web Cache, but were served by Oracle Web Cache after it fetched the content from the origin server.

  You can find the **Cacheable Misses** setting in the Web Cache Statistics page of Fusion Middleware Control.

For example, assume a cache cluster with four members. If Oracle Web Cache is operating at 1500 maximum incoming connections, with a 30 percent cacheable miss rate, then the equation to calculate capacity for this configuration looks like the following:

```
(1500 * (30/100) * (4 - 1) / 4
```

The equation calculates to 337.5. You would round up to 338, which is the capacity you would then enter for each cache cluster member.

```
1500 * .3 * 3 / 4 = 337.5
```

If you assign a capacity of 0 to a cluster member, that cluster member will not receive requests from other cluster members. However, that cluster member will forward requests to other cluster members, the owners of the content. If you assign a capacity of 0 to *all* cluster members, no requests will be forwarded between cluster members. Even when capacity is set to 0, you can still synchronize the configuration and Oracle Web Cache can automatically propagate invalidation requests to cluster members.

9.3.3.2.3 Task 1: Add Caches to the Cluster and Configure Properties Before you add a cache to the cluster, ensure the conditions described in Section 9.3.3.2.1 are met.

To add cache members to a cluster:

1. Navigate to the Web Cache Home page in the Fusion Middleware Control for one of the Oracle Web Cache instances.
2. From the Web Cache menu, select **Administration** and then select **Cluster**.

   The Cluster page displays.
3. For each cache member you want to add:
   a. Click **Add**.
   b. In the **Component** field, enter the name of the cache member.

      The **Capacity** field is auto-filled with a default value. You can modify this value. See Section 9.3.3.2.2 for more information about capacity.
4. In the **Failover Threshold** field, enter the number of allowed consecutive request failures before Oracle Web Cache considers another cache cluster member to have failed.

   The default is five failures.

   See Section 9.3.3.2.2 for further information about this field.
5. In the **Ping URL** field, enter the URL that cache cluster members will use to attempt to contact a cache cluster member that has reached its failover threshold.

   Use a URL that is cacheable and that you can guarantee is stored in each cache. The default is `_oracle_http_server_webcache_static_.html`, which is stored in the cache.

6. In the **Ping Frequency** field, enter the time, in seconds, between attempts by a cluster member to reach the failed cluster member.

   The default, 10 seconds, is a reasonable interval for most situations.

7. Click **Apply**.

9.3.3.2.4 Task 2: Enable Tracking of Session Binding

In a cache cluster, all cache cluster members must be able to determine which origin server established the session, although the request was routed originally through only one cache cluster member.

For the Oracle Web Cache you established properties for in Section 9.3.3.2.3, configure session binding with a session binding mechanism. Do not use the **Internal-Tracking** option, as it does not work for cache clusters.

To configure session binding, see Section 9.3.3.1.

9.3.3.2.5 Task 3: Synchronize Configuration to Cluster Members

When you modify the cluster and apply changes, Oracle Web Cache adds the cache-specific information from the new cache cluster members to the configuration. For those changes to take affect in all cluster members, you must synchronize the configuration and restart the cluster members.

To synchronize configuration from a newly-added cache member to the other caches in the cluster:

1. Navigate to the Web Cache Home page in the Fusion Middleware Control for the Oracle Web Cache you established properties for in Section 9.3.3.2.3.
2. From the Web Cache menu, select **Administration** and then select **Cluster**.

   The Cluster page displays.
3. Select the other cache members in the cluster, click **Synchronize**.
4. Select all the cache members, select an interval for staggering the time that the operation begins on the cache sand click **Start Up**.

The cache cluster is ready to use.

9.3.3.2.6 Removing a Cache Member from a Cluster

To remove a cache-member from a cluster, you must not only make sure that remaining cluster members no longer include that cache in cluster, but that the removed cache no longer considers itself to be part of the cluster.

To remove a cache from a cluster in Oracle Web Cache Manager:

1. Navigate to the Web Cache Home page in the Fusion Middleware Control for one of the Oracle Web Cache instances, but *not* the cache that you want to remove from the cluster.
2. From the Web Cache menu, select **Administration** and then select **Cluster**.

   The Cluster page displays.
3. Select the cache you want to remove and click **Delete.**
4. Select the other cache members in the cluster, click **Synchronize**.
5. With the other caches still selected, click **Restart**.

   All remaining caches in the cluster no longer consider the removed cache to be part of the cluster. However, the removed cache still considers itself to be part of the cluster. You will remedy this situation in the next steps.

6. Navigate to the Web Cache Home page in the Fusion Middleware Control of the cache you removed from the cluster.
7. From the Web Cache menu, select **Administration** and then select **Cluster**.

   The Cluster page displays with all the member of the cache.
8. Select a cache except the current one, and click **Delete**. Repeat until only the current cache remains in the **Cluster Members** list.
9. Click **Restart**.

9.3.3.2.7 Configuring Administration and Invalidation-Only Clusters You can configure a cluster that supports synchronizing the configuration and invalidation requests across all cache cluster members, but that does not forward requests between cache cluster members. That is, in processing requests, each cluster member acts as an individual cache and does not request objects from its peer cluster members. However, configuration changes and invalidation requests can be synchronized to cluster members.

You can use this configuration to simplify administration of many caches. It may be needed in a cluster where members are separated by a firewall. For example, you can have a cluster where two caches are located on either side of a firewall that separates the intranet from Internet. (The network settings of such a setup—of sending Internet traffic to one cache and intranet traffic to another—is beyond the scope of this document.)

To manage these caches as a cluster and avoid sharing contents between the caches, take the following steps:

1. Create a cluster and add members to it as discussed in Section 9.3.3.2.3 and Section 9.3.3.2.4, with the following exceptions:
   - For each cluster member, set the capacity to 0.
   - In the **Cluster Properties** section, select option **Invalidation requests sent to any cluster member will be propagated to all cluster members**.
2. Synchronize the configuration to all cluster members, as described in Section 9.3.3.2.5.

9.3.3.3 Configure Oracle Web Cache as a Software Load Balancer

To configure a single Oracle Web Cache server as a software load balancer:

1. Use a text editor to open `webcache.xml`, located in:

```
(UNIX) ORACLE_INSTANCE/<instance_name>/config/WebCache/<webcache_name>
(Windows) ORACLE_INSTANCE\<instance_name>\config\WebCache\<webcache_name>
```

2. Locate the `CACHE` element.
3. Add the `ROUTINGONLY` attribute to the `CACHE` element. For example:

```
...
<CACHE WCDEBUGON="NO" CHRONOSONPERNODE="NO" CAPACITY="301" VOTES="1"
INSTANCENAME="instance_name" COMPONENTNAME="component_name"
ORACLEINSTANCE="instance" HOSTNAME="host"
ORACLEHOME="directory" NAME="web_cache_name"
ROUTINGONLY="YES">
...
```

4. Save `webcache.xml`.

5. Restart Oracle Web Cache with the following command:

   ```
   opmnctl restartproc ias-component=component_name
   ```

   This executable is found in the following directory:

   ```
   (UNIX) ORACLE_INSTANCE/bin
   (Windows) ORACLE_INSTANCE\bin
   ```

6. Verify Oracle Web Cache is running in the load balancer mode from the Oracle Web Cache Manager by verifying the following status message displays beneath the **Apply Changes** and **Cancel Changes** buttons:

   ```
   Web Cache running in Routing Only Mode with current configuration
   ```

   Fusion Middleware Control Console does not provide an equivalent verification status.

7. Configure origin servers, as described in the *Oracle Fusion Middleware Administrator's Guide for Oracle Web Cache*.

8. Create site definitions and map them to the origin servers, as described in the *Oracle Fusion Middleware Administrator's Guide for Oracle Web Cache*.

9. If your application deployment requires session stickiness, enable session binding. See Section 9.3.2.3.

10 Active-Passive Topologies for Oracle Fusion Middleware High Availability

This chapter describes how to configure and manage active-passive topologies. It contains the following sections:

10.1 Oracle Fusion Middleware Cold Failover Cluster Topology Concepts

Oracle Fusion Middleware provides an active-passive model for all its components using Oracle FMW Cold Failover Clusters. In an Oracle FMW Cold Failover Cluster configuration, two or more application server instances are configured to serve the same application workload but only one is active at any particular time.

A two-node Oracle FMW Cold Failover Cluster can be used to achieve active-passive availability for Oracle Application Server middle-tier components. In an Oracle FMW Cold Failover Cluster, one node is active while the other is passive, on standby. In the event that the active node fails, the standby node is activated, and the middle-tier components continue servicing clients from that node. All middle-tier components are failed over to the new active node. No middle-tier components run on the failed node after the failover.

The most common properties of an Oracle FMW Cold Failover cluster configuration include:

- **Shared Storage**: The passive Oracle Fusion Middleware instance in an active-passive configuration has access to the same Oracle binaries, configuration files, domain directory, and data as the active instance. You configure this access by placing these artifacts in storage that can be accessed by all participating nodes in the Cold Failover Cluster configuration. Typically the active node has shared storage mounted, while the passive node's is unmounted but accessible if the node becomes active. The shared storage can be a dual ported disk device accessible to both the nodes, or a device-based storage such as a NAS or a SAN. You can install shared storage on a regular file system. With Cold Failover Clusters, you mount the volume on one node at a time. Shared storage is a key property of a Cold Failover Cluster deployment.

- **Virtual hostname**: In the Cold Failover Cluster solution, a virtual hostname and a virtual IP are shared between the two nodes (the virtual hostname maps to the virtual IP and is used interchangeably in this guide). However, only one node, the active node, can use this virtual IP at any one time. When the active node fails and the standby node is made active, the virtual IP is moved to the new active node. The new active node now services all requests through the virtual IP. The Virtual Hostname provides a single system view of the deployment. A Cold Failover Cluster deployment is configured to listen on this virtual IP. For example, if the two physical hostnames of the hardware cluster are node1.mycompany.com and node2.mycompany.com, the single view of this cluster can be provided by the name cfcvip.mycompany.com. In the DNS, cfcvip.mycompany.com maps to the virtual IP, which floats between node1 and node2. When a hardware cluster is used, it manages the failover of the virtual IP without the middle tier clients detecting which physical node is active and actually servicing requests.
- **Hardware Cluster**: A hardware cluster is typically used for Cold Failover Cluster deployments. The hardware cluster addresses the management of shared storage and virtual IP in its architecture. It plays a role in reliable failover of these shared resources, providing a robust active-passive solution. Most Cold Failover Cluster are deployed to hardware clusters that include the following:
  - Two nodes that are in the same subunit
  - A high speed private interconnect between the two nodes
  - Public network interfaces, on which the client requests are served and the virtual IP is enabled.
  - Shared storage accessible by the two nodes. This includes shared storage that acts as a quorum device as well as shared storage for Oracle Fusion Middleware and database installs.
  - Clusterware running to manage node and component failures
- **Planned Switchover and Unplanned Failover**: The typical Cold Failover Cluster deployment is a two-node hardware cluster. To maximize utilization, both of these nodes typically have some elements of the deployment running, with the other node acting as a backup node for the appropriate element if needed. For example, a deployment may have the application tier (WebLogic container) running on one node and the Web tier (Oracle HTTP Server) running on the other node. If either node is brought down for hardware or software maintenance, or if either node crashes, the surviving node is used to host the services of the down node while continuing to host its current services.

  The high-level steps for switch-over to the standby node are as follows:

  1. Stop the middle-tier service on the primary node (if the node is still available).
  2. Fail over the virtual IP from the current active node, to the passive node. This involves bringing it down on the current node and enabling it and bringing up on the passive node.
  3. Failover the shared disk from the current active node to the passive node. This involves unmounting the shared disk from the current node and mounting it on the passive node.
  4. Start the middle-tier service on the passive node, which becomes active.

  For failover management, there are two possible approaches:

  - Automated failover using a cluster manager facility

The cluster manager offers services, which allows development of packages to monitor the state of a service. If the service or the node is found to be down (either due to planned operation or unplanned operation), it automatically fails over the service from one node to the other node. The package can be developed to attempt to restart the service on a given node before failing over. Similarly, when the active node itself is down, the clusterware can detect the down state of the node and attempt to bring it up on the designated passive node. These can be automated using any clusterware to provide failover in a completely automated fashion.

Oracle Fusion Middleware provides this capability using Oracle Clusterware. Please refer to Chapter 11, "Using Oracle Cluster Ready Services" for details of managing a Cold Failover Cluster environment with Oracle Cluster-Ready Services. However, you can use any available clusterware to manage cluster failover.

– Manual failover

  For this approach, the planned switchover steps are executed manually. Both the detection of the failure and the failover itself is manual, Therefore this method may result in a longer period of service unavailability.

In active-passive deployments, services are typically down for a short period of time. This is the time taken to either restart the instance on the same node, or to failover the instance to the passive node.

Active-Passive Topologies: Advantages

- Increased availability

  If the active instance fails for any reason or must be taken offline, an identically configured passive instance is prepared to take over at any time. This provides an increased level of availability than a normal single instance deployment. Active-passive deployments also are used to provide availability and protection against planned and unplanned maintenance operation of the hardware used. When a node needs to be brought down for planned maintenance such as a patch apply or OS upgrade or any other reason, the middle ware services can be brought up on the passive node. Switchback to the old node can be done at appropriate times.

- Reduced operating costs

  In an active-passive configuration only one set of processes is up and serving requests. Management of the active instance is generally less costly than managing an array of active instances.

- Cold Failover Clusters are less difficult to implement because they do not require a load balancer, which is required in active-active topologies

- Cold Failover Clusters are less difficult to implement than active-active topologies because you are not required to configure options such as load balancing algorithms, clustering, and replication.

- Active-passive topologies better simulates a one-instance topology than active-active topologies.

- Application independence

  Some applications may not be suited to an active-active configuration. This may include applications which rely heavily on application state or on information stored locally. Singleton application by very nature are more suitable for active-passive deployments. An active-passive configuration has only one instance serving requests at any particular time.

Active-Passive Topologies: Disadvantages

- Active-passive topologies do not scale as well as active-active topologies. You cannot add nodes to the topology to increase capacity.
- State information from HTTP session state and EJB stateful session beans is not replicated, and therefore gets lost when a node terminates unexpectedly. Such state can be persisted to the database or to the file system residing on a shared storage, however, this requires additional overhead that may impact performance of the single node Cold Failover Cluster deployment.
- Active-passive deployments have a shorter downtime than a single node deployment. However, downtime is much shorter in an active-active deployment.

10.2 Configuring Oracle Fusion Middleware for Active-Passive Deployments

Oracle Fusion Middleware components come in a variety of Java EE container-deployed components and non-Java EE components. Components such as Oracle Internet Directory, Oracle Virtual Directory, Oracle HTTP Server, Oracle Web Cache, Runtime Engines of Oracle Forms, and Oracle Reports are system components. Components such as the Oracle SOA Suite, Oracle WebCenter Suite, Oracle Identity Management components, such as ODSM and DIP are Java EE components that are deployed to Oracle WebLogic Server.

WebLogic Server Administration Console and Oracle Enterprise Manager Fusion Middleware Control are also deployed to the WebLogic container. Both Java EE and system components can be deployed to Cold Failover Cluster environments. They can co-exist on the same system or on different systems. When on the same system, you can configure them to failover as a unit, sharing the same virtual IP, or failover independently using separate virtual IPs. In most Oracle Fusion Middleware deployments, a database is used either for the component metadata created using Repository Creation Utility (RCU), or for application data. In many cases, a Cold Failover Cluster middle tier deployment uses a Cold Failover Cluster database, both deployed to the same cluster. The typical deployment has the two components configured as separate failover units using different VIPs and different shared disks on the same hardware cluster.

For Oracle Fusion Middleware, the recommended procedure to create an active-passive topology is:

- Install the component as a single instance configuration. If you planned to transform this instance to a Cold Failover Cluster deployment, install it using a shared disk. This means that the Middleware home, the Instance home, in the case of system components, and the domain directory, in case of a WebLogic deployment are on a shared disk. Everything that fails over as a unit should be on a shared disk.
- After the installation, you transform the deployment into a Cold Failover Cluster deployment, and configure it to listen on a Virtual IP. The Virtual IP is configured on the current active node. It fails over, along with the Oracle Fusion Middleware deployment to the passive node when failure occurs.

This general procedure applies to the Cold Failover Cluster Oracle database. For example, the Oracle database instance is installed as a single instance deployment and subsequently transformed for Cold Failover Clusters. A Cold Failover Cluster Oracle

Fusion Middleware deployment can also use an Oracle Real Application Cluster (RAC) database.

The following sections describe the procedures for post-installation configuration to transform a single instance deployment to a Cold Failover Cluster deployment.

The rest of this chapter describes how to transform Cold Failover Clusters for each of the individual components in the Oracle Fusion Middleware suite. The first section details the procedure for the basic infrastructure components, and the subsequent section does so for the individual Oracle Fusion Middleware component. Any given deployment, for example an Oracle instance or domain, has more than one of these present in a given machine. To transform the entire instance or domain:

- Decide which components form a unit of failover.
- Deploy them on the same shared disk.

> **Note:** For details about installing and deploying Oracle Fusion Middleware components, see the installation guide for the specific Fusion Middleware component.

- Determine a virtual IP to use for this unit of failover. Typically, a single virtual IP is used for all the components, but separate IPs can be used as long as all of them fail over together.
- Apply the transformation procedure to each of the individual components to transform the deployment as a whole. Since more than one of these sections will apply for Cold Failover Clusters transformation of an installation, the order of transformation should always be as follows:
  - Transform the administration server or Enterprise Manager instance (if applicable).
  - Transform all managed servers in the deployment.
  - Transform the Oracle instances (non-Java EE deployments).

10.2.1 General Requirements for Cold Failover Clusters

As described earlier, a Cold Failover Clusters deployment has at least two nodes in the deployment. The installation is done on one of these nodes. The other node is the passive node. The requirements on these two nodes are as follows:

- The nodes should be similar in all respects at the operating system level. For example, they should be the same operating system, the same version, and the same patch level.
- The nodes should be similar in terms of the hardware characteristics. This ensures predictable performance during normal operations and on failover. Oracle suggests designing each node for capacity to handle both its normal role, as well as the additional load required to handle a failover scenario. However, if the SLA agreement indicates that during outage scenarios, reduced performance is acceptable, this is not required.
- The nodes should have the same mount point free so that mounting of shared storage can occur to the same node during normal operations and failover conditions.
- The user ID and group ID on the two nodes are similar, and the user ID and group ID of the user owning the instance is the same on both nodes.

- The `oraInventory` location is the same on both nodes and has similar accessibility of the instance or domain owner. The location of the `oraInst.loc` file, as well as the `beahomelist` file should be the same.
- Since a given instance uses the same listen ports irrespective of the machine o which it is currently active, it is required to ensure that the ports used by the Cold Failover Cluster instance are free on both the nodes.

Note: Before beginning the transformation, back up of your entire domain. Oracle also recommends creating a local backup file before editing it. For details on backing up your domain, see the *Oracle Fusion Middleware Administrator's Guide*. Oracle recommends backing up:

- All domain directories
- All Instance homes
- Optionally, the database repository and the Middleware homes
- For all the sections below, before changing any file, ensure that a local backup copy is made before editing the file.

10.2.1.1 Terminology for Directories and Directory Environment Variables

The following list describes the directories and variables used in this chapter:

- ORACLE_BASE: This environment variable and related directory path refers to the base directory under which Oracle products are installed.
- MW_HOME: This environment variable and related directory path refers to the location where Fusion Middleware (FMW) resides.
- WL_HOME: This environment variable and related directory path contains installed files necessary to host a WebLogic Server.
- ORACLE_HOME: This environment variable and related directory path refers to the location where Oracle FMW SOA Suite is installed.
- DOMAIN Directory: This directory path refers to the location where the Oracle WebLogic Domain information (configuration artifacts) is stored.
- ORACLE_INSTANCE: An Oracle instance contains one or more system components, such as Oracle Web Cache, Oracle HTTP Server, or Oracle Internet Directory. An Oracle instance directory contains updateable files, such as configuration files, log files, and temporary files.

The values used and recommended for consistency for these directories are:

- `ORACLE_BASE: /u01/app/oracle`
- `MW_HOME (Apptier): ORACLE_BASE/product/fmw`
- `WLS_HOME: MW_HOME/wlserver_10.3`

Component	ORACLE_HOME	DOMAIN_HOME	Domain Directory
Identity Management	MW_HOME/idm	IDMDomain	MW_HOME/user_ projects/domains/IDMDomain
Oracle SOA	MW_HOME/soa	SOADomain	MW_HOME/user_ projects/domains/SOADomain

Component	ORACLE_HOME	DOMAIN_HOME	Domain Directory
WebCenter	MW_HOME/wc	WCDomain	MW_HOME/user_projects/domains/WCDomain
Oracle Portal	MW_HOME/portal	PortalDomain	MW_HOME/user_projects/domains/PortalDomain
Oracle Forms	MW_HOME/forms	FormsDomain	MW_HOME/user_projects/domains/FormsDomain
Oracle Reports	MW_HOME/reports	ReportsDomain	MW_HOME/user_projects/domains/ReportsDomain
Oracle Discoverer	MW_HOME/disco	DiscoDomain	MW_HOME/user_projects/domains/DiscoDomain
Web Tier	MW_HOME/web		
Directory Tier	MW_HOME/idm		

Location for Applications Directory: ORACLE_BASE/admin/domain_name/apps

Location for Oracle Instance: ORACLE_BASE/admin/instance_name

All entities on disk that failover as a unit should preferably be on the same mount point. They can, however, be on separate shared storage, on separate mount points as well. Oracle recommends that the mount point for the shared storage is ORACLE_BASE. In most cases, this ensures that all the persistent bits of a failover unit are on the same shared storage. When more than one Cold Failover Cluster exists on a node, and each fails over independent of the other, different mount points will exist for each such failover unit.

10.2.2 Transforming Oracle Fusion Middleware Infrastructure Components

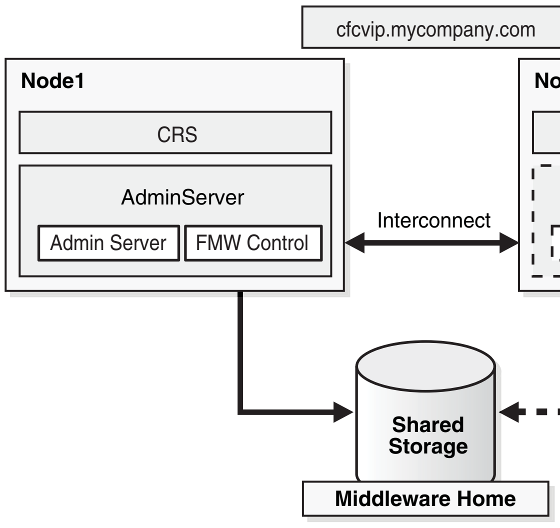
An Oracle Fusion Middleware deployment is made up of basic infrastructure components that are common across all the product sets. This section describes Cold Failover Clusters transformation steps for these components.

There are two administration server topologies supported for Cold Failover Clusters configuration. The following sections describe these two topologies and provide installation and configuration steps to prepare the administration server for Cold Failover Clusters transformation.

10.2.2.1 Administration Server Topology 1

Figure 10–1 illustrates the first supported topology for Oracle Cold Failover Clusters.

Figure 10–1 Administration Server Cold Failover Cluster Topology 1



In Figure 10–1, the Administration Server runs on a two-node hardware cluster: Node 1 and Node 2. The Administration Server is listening on the Virtual IP or hostname. The Middleware Home and the domain directory is on a shared disk that is mounted on Node 1 or Node 2 at any given point. Both the Middleware home and the domain directory should be on the same shared disk or shared disks that can fail over together. If an enterprise has multiple Fusion Middleware domains for multiple applications or environments, this topology is well suited for Administration Server high availability. A single hardware cluster can be deployed to host these multiple Administration Servers. Each Administration Server can use its own virtual IP and set of shared disks to provide high availability of domain services.

10.2.2.1.1 Topology 1 Installation Procedure To install and configure Cold Failover Clusters for the application server in this topology:

Install the Middleware Home

This installation includes the Oracle home, WebLogic home, and the Domain home on a shared disk. This disk should be mountable by all the nodes that act as the failover destination for the administration server. Depending on the storage sub-system used, the shared disk may be mountable only on one node at a time. This is the preferred configuration, even when the storage sub system allows simultaneous mounts on more than one node. This is done as a regular single-instance installation. Please refer to the component chapters for details on installing the administration server (and Enterprise Manager) alone. The overall procedure for each suite is as follows:

For Oracle SOA or Oracle WebCenter:

1. Install the WebLogic Server software.

   See the *Oracle Fusion Middleware Installation Guide for Oracle WebLogic Server*.

2. Install the Oracle Home for Oracle SOA or WebCenter.

   See the *Oracle Fusion Middleware Installation Guide for Oracle SOA Suite* or the *Oracle Fusion Middleware Installation Guide for Oracle WebCenter*.

3. Invoke the Configuration Wizard and create a domain with just the administration server.

   In the Select Domain Source screen, select the following:

- **Generate a domain configured automatically to support the following products**
- Select **Enterprise Manager** and **Oracle JRF**.

For Oracle Identity Management:

1. Install the WebLogic Server software.

   See the *Oracle Fusion Middleware Installation Guide for Oracle WebLogic Server*.

2. Using Oracle Identity Management 11g Installer, install and configure the IDM Domain using the create domain option. In the Configure Components Screen, de-select everything except **Enterprise Manager** (this is selected by default)

   See the *Oracle Fusion Middleware Installation Guide for Oracle Identity Management*.

For Oracle Portal, Forms, Reports and Discoverer:

1. Install the WebLogic Server software.
2. Using Oracle Fusion Middleware 11g Portal, Forms, Reports, and Discoverer Installer, install and configure the Classic Domain using the create domain option. In the Configure Components Screen, make sure that **Enterprise Manager** is also selected.

> **Note:** In this case, at least one more Managed Servers for the product components is also installed in this process (Adminserver by itself cannot be installed). This Managed Server must also be transformed to CFC using the specific procedure for the component. It is part of the same failover unit as the administration server.

Configuring the Administration Server for Cold Failover Clusters

To configure the administration server for Cold Failover Clusters:

1. Provision the Virtual IP using the following commands as root user:

   For Linux

   ```
   /sbin/ifconfig interface:index IP_Address netmask netmask
   /sbin/arping -q -U -c 3 -I interface IP_Address
   ```

   Where *IP Address* is the virtual IP address and the *netmask* is the associated netmask. In the following example, the IP address is being enabled on the interface eth0.

   ```
   ifconfig eth0:1 130.35.46.17 netmask 255.255.224.0
   /sbin/arping -q -U -c 3 -I eth0 130.35.46.17
   ```

   For Windows:

   ```
   netsh interface ip add address interface IP Address netmask
   ```

   Where *IP Address* is the virtual IP address and the *netmask* is the associated netmask. In the example below, the IP address is being enabled on the interface 'Local Area Connection'.

   ```
   netsh interface ip add address "Local Area connection" 130.35.46.17
   255.255.224.0
   ```

2. Transform the administration server instance to Cold Failover Clusters following the procedure in section Section 10.2.2.3, "Transforming the Administration Server for Cold Failover Clusters."

3. Validate the administration server transformation by accessing the consoles on the virtual IP.

   http://cfcvip.mycompany.com:7001/console

   http://cfcvip.mycompany.com:7001/em

4. Failover the administration server manually to the second node using the following procedure:

   a. Stop the administration server process (and any other process running out of a given Middleware Home)

   b. Unmount the shared storage from Node1 where the Middleware Home and domain directory exists.

   c. Mount the shared storage on Node2, follow storage specific commands.

   d. Disable the Virtual IP on Node1 using the following command as root user:

      For Linux:

      ```
      /sbin/ifconfig interface:index down
      ```

      Where *IP Address* is the virtual IP. In the example below, the IP address is being disabled on the interface eth0.

      ```
      /sbin/ifconfig eth0:1 down
      ```

      On Windows:

      ```
      netsh interface ip delete address interface IP_Address
      ```

      Where *IP Address* is the virtual IP address and the *netmask* is the associated netmask. In the example below, the IP address is being enabled on the interface 'Local Area Connection'.

      ```
      netsh interface ip delete address "Local Area connection" 130.35.46.17
      ```

   e. Enable the virtual IP on Node2 using similar commands as in Step 1.

   f. Start the administration server process.

      ```
      DOMAIN_HOME/bin/startWebLogic.sh
      ```

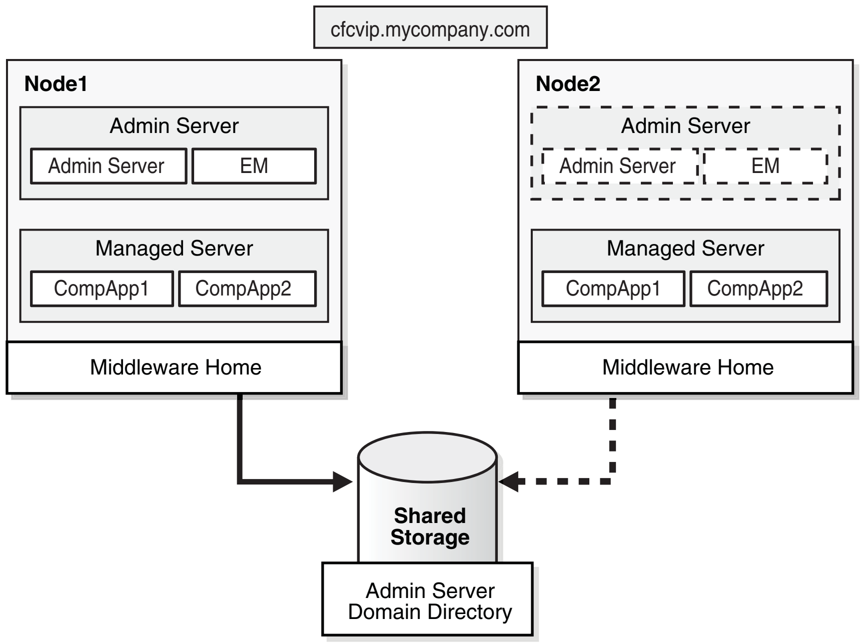
      Where *DOMAIN_HOME* is the location of your domain directory.

   g. Validate access to both the administration server and Enterprise Manager console.

10.2.2.2 Administration Server Topology 2

Figure 10–2 illustrates the second supported administration server topology for Oracle Cold Failover Clusters.

Figure 10–2 Administration Server Cold Failover Cluster Topology 2



In Figure 10–2, the administration server runs on a two-node hardware cluster: Node 1 and Node 2. The administration server is listening on the Virtual IP or hostname. The domain directory used by the administration server is on a shared disk. This is mandatory. This shared disk is mounted on Node 1 or Node 2 at any given point. The Middleware Homes, which contain the software, (WebLogic Home and the Oracle Home) are not necessarily on a shared disk. They can be on the local disk as well. The administration server uses the Middleware Home on Node1 for the software when it is running on Node1 and it uses the Middleware Home on Node2 when it is running on Node2. These two Middleware Home must be maintained to be the same in terms of deployed products, Oracle Homes, and patches. In both cases, it uses the configuration available in the shared Domain Directory/Domain Home. Since this is shared, it ensures that the same configuration is used before and after failover.

This shared domain directory may also have other Managed Servers running. It may also be used exclusively for the administration server. If the domain directory is shared with other managed servers, appropriate consideration must be made for their failover when the administration server fails over. Some of these considerations are:

1. If the shared storage can be mounted as read/write on multiple nodes simultaneously, the administration server domain directory can be shared with other managed servers. In addition, it can be failed over independently of the Managed Server. The administration server can failover and Managed Servers can continue to run independently on their designated nodes. This is possible because the administration server in this case requires only failover of the VIP, and does not require failover of the shared disk. The domain directory/domain home continues to remain available by the Managed Servers. Example of such storage include a NAS or a SAN/Direct attached storage with Cluster file system.
2. If only one node can mount the shared storage a time, sharing the administration server domain directory with a Managed Server implies that when the administration server fails over, the Managed Server that runs off the same domain directory must be shut down.

A hardware cluster may be used in this topology. The cluster helps automate failover (when used with properly configured clusterware). However, it is not required. A single hardware cluster can be deployed to host these multiple administration servers. Each administration server can use its own virtual IP and set of shared disks to provide domain services high availability.

This topology is supported for Oracle SOA Suite and Oracle Web Center Suite only.

Note: For the Oracle Identity Management, an alternate topology is also supported for Cold Failover Clusters. See the *Oracle Fusion Middleware Enterprise Deployment Guide for Oracle Identity Management* for more details.

10.2.2.2.1 Topology 2 Installation Procedure To install and configure Cold Failover Clusters for the administration server in this topology:

Install the Middleware Home

Install the Middleware Home including the Oracle Home and WebLogic Home separately on the two nodes of the domain. The administration server domain directory is created on a shared disk. This disk should be mountable by all the nodes that act as the failover destination for the administration server. Depending on the storage sub-system used, the shared disk may be mountable only on one node at a time. This is a regular single-instance installation. Refer to the product suite for details on installing the administration server, and Enterprise Manager alone. To install the Middleware Home:

For Oracle SOA Suite or Oracle WebCenter:

1. Install the Oracle WebLogic Server software on Node 1.
2. Install the Oracle Home for SOA or WebCenter on Node 1.
3. Repeat steps 1 and 2 on Node 2.
4. Start the Configuration Wizard on Node 1 and create a domain with just the administration server.

   In the Select Domain Source screen, select the following:

   - **Generate a domain configured automatically to support the following products**.
   - Select **Enterprise Manager** and **Oracle JRF**.
5. In the Specify Domain Name and Location screen, enter the domain name, and be sure the domain directory matches the directory and shared storage mount point.

Configuring the Middleware Home for Cold Failover Clusters

To configure the Middleware Home for Cold Failover Clusters:

1. Provision the Virtual IP. For example:

   For Linux:

   ```
   ifconfig eth0:1 IP_Address netmask netmask
   /sbin/arping -q -U -c 3 -I eth0 IP_Address
   ```

   Where *IP_Address* is the virtual IP address and the *netmask* is the associated netmask. In the following example, the IP address is being enabled on the interface eth0.

   ```
   ifconfig eth0:1 130.35.46.17 netmask 255.255.224.0
   /sbin/arping -q -U -c 3 -I eth0 130.35.46.17
   ```

For Windows:

```
netsh interface ip add address interface IP_Adress netmask
```

Where *IP_Address* is the virtual IP address and the *netmask* is the associated netmask. In the example below, the IP address is being enabled on the interface "Local Area Connection".

```
netsh interface ip add address "Local Area connection" 130.35.46.17
255.255.224.0
```

2. Transform the administration server instance to Cold Failover Clusters using the procedure in Section 10.2.2.3, "Transforming the Administration Server for Cold Failover Clusters."
3. Validate the administration server by accessing the consoles on the virtual IP.

   http://cfcvip.mycompany.com:7001/console

   http://cfcvip.mycompany.com:7001/em
4. Failover the administration server manually to the second node:
   a. Stop the administration server process (and any other process running out of a given Middleware Home).
   b. Unmount the shared storage from Node1 where the Middleware Home or domain directory exists.
   c. Mount the shared storage on Node2, following storage specific commands.
   d. Disable the virtual IP on Node1:

      For Linux:

      ```
      ifconfig interface:index down
      ```

      In the following example, the IP address is being disabled on the interface `eth0`.

      ```
      ifconfig eth0:1 down
      ```

      For Windows:

      ```
      netsh interface ip delete address interface addr=IP Address
      ```

      Where *IP Address* is the virtual IP address. In the following example, the IP address is enabled on the interface `'Local Area Connection'`.

      ```
      netsh interface ip delete address 'Local Area connection' addr=130.35.46.17
      DOMAIN_HOME/bin/startWebLogic.sh
      ```

      Where *DOMAIN_HOME* is the location of your domain directory.
   e. Enable the virtual IP on Node 2.
   f. Start up the administration server process using the following command:

      ```
      DOMAIN_HOME/bin/startWebLogic.sh
      ```

      Where *DOMAIN_HOME* is the location of your domain directory.
   g. Validate access to both administration server and Enterprise Manager console.

10.2.2.3 Transforming the Administration Server for Cold Failover Clusters

To transform the Administration Server installed on a shared disk from Node 1, follow the steps in this section. These step transform the container, therefore, both the WebLogic Server Administration Console and Oracle Enterprise Manager Fusion Middleware Control, are transformed for Cold Failover Clusters. This results in other components such as OWSM-PM, deployed to this container, to become Cold Failover Clusters ready as well. The address for all of these services transforms cfcvip.mycompany.com. After installation, to transform a non Cold Failover Cluster instance, to a Cold Failover Cluster:

1. Login into WebLogic Server Administration Console.
2. Create a Machine for the Virtual Host
   a. Select **Environment**, and then **Machines**.
   b. Click **Lock & Edit**.
   c. Click **New**.
   d. In the Name field, enter **cfcvip.mycompany.com**
   e. Select the appropriate operating system.
   f. Click **OK**.
   g. Click the **Servers** tab.
   h. Click **Add**.
   i. Select an existing server, and associate it with this machine.

      In the select server drop down list ensure AdminServer is selected.
   j. Click **Activate Changes**.
3. Configure the Admin Server to Listen on cfcvip.mycompany.com.
   a. Select **Environment**, and then **Servers** from the Domain Structure menu.
   b. Click **Lock and Edit** from the Change Center.
   c. Click on the administration server (**AdminServer**)
   d. Change the Listen Address to **cfcvip.mycompany.com**
   e. Click **Save**.
   f. Click **Activate Changes**.
   g. Restart the administration server.

Note: Since it is typically expected that transformation to Cold Failover Cluster for the administration server is done at domain creation time No other changes to other parts of the domain are expected. If this change happens post domain creation, and other components are installed in the domain, follow the steps in the following administration server transformation section.

Changing Client Side Configuration for Administration Server

Any existing entities in the domain must communicate with the administration server using the new address. For example, when starting the Managed Servers manually, the administration server address should be specified as `cfcvip.mycompany.com`.

In the instance.properties file, located in the *INSTANCE_HOME*/OPMN/opmn directory, make the following change:

```
adminHost=cfcvip.mycompany.com
```

If the Oracle Instance is to be registered or re-registered with a Cold Failover Clusters administration server using the OPMN registration commands, the `AdminHost` location in the opmnctl command should reference the new location of the administration server (cfcvip.mycompany.com).

Changing Client Side configuration for Oracle Enterprise Manager

Since the Enterprise Manager is part of the same container where the administration server runs, transforming the administration server to Cold Failover Clusters also transforms the Enterprise Manager. If there are existing Enterprise Manager Agents configured to be part of the domain, these agent configurations must use the new location for the Enterprise Manager. To configure the new location for Enterprise Manager, use the following steps for each agent:

1. Set the directory to *ORACLE_INSTANCE*/EMAGENT/emagent_asinst_1/sysman/config.
2. In the `emd.properties` file, change `node1.mycompany.com` to `cfcvip.mycompany.com` in the following attributes:
   - REPOSITORY_URL
   - EmdWalletSrcUrl
3. Stop and restart the agent using the following commands:

   ```
   cd INSTANCE_HOME/EMAGENT/emagent_dir/bin
   ./emctl stop agent
   ./emctl start agent
   ./emctl status agent
   ```

   This shows the Repository URL and it should now point to the new host.

10.2.2.4 Transforming Oracle WebLogic Managed Servers

All Oracle Fusion Middleware components are deployed to a Managed Server. An important step to convert an application or component that is deployed to Oracle WebLogic Server to Cold Failover Clusters is to change its listen address to the virtual IP being used. This change is done for the specific Managed Server to which the component has been deployed. You can make this change using the WebLogic Server Administration Console or using WLST commands.

The following example describes the generic steps for Cold Failover Clusters transformation of a Managed Server named WLS_EXMPL. These steps apply to any Managed Server in the Fusion Middleware components.

10.2.2.4.1 Transforming an Oracle WebLogic Managed Server using the Fusion Middleware Administration Console For this procedure, the WebLogic Server Administration Console must be running. In the following example, cfcvip.mycompany.com is the virtual IP used for the Cold Failover Clusters, and WLS_EXMPL is the managed server to be transformed.

1. Log into the WebLogic Server Administration Console.
2. Create a machine for the virtual host:
   a. Select **Environment > Machines**.

   b. Click **Lock & Edit**.
   c. Click **New**.
   d. For the **Name** field, enter **cfcvip.mycompany.com**
   e. For the **Machine OS** field, select the appropriate operating system.
   f. Click **OK**.
   g. Click the newly created Machine.
   h. Click **Node Manager** tab.
   i. Update Listen Address: **cfcvip.mycompany.com**.
   j. Click **Save**.
   k. Click **Activate Changes**.
3. Stop the WLS_EXMPL Managed server:
   a. Choose **Environment > Servers**.
   b. Click **Control**.
   c. Select **WLS_EXMPL**.
   d. Select **Force Shutdown Now** in the **Shutdown** drop-down menu.
4. Associate the WLS_EXMPL Managed Server with the VirtualHost Machine:
   a. Choose **Environment > Servers**.
   b. Click **Lock & Edit**.
   c. Click **Configuration**.
   d. Select **WLS_EXMPL**.
   e. For Machine, assign the newly created Machine by assigning it from the pull down menu.
   f. For **Listen Address**, enter **cfcvip.mycompany.com**.
   g. Click **Save**.
   h. Click **Activate Changes**.
5. Start the WLS_EXMPL Managed Server:
   a. Choose **Environment > Servers**.
   b. Click **Control**.
   c. Select **WLS_EXMPL**.
   d. Click **Start**.

10.2.2.4.2 Transforming an Oracle WebLogic Managed Server using the WLST Command Line
You can transform an Oracle WebLogic managed server using WLST commands as well.

Oracle recommends shutting down the managed server you are transforming before performing these steps.

To transform a Managed Server using the WLST command line in online mode (with the WebLogic Server administration server up):

1. In the command line, enter:

```
WLS_HOME/server/bin/setWLSEnv.sh
WLS_HOME/common/bin/wlst.sh
```

2. In WLST, enter the following commands:

```
wls:/offline>connect(<username>,<password>,<AdminServer location>)
```

For example:

```
wls:/offline>connect('WebLogic', 'welcome1', 't3://admin.mycompany.com:7001')

wls:/DomainName/serverConfig> edit()
wls:/DomainName/edit> startEdit()
wls:/DomainName/edit !> create('cfcvip.mycompany.com','Machine')
wls:/DomainName/edit !>
cd('Machines/cfcvip.mycompany.com/NodeManager/cfcvip.mycompany.com')
wls:/DomainName/edit !> set('ListenAddress', 'cfcvip.mycompany.com')
wls:/DomainName/edit !>cd ('Servers')
wls:/DomainName/edit/Servers !>cd ('WLS_EXMPL')
wls:/DomainName/edit/Servers/WLS_EXMPL !>set('Machine',' cfcvip.mycompany.com
')
wls:/DomainName/edit/Servers/WLS_EXMPL !>set('ListenAddress','
cfcvip.mycompany.com ')
wls:/DomainName/edit/Servers/WLS_EXMPL !> save()
wls:/DomainName/edit/Servers/WLS_EXMPL !> activate()
wls:/DomainName/edit/Servers/WLS_EXMPL> exit()
```

Stop (if not already down) and start the Managed server.

Once the Managed server transformation is completed, all references to it should use the new Listen Address - cfcvip.mycompany.com. If Oracle HTTP Server serves as a front end to this Managed server, then any mod_wls_ohs configuration with mount points referring to applications in this Managed server should be changed to route to the new listening end point.

10.2.2.5 Transforming Node Manager

Node Manager can be used in a Cold Failover Cluster environment. The possible configurations are:

- Using a Node Manager that listens on the virtual IP as well and fails over with the rest of the Cold Failover Clusters stack. With ASCRS based deployments, the Node Manager must be part of the same Middleware Home as where the Cold Failover Clusters Fusion Middleware instance is running. This Node Manager is assumed to be dedicated for this Fusion Middleware instance. In this case, Oracle recommends having additional Network Channels listening on the localhost configured. Other Node Managers may co-exist on the box listening on other ports. For more details, see Chapter 11, "Using Oracle Cluster Ready Services."
- A Node Manager that does not failover with the rest of the Cold Failover Cluster stack. In this case, Node Manager is not configured for Cold Failover Cluster and listens on all IPs on the machine, and not specifically on the virtual IP for Cold Failover Cluster. The failover nodes also have a similarly configured Node Manager already available and configured. The Machine associated with the WebLogic instance communicates with the Node Manager on the localhost. For more details, see the *Oracle Fusion Middleware Node Manager Administrator's Guide for Oracle WebLogic Server*.

For Cold Failover Cluster in general, port usage should be planned so that there are no port conflicts when failover occurs.

To convert the Node Manager to Cold Failover Clusters:

1. If Node Manager is running, stop it.

   The `nodemanager.properties` file is created only after the first start of nodemanager.

   Restart the node manager if necessary.

2. In the nodemanager.properties file located in the `WLS_HOME`/common/nodemanager/ directory, set the `ListenAddress` to the virtual IP.

   For example:

   ```
   ListenAddress=cfcvip.mycompany.com
   ```

3. Restart the node Manager using the `StartNodeManager.sh` file, located in the *WLS_HOME*/server/bin directory. For ASCRS based deployment, the node manager is started using *WLS_HOME*/server/bin/cfcStartNodemanager.sh.

   See Chapter 11, "Using Oracle Cluster Ready Services." for more details.

   Note: If needed, start the node manager only using the `cfcStartNodemanager.sh` script instead of the `startNodeManager.sh script`.

10.2.2.6 Transforming Oracle Process Management and Notification Server

Oracle Process Management and Notification Server (OPMN) is used for Process Management of system components and is part of the application server instance.

Oracle recommends keeping the default OPMN configuration in a Cold Failover Cluster environment. No further steps are necessary for Cold Failover Cluster transformation of the OPMN process itself.

If you are transforming an Oracle Instance for Cold Failover Clusters and it has already been registered with an administration server, make the following changes in the topology.xml file, located in the *DOMAIN_HOME*/opmn directory of the administration server domain. Change host name entries for this specific Oracle instance (being transformed to Cold Failover Clusters) to `cfcvip.mycompany.com`.

For example, for an Oracle HTTP Server instance transformed to Cold Failover Clusters, set the following in the topology.xml file

```
<property name="HTTPMachine" value="cfcvip.mycompany.com"/>
```

For the instance itself:

```
<ias-instance id="asinst " instance-home="/11gr1as3/MW/asinst"
host="cfcvip.mycompany.com" port="6701">
```

10.2.2.7 Transforming Oracle Enterprise Manager for an Oracle Instance

When an Oracle instance such as Oracle Internet Directory, Oracle Virtual Directory, Web Tier components, and Oracle Portal, Forms, Reports, and Discoverer components is transformed to Cold Failover Clusters, the Enterprise Manager agent that is part of this Oracle instance must be transformed to Cold Failover Clusters as well.

To transform the Enterprise Manager agent:

1. Stop the Enterprise Manager agent using the following command:

```
cd INSTANCE_HOME/EMAGENT/emagent_dir/bin
./emctl stop agent
```

2. Set the directory to *ORACLE_INSTANCE*/EMAGENT/emagent_*instance name*/sysman/config.
3. In the `emd.properties` file, change `node1.mycompany.com` to `cfcvip.mycompany.com` for the `emd_url` attribute.
4. Change the `targets.xml` file on the agent side:

   ```
   cd INSTANCE_HOME/EMAGENT/emagent_dir/sysman/emd
   targets.xml targets.xml.org
   ```

   Modify targets.xml so that it has only targets related to the host and `oracle_emd`. Remove all other entries. For example:

   ```
   <Targets AGENT_TOKEN="ad4e5899e7341bfe8c36ac4459a4d569ddbf03bc">
          <Target TYPE="oracle_emd" NAME=cfcvip.mycompany.com:<port>"/>
          <Target TYPE="host" NAME=cfcvip.mycompany.com  DISPLAY_
   NAME=cfcvip.mycompany.com/>
   </Targets>
   ```

5. Stop and restart the agent

   ```
   cd INSTANCE_HOME/EMAGENT/emagent_dir/bin
   ./emctl start agent
   ```

Make the following changes for the Enterprise Manager server in the administration server domain directory:

1. Set your directory to *MW_HOME*/user_projects/domains/*domain_name*/sysman/state.
2. In the `targets.xml` file, located in *MW_HOME*/user_projects/domains/*domain_name*/sysman/state directory, modify the hostname from `node1.mycompany.com` to `cfcvip.mycompany.com`

10.2.2.8 Transforming Web Tier Components and Clients

The Web tier is made up of two primary components, Oracle HTTP Server and Oracle Web Cache. The next two sections describe how to transform Oracle HTTP Server and Oracle Web Cache for Cold Failover Clusters.

10.2.2.8.1 Transforming Oracle HTTP Server To transform Oracle HTTP Server for Cold Failover Clusters:

In *INSTANCE_HOME*/config/OHS/*component_name*/httpd.conf, change the following attributes

```
Listen cfcvip.mycompany.com:<port> #OHS_LISTEN_PORT
Listen cfcvip.mycompany.com:<port> #OHS_PROXY_PORT
ServerName cfcvip.mycompany.com
```

Clients of Oracle HTTP Server

If an Oracle Web Cache instance is routing to Oracle HTTP Server that has been transformed to Cold Failover Clusters, in *INSTANCE_HOME*/config/WebCache/component_name/webcache.xml, change the following attributes:

Change node1.mycompany.com to cfcvip.mycompany.com, where node1.mycompany.com is the previous address of the Oracle HTTP server before transformation.

```
HOST ID="h1" NAME="cfcvip.mycompany.com" PORT="8888" LOADLIMIT="100"
 OSSTATE="ON"/>
<HOST ID="h2" NAME="cfcvip.mycompany.com" PORT="8890" LOADLIMIT="100" OSSTATE="ON"
 SSLENABLED="SSL"/>
```

10.2.2.8.2 Transforming Oracle Web Cache To transform an Oracle Web Cache for Cold Failover Clusters:

1. Set up an alias to the physical hostname on both nodes of the cluster in /etc/hosts.

   This is an alias to the IP address of the node. Set this in /etc/hosts for Linux and Windows location for Windows. The alias name is wcprfx.mycompany.com For example, On node Node1, the /etc/hosts file (on UNIX), the entry would be **n.n.n.n node1 node1.mycompany.com wcprfx wcprfx.mycompany.com**

   On the failover node Node2, the /etc/hosts file (on UNIX), the entry would be **n.n.n.m node2 node2.mycompany.com wcprfx wcprfx.mycompany.com**.

   On Windows, this change should be done on all nodes in the file located at C:\SystemRoot\system32\drivers\etc\hosts. SystemRoot is either Winnt or Windows.

2. In *INSTANCE_HOME*/config/WebCache/wc1/webcache.xml:

   - Change `node.mycompany.com` to `cfcvip.mycompany.com`node1.mycompany.com is where Oracle Web Cache was installed, and the host address it is listening on before transformation.

     ```
     SITE NAME="cfcvip.mycompany.com"
     ```

   - Change the Virtual Host Name entries to be cfcvip.mycompany.com for the SSL and non-SSL ports. For example:

     ```
     <HOST SSLENABLED="NONE" ISPROXY="NO" OSSTATE="ON" NUMRETRY="5"
      PINGINTERVAL="10" PINGURL="/" LOADLIMIT="100" PORT="8888"
      NAME="cfcvip.mycompany.com" ID="h0"/>
             <HOST SSLENABLED="SSL" ISPROXY="NO" OSSTATE="ON" NUMRETRY="5"
      PINGINTERVAL="10" PINGURL="/" LOADLIMIT="100" PORT="8890"
      NAME="cfcvip.mycompany.com" ID="h3"/>
             <VIRTUALHOSTMAP PORT="8094" NAME="cfcvip.mycompany.com">
                 <HOSTREF HOSTID="h3"/>
             </VIRTUALHOSTMAP>
             <VIRTUALHOSTMAP PORT="8090" NAME="cfcvip.mycompany.com">
                 <HOSTREF HOSTID="h0"/>
             </VIRTUALHOSTMAP>
     ```

   - Change cache name entries to be based of wcprfx.mycompany.com where wcprfx.mycompany.com is an alias created in /etc/hosts on all nodes of the cluster. For example:

     ```
     <CACHE WCDEBUGON="NO" CAPACITY="30" VOTES="1" INSTANCENAME="asinst_1"
      COMPONENTNAME="wc1" ORACLEINSTANCE="/mnt1/ Oracle/Middleware/asinst_1"
      HOSTNAME="wcprfx.mycompany.com" ORACLEHOME="/mnt1/ Oracle/Middleware/as
     _1" NAME=" wcprfx.mycompany.com-WebCache">
     ```

   - In the MULTIPORT section, change IPADDR from **ANY** to **cfcvip.mycompany.com** for the following:

     ```
     PORTTYPE="NORM"
     ```

```
SSLENABLED="SSL" PORTTYPE="NORM"
PORTTYPE="ADMINISTRATION"
PORTTYPE="INVALIDATION"
PORTTYPE="STATISTICS"
```

For example:

```
 <MULTIPORT>
            <LISTEN PORTTYPE="NORM" PORT="8090"
 IPADDR="stbdd01-vip.us.oracle.com"/>
            <LISTEN SSLENABLED="SSL" PORTTYPE="NORM" PORT="8094"
 IPADDR="cfcvip.mycompany.com">
              <WALLET>/mnt1/Oracle/Middleware/asinst_
1/config/WebCache/wc1/keystores/default</WALLET>
            </LISTEN>
            <LISTEN PORTTYPE="ADMINISTRATION" PORT="8091"
 IPADDR="cfcvip.mycompany.com"/>
            <LISTEN PORTTYPE="INVALIDATION" PORT="8093" IPADDR="
 cfcvip.mycompany.com"/>
            <LISTEN PORTTYPE="STATISTICS" PORT="8092"
 IPADDR="cfcvip.mycompany.com"/>
        </MULTIPORT>
```

10.2.2.9 Instance-specific considerations

In a Cold Failover Clusters environment, a failover node (node2.mycompany.com) must be equivalent to the install machine (node1.mycompany.com) in all respects. To make the failover node equivalent to the installation node, perform the following procedure on the failover instance:

10.2.2.9.1 UNIX Platforms For UNIX platforms follow these steps:

1. Failover the Middleware Home from Node 1 (the installation node) to the failover node (Node 2). This should be done manually following the mount/unmount procedure described earlier.
2. As root, do the following:
   - Create an `oraInst.loc` file located in the file `/etc` directory identical to the one on Node1.
   - Run the `oracleRoot.sh` file located in the *ORACLE_HOME* directory node2, if required, and available for the product suite.
3. Create the `oraInventory` on the second node, by using the `attachHome` command located in *ORACLE_HOME*/oui/bin/attachHome.sh

10.2.2.9.2 Windows Platform For Windows Platforms, follow these steps:

For AS instances (Web Tier, OID/OVD for IDM installs, Oracle Portal, Form, Reports, and Discoverer):

1. To create the OPMN service on Machine 2 run the following commands:

   ```
   sc create OracleProcessManager_instance_name  binPath= "ORACLE_
   HOME\opmn\bin\opmn.exe -S -I Instance_Home"
   ```

   For example:

   ```
   sc create OracleProcessManager_asinst binPath= "X:\Middleware\im_
   oh\opmn\bin\opmn.exe -S -I X:\Middleware\asinst"
   ```

2. On both Machine 1 and 2, set the service OracleProcessManager_*instance_name* to be started manually.

   ```
   sc config OracleProcessManager_instance_name start= demand
   ```

For all installations:

To copy the Start Menu from Machine 1 to Machine 2, copy the files under `C:\Document and Settings\All Users\Start Menu\Programs\<Menu>` from Machine 1 to Machine 2. You can do this with any Windows Backup tool, or Zip utility. For example, using the Windows Backup Utility:

1. Invoke the Windows Backup Utility by selecting **Accessories**, **System Tools**, and then **Backup**.
2. Select **C:\Document and Settings\All Users\Start Menu\Programs\<Menu>**
3. Copy the backup file to the second node.
4. Restore the backup to the same location: `C:\Document and Settings\All Users\Start Menu\Programs\`

For Node Manager, use the standard WebLogic Server procedure to configure Node Manager as a service.

Node Manager is configured as a service using the `WL_HOME\server\bin\installNodeMgrSvc.cmd` command. To configure Node Manager:

- It must be set on both Machines.
- Node Manager should be set to start manually on both nodes.

To configure Node Manager on a Machine,

1. Ensure that the Middleware Home is available on the machine.
2. Install the service running the above command.
3. Set it to be started manually using the following command:

   ```
   sc config nodemanager_service_name start= demand
   ```

10.2.3 Transforming Oracle Fusion Middleware Components

This section describes the considerations and the transformation steps for each of the Oracle product suites. For detailed explanation on the product components, see the appropriate component chapter in this guide.

10.2.3.1 Transforming Oracle Internet Directory and Its Clients

This section describes how to transform Oracle Internet Directory and its clients.

10.2.3.1.1 Transforming Oracle Internet Directory Follow these steps to transform an Oracle Internet Directory server:

1. In a text editor, create a file named `oidcfc.ldif` that sets up the virtual IP `cfcvip.mycompany.com` for the Oracle Internet Directory server:

   ```
   dn: cn=oid1,cn=osdldapd,cn=subconfigsubentry
   changetype: modify
   replace: orclhostname
   orclhostname: cfcvip.mycompany.com
   ```

2. Run the following command:

```
ORACLE_HOME/bin/ldapmodify -p <oidPort> -h <oidHost> -D cn=orcladmin -w
<adminPasswd> -f oidcfc.ldif
```

   OID must be up at the time of running this command. OidHost used in the above command is the physical hostname listening end point that OID was installed with.

3. Stop and restart the Oracle Internet Directory server using opmnctl:

   For example:

```
ORACLE_INSTANCE/bin/opmnctl stopproc ias-component=oid1
ORACLE_INSTANCE/bin/opmnctl startproc ias-component=oid1
```

4. Re-Register OID with the Admin Server

```
ORACLE_INSTANCE/bin/opmnctl updatecomponentregistration -adminHost
 myAdminHost -adminPort 7001 -adminUsername weblogic -componentType OID
 -componentName oid1 -Host cfcvip.mycompany.com
```

10.2.3.1.2 Transforming Oracle Internet Directory Clients

To transform an Oracle Internet Directory client:

1. All clients of the Oracle Internet Directory server should use the virtual IP cfcvip.mycompany.com to access that Oracle Internet Directory server.
2. Oracle Directory Integration Platform installation that uses the Oracle Internet Directory server must access the Oracle Internet Directory server using the virtual IP. To do this:
   a. In a text editor, open the `dip-config.xml` file located in the *DOMAIN_HOME*/servers/wls_ods1/stage/DIP/11.1.1.1.0/DIP/configuration directory.

      For example:

```
MW_HOME/user_projects/domains/IDMDomain/servers/wls_ods1/stage/DIP/
11.1.1.1.0/DIP/configuration/dip-config.xml
```

   b. Enter the following value to set the LDAP address to the virtual IP:

```
OID_NODE_HOST>cfcvip.mycompany.com</OID_NODE_HOST>
```

3. An Oracle Directory Services Manager instance managing the Oracle Internet Directory server in Section 10.2.2.8.1, "Transforming Oracle HTTP Server" must use the virtual IP to connect to the Oracle Internet Directory server.

   For example, from the Oracle Directory Services Manager screen, verify that you can connect to Oracle Internet Directory using the virtual server by following these steps:

   a. Select the **Connect to a directory --> Create A New Connection** link in the upper right hand corner.
   b. In the New Connection screen, fill in the connection information below and click **Connect**:
      * **Directory Type**: `OID`
      * **Name**: `OIDHA`
      * **Server**: `cfcvip.mycompany.com`

  * **Port**: `389`
  * **SSL Enabled**: Leave blank
  * **User Name**: `cn=orcladmin`
  * **Password**: `********`
  * **Start Page**: `Home` (default)

10.2.3.2 Transforming Oracle Virtual Directory and Its Clients

This section describes how to transform Oracle Virtual Directory and its clients.

10.2.3.2.1 Transforming Oracle Virtual Directory Follow these steps to transform an Oracle Virtual Directory server:

1. In a text editor, open the `listeners.os_xml` file in the *ORACLE_INSTANCE*/config/OVD/*componentname* directory.
2. Enter the following value to set the LDAP address to the virtual IP:

   ```
   <host>cfcvip.mycompany.com</host>
   ```

3. Restart the Oracle Virtual Directory server using opmnctl.

   For example:

   ```
   ORACLE_INSTANCE/bin/opmnctl stopproc ias-component=ovd1
   ORACLE_INSTANCE/bin/opmnctl startproc ias-component=ovd1
   ```

10.2.3.2.2 Transforming Oracle Virtual Directory Clients All clients of Oracle Virtual Directory must use the virtual IP `cfcvip.mycompany.com` to access Oracle Virtual Directory. For example, when using Oracle Directory Services Manager to administer a Cold Failover Cluster Oracle Virtual Directory instance, create a connection using `cfcvip.mycompany.com` as the location of the Oracle Virtual Directory instance.

10.2.3.3 Transforming Oracle Directory Integration Platform and Oracle Directory Services Manager and Their Clients

This section describes how to transform Oracle Directory Integration Platform, Oracle Directory Services Manager, and their clients.

10.2.3.3.1 Transforming Oracle Directory Integration Platform and Oracle Directory Services Manager Oracle Directory Integration Platform and Oracle Directory Services Manager are deployed to a Managed Server. The procedure for CFC transformation is to configure the Managed Server to which they are deployed to listen on the `cfcvip.mycompany.com` virtual IP. Follow the steps in Section 10.2.2.4, "Transforming Oracle WebLogic Managed Servers" to configure the WLS_ODS managed server to listen on the `cfcvip.mycompany.com` virtual IP.

10.2.3.3.2 Transforming Oracle Directory Integration Platform and Oracle Directory Services Manager Clients Follow these steps to transform Oracle Directory Integration Platform and Oracle Directory Services Manager clients:

1. Clients of Oracle Directory Integration Platform and Oracle Directory Services Manager must use the virtual IP `cfcvip.mycompany.com` to access these applications.
2. When Oracle HTTP Server is the front end for Oracle Directory Services Manager, the WebLogic configuration for Oracle Directory Services Manager must specify

the virtual IP `cfcvip.mycompany.com` as the address for the WLS_ODS Managed Server. To do this, open the mod_wl_ohs file in a text editor and make these edits for the mount points used by Oracle HTTP Server and Oracle Directory Services Manager:

```
#Oracle Directory Services Manager
<Location /odsm>
SetHandler weblogic-handler
WebLogicHost  cfcvip.mycompany.com:<port>
</Location>
```

10.2.3.4 Transforming Oracle Identity Federation and Its Client

This section describes how to transform Oracle Identity Federation and its clients.

10.2.3.4.1 Transforming Oracle Identity Federation Oracle Identity Federation is a component that is deployed to a Managed Server. The procedure for Cold Failover Clusters transformation is to configure the Managed Server to which it is deployed to listen on the `cfcvip.mycompany.com` virtual IP. Follow the steps in Section 10.2.2.4, "Transforming Oracle WebLogic Managed Servers" to configure the WLS_OIF Managed Server to listen on the `cfcvip.mycompany.com` virtual IP. Since Oracle Identity Federation Cold Failover Clusters deployments are likely to be split into Service Provider and Identity Provider, more than one instance of WLS_OIF is likely to exist in a given deployment. Use the same Cold Failover Clusters procedure for both WLS_OIF instances.

After configuring the Managed Server to listen on the `cfcvip.mycompany.com` virtual IP, log into the Oracle Enterprise Manager Fusion Middleware Control and perform these steps:

1. Navigate to **Farm > Identity and Access > OIF**.
2. In the right frame, navigate to **Oracle Identity Federation > Administration** and then make these changes:
   a. **Server Properties**: change the host to `cfcvip.mycompany.com`
   b. **Identity Provider > Common**: change the **providerId** to `cfcvip.mycompany.com`
   c. **Service Provider > Common**: change the **providerId** to `cfcvip.mycompany.com`
   d. **Data Stores**: If LDAP is the data store, then replace the value of **Connection URL for User Data Store and Federation Data Store** with `cfcvip.mycompany.com`
   e. **Authentication Engine > LDAP Directory**: Set the **ConnectionURL** to `cfcvip.mycompany.com`

The metadata needs to be generated after the above changes are done.

10.2.3.4.2 Transforming Oracle Identity Federation Clients Follow these steps to transform Oracle Identity Federation clients:

1. Clients of Oracle Identity Federation must use the virtual IP `cfcvip.mycompany.com` to access these applications.
2. When Oracle HTTP Server is the front end for Oracle Identity Federation, the WebLogic configuration for Oracle Directory Services Manager must specify the virtual IP `cfcvip.mycompany.com` as the address for the WLS_OIF Managed

Server. To do this, open the mod_wl_ohs file in a text editor and make these edits for the mount points used by Oracle HTTP Server and Oracle Directory Services Manager:

```
#Oracle Identity Federation
<Location /oif>
SetHandler weblogic-handler
WebLogicHost  cfcvip.mycompany.com:<port>
</Location>
```

10.2.3.5 Transforming an Oracle SOA Suite

Oracle SOA Suite is made up of Java EE components deployed to a managed server. The typical configuration has deployments of OWSM-PM applications and SOA applications. The SOA applications are always deployed to a separate managed server. In many Cold Failover Clusters deployments OWSM-PM is deployed to the Administration Server, however, they can be deployed to a managed server by itself, for example, WLS_OWSM, or to the SOA managed server. In all cases, since these are Java EE components, the Cold Failover Clusters transformation procedure involves configuring the managed server to which they are deployed to listen on the virtual IP. See Section 10.2.2.4, "Transforming Oracle WebLogic Managed Servers" for information on transforming managed servers WLS_OWSM and WLS_SOA.

In addition, follow these steps to transform a SOA component managed server:

1. Set the front-end host for WLS_SOA managed server to cfcvip.mycompany.com.
   a. Log into Oracle WebLogic Server Administration Console.
   b. In the **Environment** section, select **Servers**.
   c. Select the name of the managed server.
   d. Select **Protocols**, then select **HTTP**.
   e. In the **Frontend Host** field, enter the host name as **cfcvip.mycompany.com**.
   f. Set the Frontend Port to the HTTP port
   g. Click **Save**.
   h. Activate the changes.
   i. Restart the Managed Server.
2. When using Oracle HTTP Server as the front end, the mod WebLogic configuration for the applications deployed to WLS_OWSM and WLS_SOA should provide the VIP cfcvip.mycompany.com as the address of these managed server. This configuration change is done in mod_wl_ohs.conf for the mount points used by SOA components. For example:

```
#SOA soa-infra app
<Location /soa-infra>
    SetHandler weblogic-handler
    WebLogicHost cfcvip.mycompany.com:<port>
</Location>

# Worklist
<Location /integration/>
    SetHandler weblogic-handler
    WebLogicHost cfcvip.mycompany.com:<port>
</Location>
```

```
# B2B
<Location /b2b>
    SetHandler weblogic-handler
    WebLogicHost cfcvip.mycompany.com:<port>
</Location>

# UMS prefs
<Location /sdpmessaging/userprefs-ui >
    SetHandler weblogic-handler
    WebLogicHost cfcvip.mycompany.com:<port>
</Location>

# WSM
<Location /wsm-pm>
          SetHandler weblogic-handler
    WebLogicHost cfcvip.mycompany.com:<port>
</Location>

# workflow
<Location /workflow>
    SetHandler weblogic-handler
    WebLogicHost cfcvip.mycompany.com:<port>
</Location>
)
```

10.2.3.6 Transforming an Oracle WebCenter Suite

The Web Center Suite is made up of Java EE components. The typical WebCenter Suite deployment has three managed servers, though more are possible when you have WebCenter custom applications. Typical managed server deployments include:

- WLS_SPACES
- WLS_PORTLETS
- WLS_SERVICES

These are Java EE components, therefore, the procedure for Cold Failover Clusters transformation is to configure the Managed Server to which they are deployed to listen on the Virtual IP. In many Cold Failover Cluster deployments of Oracle WebCenter, OWSM-PM may be deployed to WLS_Portlets and WLS_Spaces container. See Section 10.2.2.4, "Transforming Oracle WebLogic Managed Servers" for information on transforming Oracle WebCenter Suite managed servers.

In addition, when you use Oracle HTTP Server as the front end, the mod_weblogic configuration for WebCenter Suite applications the managed servers should provide the VIP cfcvip.mycompany.com as the address of these managed servers. This configuration change is done in mod_wl_ohs.conf for the mount points used by WebCenter components. For example:

```
LoadModule weblogic_module "${ORACLE_HOME}/ohs/modules/mod_wl_ohs.so"

# Spaces

<Location /webcenter>
          SetHandler weblogic-handler
    WebLogicHost cfcvip.mycompany.com:<port>
</Location>

<Location /webcenterhelp>
          SetHandler weblogic-handler
    WebLogicHost cfcvip.mycompany.com:<port>
```

```
</Location>

<Location /rss>
          SetHandler weblogic-handler
    WebLogicHost cfcvip.mycompany.com:<port>
</Location>

# Portlet

<Location /portalTools>
          SetHandler weblogic-handler
    WebLogicHost cfcvip.mycompany.com:<port>
</Location>

<Location /wsrp-tools>
          SetHandler weblogic-handler
    WebLogicHost cfcvip.mycompany.com:<port>
</Location>

# WSM

<Location /wsm-pm>
          SetHandler weblogic-handler
    WebLogicHost cfcvip.mycompany.com:<port>
</Location>

# Discussions and Wiki

<Location /owc_discussions>
          SetHandler weblogic-handler
    WebLogicHost cfcvip.mycompany.com:<port>
</Location>

<Location /owc_wiki>
          SetHandler weblogic-handler
    WebLogicHost cfcvip.mycompany.com:<port>
</Location>
```

10.2.3.7 Transforming Oracle Portal, Forms, Reports, and Discoverer

is made up of these four separate components. Cold Failover Clusters configuration varies for each Oracle Portal, Forms, Reports, and Discoverer component. This section documents the steps for Cold Failover Clusters transformation of Oracle Portal, Forms, Reports, and Discoverer.

10.2.3.7.1 Transforming Oracle Forms for Cold Failover Clusters Oracle Forms is made up of both Java EE and system components. To configure Oracle Forms for Cold Failover Clusters:

Note: The transformation process requires the domain be shutdown before editing these files.

1. Transform the WLS_FORMS managed server. See Section 10.2.2.4, "Transforming Oracle WebLogic Managed Servers" for this procedure. Start the Domain administration server to do the Managed server transformation, and shut it down once the Managed Server transformation is finished.

2. Transform the Oracle Forms OPMN. See Section 10.2.2.6, "Transforming Oracle Process Management and Notification Server" for this procedure.
3. Transform Enterprise Manager. See Section 10.2.2.7, "Transforming Oracle Enterprise Manager for an Oracle Instance" for this procedure.
4. Transform Oracle Forms Web tier components. See Section 10.2.2.8, "Transforming Web Tier Components and Clients" for this procedure.
5. Reregister Single Sign-On using the steps described in the Section 12.6.4.8.4, "Enable Single Sign On."
6. In the forms.conf file located in the *INSTANCE_HOME*/config/OHS/ohs1/moduleconf directory, change the mod weblogic configuration.

   Start the administration server and Managed Servers in the WebLogic Domain as well as the Oracle instances to validate the Cold Failover Cluster installation:

   ```
   <Location /Forms>
       SetHandler weblogic-handler
       WebLogicHost cfcvip.mycompany.com:<port>
       DynamicServerList OFF
   </Location>
   ```

10.2.3.7.2 Transforming Oracle Reports for Cold Failover Clusters

Oracle Reports is made up of both Java EE and system components. To configure Oracle Reports for Cold Failover Clusters:

1. Transform the WLS_REPORTS managed server. See Section 10.2.2.4, "Transforming Oracle WebLogic Managed Servers" for this procedure. Start the Domain administration server to do the Managed Server transformation, and shut it down once the Managed Server transformation is finished.
2. Transform the Oracle Reports OPMN. See Section 10.2.2.6, "Transforming Oracle Process Management and Notification Server" for this procedure

   In addition, do the following:

   a. In the `opmn.xml` file, located in the *INSTANCE_HOME*/config/OPMN/opmn, directory, change the Reports server name in `ias-component` to reflect its new name.

      For example:

      ```
      <ias-component id="ReportsServer_virtual_hostname_instance_name">
      <process-set id="ReportsServer_virtual_hostname_instance_name"
       restart-on-death="true" numprocs="1">
      </ias-component>
      ```

      For example:

      ```
      <ias-component id="ReportsServer_cfcvip_asinst1">
         <process-set id="ReportsServer_ cfcvip asinst1"restart-on-death="true"
         numprocs="1">
      </ias-component>
      ```

   b. In *DOMAIN_HOME*/opmn/topology.xml of the administration server domain home, change all occurrences of the Report server name from ReportsServer_physical hostname_instance_name (example: ReportServer_node1_asinst) to ReportsServer_virtual_hostname_instance_name (example: ReportServer_cfcvip_asinst)

3. Transform Oracle Enterprise Manager. See Section 10.2.2.7, "Transforming Oracle Enterprise Manager for an Oracle Instance" for this procedure.
4. Transform Oracle Reports Web tier components. See Section 10.2.2.8, "Transforming Web Tier Components and Clients" for this procedure.
5. Reregister Single Sign-On. To do this perform the steps described in the Section 12.6.4.8.4, "Enable Single Sign On."
6. In the reports_ohs.conf file located in the *INSTANCE_HOME*/config/OHS/ohs1/moduleconf directory, change the following:

```
<Location /reports>
    SetHandler weblogic-handler
    WebLogicHost cfcvip.mycompany.com:port
    WebLogicPort 9001
</Location>
```

7. In the directory *INSTANCE_HOME*/config/ReportsServerComponent, rename the following directory:

   ```
   ReportsServer_physical_hostname_instance_name (for example: ReportServer_node1_asinst)
   ```

   To the following:

   ```
   ReportsServer_VIP_instance_name (for example: ReportServer_cfcvip_asinst)
   ```

   Note: All new logs for the report server go to ReportsServer_*Virtual_Hostname_Instance_Name* after restart of the instance.

8. In the renamed directory, replace physical hostname with a virtual hostname in the following files:
   - component-logs.xml
   - logging.xml
9. In the `targets.xml` file, located in the *INSTANCE_HOME*/EMAGENT/emagent_dir/sysman/emd directory, change the following:
   a. Change all hostnames from `node1.mycompany.com` to `cfcvip.mycompany.com`
   b. Change the report server name from `ReportsServer_physical_hostname_instance_name` (for example: `ReportServer_node1_asinst`) to `ReportsServer_VIP_instance_name` (for example: `ReportServer_cfcvip_asinst`) for the following two elements:

      Target TYPE="oracle_repserv"

      Target TYPE="oracle_repapp"
10. In the *INSTANCE_HOME*/reports directory, replace the physical hostname with a virtual hostname in reports_install.properties.
11. In the *INSTANCE_HOME*/reports/server directory, rename the following files:
    - Rename reportsserver_*physical hostname_instance name*.dat to reportsserver_*virtual host name_instance name*.dat

- Rename rep_wls_reports_*physical hostname_instance name*.dat to rep_wls_reports_*virtual hostname_instance name*.dat
- Start the administration server and Managed Servers in the WebLogic Domain, as well as the Oracle instances to validate the Cold Failover Cluster installation.

12. In the *DOMAIN_HOME*/servers/WLS_REPORTS/stage/reports/reports/configuration directory, replace physical hostname with virtual hostname in the rwservlet.properties file (including changes to any occurrence of the reports sever name).
13. In the *DOMAIN_HOME*/servers/WLS_REPORTS/stage/reports/reports/META-INF configuration directory, replace physical hostname with virtual hostname in the mbeans.xml file (including changes to any occurrence of the reports sever name).

10.2.3.7.3 Transforming Oracle Discoverer for Cold Failover Clusters Oracle Discoverer is made up of both Java EE and system components. To configure Oracle Discoverer for Cold Failover Clusters:

1. Transform the WLS_DISCO managed server. See Section 10.2.2.4, "Transforming Oracle WebLogic Managed Servers" for this procedure. Start the domain administration server to do the Managed Server transformation, and shut it down once the Managed Server transformation is finished.
2. Transform the Oracle Discoverer OPMN. See Section 10.2.2.6, "Transforming Oracle Process Management and Notification Server" for this procedure.
3. Transform Oracle Enterprise Manager. See Section 10.2.2.7, "Transforming Oracle Enterprise Manager for an Oracle Instance" for this procedure.
4. Transform Oracle Discoverer Web tier components. See Section 10.2.2.8, "Transforming Web Tier Components and Clients" for this procedure.
5. Reregister Single Sign-On. To do this perform the steps described in the Section 12.6.4.8.4, "Enable Single Sign On."
6. In the reports_ohs.conf file located in the *INSTANCE_HOME*/config/OHS/ohs1/moduleconf directory, change the following:

```
<IfModule mod_weblogic.c>
<Location /discoverer>
SetHandler weblogic-handler
WebLogicCluster cfcvip.mycompany.com:<port>
DynamicServerList ON
</Location>
</IfModule>
 >
```

7. In configuration.xml file located in the DOMAIN_HOME/servers/WLS_DISCO/stage/discoverer/11.1.1.1.0/discoverer/configuration/ directory, change the following:

```
applicationURL="http://cfcvip.mycompany.com:8090/discoverer">
```

   The port here in the applicationURL above is the Oracle HTTP Server port (assuming the Oracle HTTP Server has also been transformed to Cold Failover Clusters.)

Start the administration server and Managed Servers in the WebLogic Domain, as well as the Oracle Instances to validate the Cold Failover Cluster installation.

8.

10.2.3.7.4 Transforming Oracle Portal for Cold Failover Clusters

Oracle Portal is made up of both Java EE and system components. To configure Oracle Portal for Cold Failover Clusters:

1. Transform the WLS_PORTAL managed server. See Section 10.2.2.4, "Transforming Oracle WebLogic Managed Servers" for this procedure. Start the domain administration server to do the Managed Server transformation, and shut it down once the Managed Server transformation is finished.
2. Transform the Oracle Portal OPMN. See Section 10.2.2.6, "Transforming Oracle Process Management and Notification Server" for this procedure.
3. Transform Oracle Enterprise Manager. See Section 10.2.2.7, "Transforming Oracle Enterprise Manager for an Oracle Instance" for this procedure.
4. Transform Oracle Portal Web tier components. See Section 10.2.2.8, "Transforming Web Tier Components and Clients" for this procedure.

   Additionally, reset the invalidation and Administrator password:

   a. Login to the Oracle Enterprise Manager Fusion Middleware Console and in the navigator window, expand the **Web Tier** tree.
   b. Click on the Web Cache component, for example, **wc1**.
   c. From the drop-down list at the top of the page, select **Administration** and then **Passwords**:

      Enter a new invalidation password, confirm it, and click **Apply**.

      Enter a new administrator password, confirm it, and click **Apply**.
5. Reregister Single Sign-On. To do this, perform the steps described in the Section 12.6.4.8.4, "Enable Single Sign On."
6. In the portal.conf file located in the INSTANCE_HOME/config/config/OHS/ohs1/moduleconf, change the mod weblogic configuration, change the mod weblogic configuration:

```
# WLS routing configuration
<Location /portal>
SetHandler weblogic-handler
WebLogicHost cfcvip.mycompany.com
WebLogicPort <port>
</Location>

<Location /portalTools>
SetHandler weblogic-handler
WebLogicHost cfcvip.mycompany.com
WebLogicPort <port>
</Location>

<Location /wsrp-tools>
SetHandler weblogic-handler
WebLogicHost cfcvip.mycompany.com
WebLogicPort <port>
</Location>

<Location /richtextportlet>
```

```
SetHandler weblogic-handler
WebLogicHost cfcvip.mycompany.com
WebLogicPort <port>
</Location>

<Location /jpdk>
SetHandler weblogic-handler
WebLogicHost cfcvip.mycompany.com
WebLogicPort <port>
</Location>

<Location /portalHelp>
SetHandler weblogic-handler
WebLogicHost cfcvip.mycompany.com
WebLogicPort <port>
</Location>

<Location /portalHelp2>
SetHandler weblogic-handler
WebLogicHost cfcvip.mycompany.com
WebLogicPort <port>
</Location>
```

7. Rewire the Portal Repository:

   a. Log into the domain WebLogic Server Enterprise Manager using the following URL:

      http://cfcvip.mycompany.com:7001/em

   b. Change `InValidation` and Administrator password.

      In the Navigator Window Expand the **Web Tier** tree.

      Click the component **wc1**.

      From the drop-down list at the top of the page, select **Administration - Passwords**.

      Enter a new invalidation password, confirm it, and click **Apply**.

      Enter a new administrator password, confirm it and click **Apply**.

   c. Expand the **Fusion Middleware** menu in the left pane.

   d. Select **Classic**.

   e. Click **Portal**

      The **Portal Domain** information page appears.

   f. Right-click on **Portal** and select **Settings**, and then **Wire Configuration**.

   g. Enter the following information for **Portal Midtier**

      Host: Enter the Cold Failover Clusters Virtual IP name of the Web Cache host cfcvip.mycompany.com.

      Port: Enter the Web Cache port being used (HTTP or non HTTP)

      SSL Protocol: Enter this value if appropriate.

   h. Enter the following information for **Web Cache**:

      Host: Enter the Cold Failover Clusters Virtual IP name of the Web Cache host cfcvip.mycompany.com mysite.mycompany.com

Invalidation port: Enter the Invalidation port, for example, **9401**.

Invalidation User Name: Enter the user name for Portal invalidations.

Invalidation Password: Enter the password for this account.

i. Click **Apply** to start the rewire.

j. When the rewire is complete, click the **Portal** menu option again, and ensure that the Portal URL is now the following:

https://cfcvip.mycompany.com:*WCHTTPPort*/portal/pls/portal

8. Change Host Assertion in Oracle WebLogic Server.

   Because the Oracle HTTP Server acts as a proxy for WebLogic, by default certain CGI environment variables, including the host and port, are not passed through to WebLogic. TO ensure that Web Logic is ware that it is using a virtual site name and port so that it can generate internal URLs appropriately:

   a. Log in to the WebLogic Server Administration Console using the following URL:

      http://cfcvip.mycompany.com:7001/console

   b. Select **WLS_PORTAL** from the home page or select **Environment**, and then **Clusters** from the **Domain Structure** menu.

   c. Click **Lock & Edit** in the **Change Center** window to enable editing.

   d. Click **Protocol**, and select **HTTP**.

   e. Enter the following values:

      Frontend Host: **Cfcvip.mycompany.com**

      Frontend HTTP Port: **WCHTTPPort (8090)**

      Frontend HTTPS Port: **WCHTTPSPort (8094)**

      This ensures that any HTTPS URLs created from within WebLogic are directed to port 443 on the load balancer.

   f. Click **Activate Changes** in the **Change Center** window to enable editing.

   g. Restart the WLS_PORTAL managed server

10.2.3.7.5 Transforming Oracle Business Activity Management (BAM) Oracle BAM is made up of Java EE components deployed to a managed server. Since these are Java EE components, the Cold Failover Clusters transformation procedure involves configuring the managed server to which they are deployed to listen on the virtual IP. See Section 10.2.2.4, "Transforming Oracle WebLogic Managed Servers" for information on transforming managed servers WLS_BAM.

When using Oracle HTTP Server as the front end, the mod WebLogic configuration for the applications deployed to WLS_BAM should provide the VIP cfcvip.mycompany.com as the address of these managed server. This configuration change is done in the `mod_wl_ohs.conf` file for the mount points used by SOA components. For example:

```
# BAM Web Application
<Location /OracleBAM >
    SetHandler weblogic-handler
WebLogicHost  cfcvip.mycompany.com:<port>
</Location>
```

10.2.3.7.6 Transforming a Custom ADF Deployment

For a deployment that uses a custom ADF application, cold failover clusters can be used in the same way as any of the Fusions Middleware deployment. The domain is created in this case using the installation from Oracle Application Developer DVD. Since this is primarily a Java EE components. the Cold Failover Clusters transformation procedure involves configuring the managed server to which they are deployed to listen on the virtual IP. See Section 10.2.2.4, "Transforming Oracle WebLogic Managed Servers" for information on transforming managed servers.

When using Oracle HTTP Server as the front end, the mod WebLogic configuration for the applications deployed to WLS_ADF (the name of the managed server for the customer app) should provide the VIP cfcvip.mycompany.com as the address of these managed server. This configuration change is done in mod_wl_ohs.conf for the mount points used by SOA components. For example:

```
# BAM Web Application
<Location /ADFApplicationMountPoint >
    SetHandler weblogic-handler
WebLogicHost  cfcvip.mycompany.com:<port>
</Location>
```

10.2.3.8 Single Sign-On Re-registration (If required)

Single Sign-On (SSO) re-registration typically applies only to Oracle Portal, Forms, Reports, and Discoverer. Once the front end listening endpoint on Oracle HTTP Server for this tier has been changed to the Virtual IP, it becomes necessary to do SSO re-registration so that the URL to be protected is configured with the virtual IP. To re-register SSO, perform these steps on the 10.1.x installation of Identity Management where the SSO server resides:

1. Set the *ORACLE_HOME* variable to the SSO ORACLE_HOME location.
2. Execute *ORACLE_HOME*/sso/bin/ssoreg.sh (ssoreg.bat for Windows) with the following parameters:

   ```
   -site_name cfcvip.mycompany.com:port
   -mod_osso_url http://cfcvip.mycompany.com
   -config_mod_osso TRUE
   -oracle_home_path ORACLE_HOME
   -config_file /tmp/osso.conf
   -admin_info cn=orcladmin
   -virtualhost
   -remote_midtier
   ```

3. Set the ORACLE_HOME variable to the SSO ORACLE_HOME location.
4. Copy /tmp/osso.conf file to the Discoverer mid-tier home location:

   ORACLE_INSTANCE/config/OHS/ohs1
5. Restart Oracle HTTP Server by issuing the following command from the *ORACLE_HOME*/opm/bin/directory:

   ```
   opmnctl restartproc process-type=OHS
   ```

6. Log in to the SSO server through the following URL:

   ```
   http://login.mycompany.com/pls/orasso
   ```

7. In the **Administration** page and then **Administer Partner** applications, delete the entry for **node1.mycompany.com**.

10.2.4 Transforming an Oracle Database

In an Oracle Fusion Middleware deployment, the Oracle database plays an important role. In a typical Cold Failover Clusters deployment of Oracle Fusion Middleware, the database is also deployed as a cold failover cluster. This section described the steps to transform a single instance Oracle database to a Cold Failover Cluster database. This transformation should be done before seeding the database using RCU and subsequent Fusion Middleware installations that use this seeded database. To enable the database for Cold Failover Clusters:

1. Change the listener configuration used by the database instance.

   This requires changes to the listener.ora file. Ensure that the HOST name in the listener configuration has the value of the virtual hostname. In addition, ensure that the listener port is not in use by any other process (Oracle or third party).

   ```
   <listener_name>  =
    (DESCRIPTION_LIST =
      (DESCRIPTION =
            (ADDRESS = (PROTOCOL = TCP)(HOST = <virtual hostname>)(PORT = <port>))
      )
   )
   ```
   For example:

   ```
   LISTENER_CFCDB =
    (DESCRIPTION_LIST =
      (DESCRIPTION =
            (ADDRESS = (PROTOCOL = TCP)(HOST cfcdbhost.mycompany.com)(PORT =
   1521))
      )
    )
   ```

2. Change the tnsnames.ora file.

   Change an existing tns service alias entry or create a new one:

   ```
   <tns alias name> =
    (DESCRIPTION =
      (ADDRESS_LIST =
        (ADDRESS = (PROTOCOL = TCP)(HOST = <virtual hostname>)(PORT = <port>))
      )
      (CONNECT_DATA =
         (SERVER = DEDICATED)
         (SERVICE_NAME = <db service name>)
         (INSTANCE_NAME = <db instance name>)
      )
    )

   )
   ```
   For example:

   ```
   CFCDB =
    (DESCRIPTION =
      (ADDRESS_LIST =
         (ADDRESS = (PROTOCOL = TCP)(HOST = cfcdbhost.mycompany.com)(PORT = 1521))
      )
      (CONNECT_DATA =
         (SERVER = DEDICATED)
         (SERVICE_NAME = cfcdb)
         (INSTANCE_NAME = cfcdb)
      )
    )
   ```

3. Change the local sp file to update the local_listener parameter of the instance.

   Login as sysdba using sqlplus:

   ```
   SQL> alter system set local_listener='<tns alias name>' scope=both;
   )
   ```
   For example:

   ```
   SQL> alter system set local_listener='CFCDB' scope=both;
   ```

4. Shutdown and restart the listener.
5. Shutdown and restart the database instance.
6. Create the database service for the application server.

   oracle recommends a dedicated service separate from the default database service used with the Oracle Application server. To create this service, execute the following sqlplus command:

   ```
   SQL> execute DBMS_SERVICE.CREATE_SERVICE(
      '<cfc db service name>'   '<cfc db network name>' ) \
   ```

   For example:

   ```
   SQL> execute DBMS_SERVICE.CREATE_SERVICE(
      'cfcdb_asservice'   'cfcdb_asservice' )

   SQL> execute DBMS_SERVICE.START_SERVICE(   'cfcdb_asservice' )
   ```

   Additional parameters for this service may be set depending on the needs of the installation. See the *Oracle Database PL/SQL Packages and Types Reference* for details about the DBMS_SERVICE command.

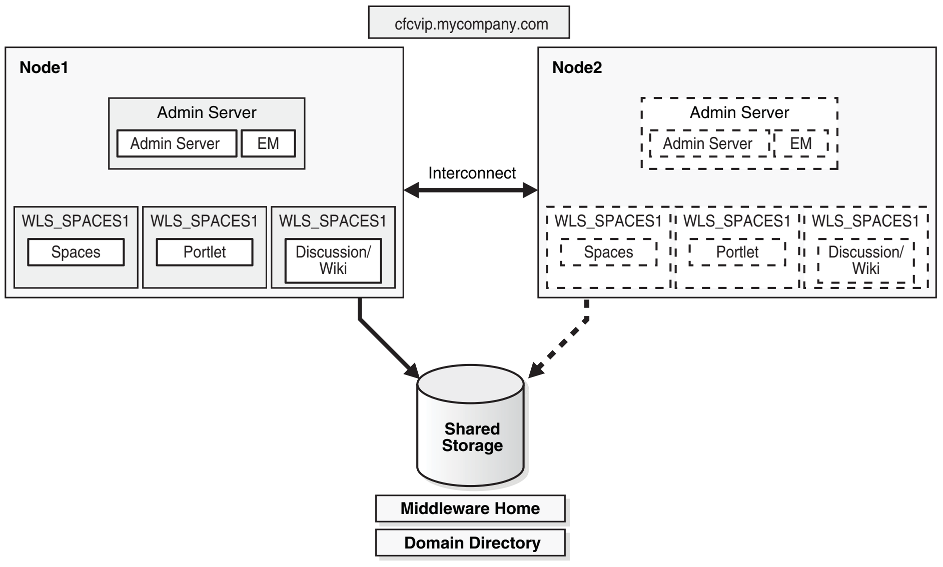
10.3 Oracle Fusion Middleware Cold Failover Cluster Example Topologies

In this section illustrates some example Cold Failover Clusters topologies. Since there are many possible combinations of topologies, these topologies are illustrative only. To achieve these topologies, more than one of the transformation steps apply. Refer to the steps mentioned earlier to configure the transformation.

10.3.1 Example Topology 1

Figure 10–3 shows an Oracle WebCenter Cold Failover Clusters deployment. Both the administration server and the WebCenter Managed Servers are in the domain, and failover as unit. Therefore, they share the same virtual IP and are installed together on the same shared disk. There may be an Oracle HTTP Server front ending this topology. It is on a separate node in the example topology. It can also be on the same node, and can be part of the Cold Failover Clusters deployment. In this example, the database is also on a separate node. However, it is equally likely that the database is on the same cluster and is also Cold Failover Clusters-based (using its own virtual IP and shared disk).

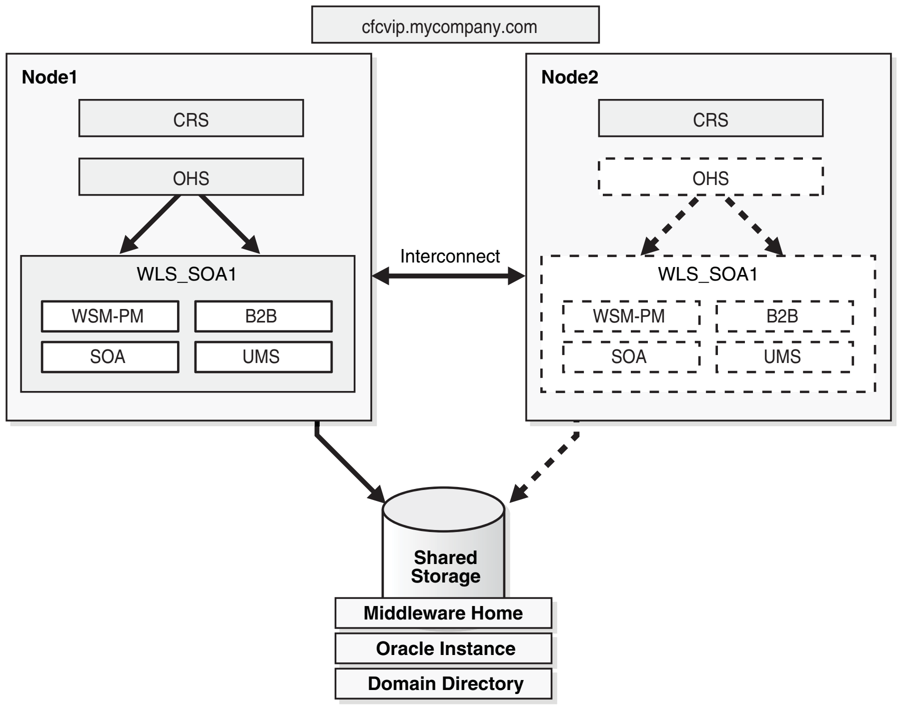
Figure 10–3 Cold Failover Cluster Example Topology 1



10.3.2 Example Topology 2

Figure 10–4 shows an example SOA Cold Failover Clusters deployment. In this example, only the SOA instance is deployed as Cold Failover Clusters, and the administration server is on a separate node. The database is also on a separate node in this example topology. Oracle HTTP Server in this case is part of the Cold Failover Clusters deployment, and part of the same failover unit as the SOA Managed Servers. Important variants of this topology include a Cold Failover Clusters administration server on the same hardware cluster. It may share the same virtual IP and shared disk as the SOA Managed Servers (SOA and administration server are part of the same failover unit) or use a separate virtual IP, and shared disk (administration server fails over independently). Similarly, depending on the machine capacity, the database instance can also reside on the same hardware cluster.

Figure 10–4 Cold Failover Cluster Example Topology 2

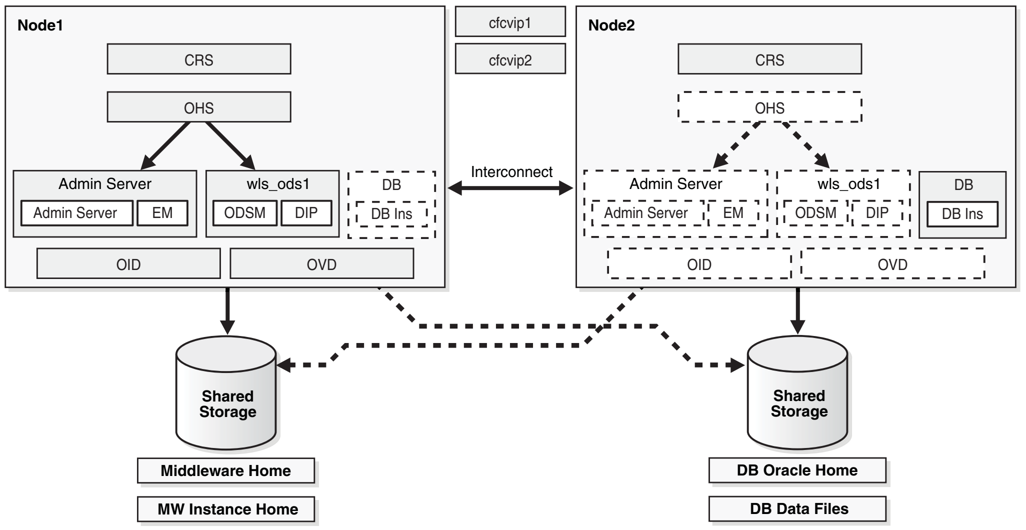


10.3.3 Example Topology 3

Figure 10–5 shows an Oracle Identity Management deployment. In this example topology, all components are on a two-node hardware cluster. Identity Management fails over as a unit, and both the Java EE (administration server and WLS_ods Managed Server) and system components are part of the same failover unit. They share the same virtual IP and shared disk (cfcvip1.mycompany.com). The database is also on the same hardware cluster. It uses a different virtual IP (cfcvip2.mycompany.com), and a different set of shared disks. During normal operations, the database runs on Node2 and the IDM stack runs on Node1. The other node acts as a back up for each.

This topology is the recommended topology for most Cold Failover Clusters deployments. The example is for Identity Management, but this is true for the oracle SOA, Oracle Web Center, and Oracle Portal, Forms, Reports, and Discoverer suites as well. In this recommended architecture, Oracle Fusion Middleware runs as one node of the hardware cluster. The Oracle database runs on the other node. Each node is a backup for the other. The oracle Fusion Middleware instance and the database instance failover independently of each other, using different shared disks and different VIPs. This architecture also ensures that the hardware cluster resources are optimally utilized.

Figure 10–5   Cold Failover Cluster Example Topology 3



11 Using Oracle Cluster Ready Services

This chapter describes conceptual information as well as configuration procedures for Oracle Cluster Ready Services. It contains the following sections:

11.1 Introduction to Oracle Clusterware

Oracle Clusterware manages the availability of user applications and Oracle databases in a clustered environment. In an Oracle Real Application Clusters (RAC) environment, Oracle Clusterware manages all of the Oracle Database processes automatically. Anything managed by Oracle Clusterware is known as a cluster resource, which could be a database, an instance, a service, a listener, a virtual IP (VIP) address, or an application process.

Oracle Clusterware was initially created to support Oracle RAC. Its benefits, however, are not limited to the support for Oracle RAC; it can also be used to manage other applications through add-on modules or scripts. It is this flexibility and extensibility that Oracle Clusterware forms the basis of a high availability solution for Oracle Fusion Middleware.

For more information about Oracle Clusterware, see the *Oracle Clusterware Administration and Deployment Guide*. You can find this guide on the Oracle Technology Network.

11.2 Cluster Ready Services and Oracle Fusion Middleware

Oracle Clusterware includes a high availability framework. This framework provides an infrastructure to protect any resource. In Oracle Clusterware terminology, a resource refers to an object that is created by Oracle Clusterware to identify the entity to be managed, such as an application, a virtual IP or a shared disk. Oracle Clusterware ensures that its managed resources start when the system starts. Oracle Clusterware also monitors the resources to make sure that they are always available.

With this high availability framework, Oracle Clusterware manages resources through an action program supplied by the user when this resource is created. For example, a resource is created for an application running in a single process and the action script knows how to start, and stop this process, and check its state. If an application fails, Oracle Clusterware attempts to restart the process using this script. If the node on which the application is currently running fails, Oracle Clusterware attempts to restart it on another node if the application is configured properly. You can configure application monitoring frequency, starting, and stopping, and the application dependencies.

Oracle Fusion Middleware is a critical application environment and benefits from Oracle Clusterware's high availability feature. Oracle Clusterware includes a add-on component, Application Server Cluster Ready Service (ASCRS) that makes it easier to manage middleware resources.

ASCRS is made up of a frontend and a backend. The frontend is a command line interface, *ascrsctl*, with which you can perform administrative tasks, such as resource creation, deletion, update, start, stop or switchover between cluster nodes. The backend is the action logic needed for the various supported resources. The frontend and backend have their own separate log files.

ASCRS supports virtual IP, shared disk, database listener, database instance and WebLogic Administration Server. You can create Oracle Clusterware resources from these middleware components, allowing Oracle Clusterware and ASCRS to maintain their high availability within the cluster.

Oracle Clusterware and ASCRS provide a means to improve the survivability of the various resources when their hosting environment is corrupted or lost. They do not protect disk corruption or application malfunctioning caused by disk corruption. In addition, since ASCRS does not do sure-fire validation for some user input, such as Oracle SID, inadvertent user input errors may also cause failure for a successful resource management.

ASCRS supports Oracle Clusterware 10.2.0.4 or 11.1.0.7 and higher.

ASCRS has online help that can be invoked using the following command:

```
ascrsctl help -c command -t resource_type
```

For example, to see online help for creating a of virtual IP resource use the following command:

```
ascrsctl help -c create -t vip
```

To view the contents of ascrsctl online help, see Appendix H, "ascrsctl Online Help."

11.3 Installing and Configuring Oracle Clusterware with CRS

As an extension of CRS, ASCRS must be installed within each CRS home on each node of the cluster and configured separately before it can be used.

With the ASCRS command line tool ascrsctl, you can give ASCRS control of various middleware components. Once a component is controlled by CRS, its runtime state is closely monitored and CRS takes the proper actions if the component fails. With ascrsctl, you can create a CRS resource on which you can perform start, stop, update, switchover, monitor status, and delete operations.

11.3.1 Installing ASCRS

Install ASCRS on each node of the cluster. To successfully install ASCRS on a particular node, check the following:

- Operating system: ASCRS is supported only on Unix platforms. The system version and patch level should be compatible to the CRS version supported on that platform.
- CRS and version: CRS is installed on this system, started, and functioning correctly. The CRS version must be 10.2.0.4 or higher. For information about installing Oracle Clusterware and CRS, see the *Oracle Clusterware Installation Guide for Linux*.
- User account: The ASCRS installation user should be the same as the owner of the CRS home.

Note: To install ASCRS for CRS 10.2.0.4, Sun JDK (or JRE) 1.5 or higher must be installed on the local system. It is needed by ascrsctl, the command line tool.

To install ASCRS:

1. Login as the CRS owner.
2. Insert the Oracle Fusion Middleware Companion CD and run the following commands to unzip and install the ascrs.zip file:

```
cd CRS_HOME
unzip Disk1/ascrs/ascrs.zip
cd ascrs/bin
setup
```

If the CRS version is 10.2.0.4, and JDK(JRE) 1.5+ or higher is installed, run the following command:

```
setup -j JDK/JRE_HOME
```

When the installation is complete, the following ASCRS directory structure appears:

```
CRS_HOME/ascrs/bin
CRS_HOME/ascrs/config
CRS_HOME/ascrs/lib
CRS_HOME/ascrs/log
CRS_HOME/ascrs/public
CRS_HOME/ascrs/perl
CRS_HOME/ascrs/sql
CRS_HOME/ascrs/wlst
```

11.3.2 Configuring ASCRS with Oracle Fusion Middleware

After installing ASCRS, it is ready for use with the default configuration. To customize logging locations, logging levels or the default CRS properties, edit the configuration files `config.xml` and `ascrs.properties` files located in the *CRS_HOME*/ascrs/config directory.

The config.xml file contains the configuration for ascrsctl ASCRS agent logging. To change either of their locations, specify an existing path name or a path name within this CRS home using the *ORACLE_HOME* prefix. The available logging levels in the decreasing order of verbosity are ALL, FINEST, FINER, FINE, INFO, WARNING and SEVERE.

The following represents the default `config.xml` file included with ASCRS:

```
<?xml version="1.0" ?>
<config>
  <ascrsctl>
    <display level="normal"/>
          <log path="${ORACLE_HOME}/ascrs/log/ascrsctl.log"
               level="FINER"/>
  </ascrsctl>
  <agent>
    <log path="${ORACLE_HOME}/ascrs/log" level="FINER"/>
  </agent>
</config>
```

The `ascrs.properties` file contains the attributes used in resource creation.

The following represents the default `ascrs.properties` file:

```
normal.vip.AUTO_START=1
normal.vip.CHECK_INTERVAL=7
normal.vip.FAILOVER_DELAY=5
normal.vip.FAILOVER_INTERVAL=50
normal.vip.FAILOVER_THRESHOLD=5
normal.vip.RESTART_ATTEMPTS=2
normal.vip.SCRIPT_TIMEOUT=30
normal.vip.START_TIMEOUT=30
normal.vip.STOP_TIMEOUT=30

# vip resource policies
fast.vip.AUTO_START=1
fast.vip.CHECK_INTERVAL=5
fast.vip.FAILOVER_DELAY=4
fast.vip.FAILOVER_INTERVAL=30
fast.vip.FAILOVER_THRESHOLD=5
fast.vip.RESTART_ATTEMPTS=2
fast.vip.SCRIPT_TIMEOUT=30
fast.vip.START_TIMEOUT=30
fast.vip.STOP_TIMEOUT=30
# disk resource policies

normal.disk.AUTO_START=1
normal.disk.CHECK_INTERVAL=7
normal.disk.FAILOVER_DELAY=5
normal.disk.FAILOVER_INTERVAL=50
normal.disk.FAILOVER_THRESHOLD=5
normal.disk.RESTART_ATTEMPTS=2
normal.disk.SCRIPT_TIMEOUT=30
normal.disk.START_TIMEOUT=30
normal.disk.STOP_TIMEOUT=30

fast.disk.AUTO_START=1
fast.disk.CHECK_INTERVAL=5
fast.disk.FAILOVER_DELAY=4
fast.disk.FAILOVER_INTERVAL=30
fast.disk.FAILOVER_THRESHOLD=5
fast.disk.RESTART_ATTEMPTS=2
fast.disk.SCRIPT_TIMEOUT=30
fast.disk.START_TIMEOUT=30
fast.disk.STOP_TIMEOUT=30

# Oracle database listener resource policies
normal.dblsnr.AUTO_START=1
normal.dblsnr.CHECK_INTERVAL=50
normal.dblsnr.FAILOVER_DELAY=20
normal.dblsnr.FAILOVER_INTERVAL=300
normal.dblsnr.FAILOVER_THRESHOLD=5
normal.dblsnr.RESTART_ATTEMPTS=4
normal.dblsnr.SCRIPT_TIMEOUT=60
normal.dblsnr.START_TIMEOUT=60
normal.dblsnr.STOP_TIMEOUT=60

fast.dblsnr.AUTO_START=1
fast.dblsnr.CHECK_INTERVAL=40
fast.dblsnr.FAILOVER_DELAY=20
fast.dblsnr.FAILOVER_INTERVAL=250
fast.dblsnr.FAILOVER_THRESHOLD=5
fast.dblsnr.RESTART_ATTEMPTS=4
fast.dblsnr.SCRIPT_TIMEOUT=60
fast.dblsnr.START_TIMEOUT=60
fast.dblsnr.STOP_TIMEOUT=60

# Oracle database resource policies
normal.db.AUTO_START=1
normal.db.CHECK_INTERVAL=120
normal.db.FAILOVER_DELAY=30
normal.db.FAILOVER_INTERVAL=700
normal.db.FAILOVER_THRESHOLD=5
normal.db.RESTART_ATTEMPTS=4
normal.db.SCRIPT_TIMEOUT=300
normal.db.START_TIMEOUT=300
normal.db.STOP_TIMEOUT=300

fast.db.AUTO_START=1
fast.db.CHECK_INTERVAL=60
fast.db.FAILOVER_DELAY=20
fast.db.FAILOVER_INTERVAL=400
fast.db.FAILOVER_THRESHOLD=5
fast.db.RESTART_ATTEMPTS=4
fast.db.SCRIPT_TIMEOUT=300
fast.db.START_TIMEOUT=300
fast.db.STOP_TIMEOUT=300

# Oracle WebLogic server resource policies
normal.as.AUTO_START=1
normal.as.CHECK_INTERVAL=50
normal.as.FAILOVER_DELAY=20
normal.as.FAILOVER_INTERVAL=350
normal.as.FAILOVER_THRESHOLD=5
normal.as.RESTART_ATTEMPTS=4
normal.as.SCRIPT_TIMEOUT=600
normal.as.START_TIMEOUT=600
normal.as.STOP_TIMEOUT=600

fast.as.AUTO_START=1
fast.as.CHECK_INTERVAL=40
fast.as.FAILOVER_DELAY=10
fast.as.FAILOVER_INTERVAL=300
fast.as.FAILOVER_THRESHOLD=5
fast.as.RESTART_ATTEMPTS=4
fast.as.SCRIPT_TIMEOUT=600
fast.as.START_TIMEOUT=600
fast.as.STOP_TIMEOUT=600
```

Consult Oracle Clusterware documentation for the definitions of these properties before editing their values.

Since computing environments vary in speed, Oracle recommends measuring the application's start and stop latency before setting the script, start, and stop timeout values. These values may be twice as much as the observed latencies.

11.4 Using ASCRS to Manage Resources

With the ascrsctl command line you manage CRS resources created for Fusion Middleware components. With this tool you can create, update, start, stop, switch and delete resources.

11.4.1 Creating CRS Managed Resources

ASCRS supports WebLogic Administration Server, Oracle database, Oracle database listener, virtual IP and shared disk. After a CRS managed resource is created for one of these components, it can be controlled by CRS.

CRS resources created with the ascrsctl command line follow a naming convention. Follow this naming convention to ensure that the components function correctly. Oracle also recommends using the CRS installation exclusively Oracle Fusion Middleware, so that all the manageable CRS resources are created with ascrsctl to avoid errors caused by naming inconsistencies.

Under this naming convention, the resource name follows the following format:

```
ora.name.cfctype
```

The *name* refers to the short name of the resource, for example, **sharedisk**, or **myvip**. The *type* refers to one of the resource types, such as **vip**, **disk**, **db**, **dblsnr** or **as**.

For example, to create a virtual IP resource from the virtual IP 192.168.1.10 on network interface eth0 with netmask 255.255.255.0, use the following command:

```
ascrsctl create -name myvip -type vip -ipAddr 192.168.1.10 -netmask 255.255.255.0
-interface eth0
```

11.4.1.1 Creating a Virtual IP Resource

Oracle Cluster Ready Services includes a high availability framework that provides an infrastructure to protect any resource. In Oracle Clusterware terminology, a resource refers to an object that is created by Oracle Clusterware to identify the entity to be managed such as an application, a virtual IP or a shared disk. If the auto start for each resource is set to 1, Oracle Clusterware ensures that resources that it manages start when CRS starts. Oracle Clusterware also monitors the resources to make sure that they are always available.

Virtual IP resource is a system resource. The create or update command generates a script to be executed as root user to complete the operation.

See Section 11.4.1, "Creating CRS Managed Resources" for detailed syntax and complete command line options for virtual IP resources.

11.4.1.2 Creating a Shared Disk Resource

In a Fusion Middleware environment, shared disks are those disk storages that are used to hold Oracle database software, the database data files, and WebLogic servers.

Shared disks allow the use of the same data when the database or WebLogic instance is switched among the nodes within a cluster.

To create a shared disk resource, run the following ascrsctl command that includes a valid mount point, a mount command, and an unmount command:

```
ascrsctl create -n sharedisk -type disk -path /asdisk -mc "/bin/mount
 /dev/sda /asdisk" -umc "/bin/umount /asdisk"
```

When creating a shared disk resource, carefully consider the following:

- Before creating a shared disk resource, create an empty signature file named `.ascrssf` on the root of the shared disk. This file is used by CRS after the resource is created.
- You can specify a special no operation, or `nop` command for either the mount or unmount commands. You can use it for the mount command if the shared disk is never offline. If the disk does go off line for some reason, CRS will detects it and mark it as down. The `nop` command can be used for the unmount command the disk does not need to be unmounted by CRS. In such a case, be absolutely sure that the disk does not need to be unmounted. There are potential disk corruption issues if the shared disk is mounted on two nodes without protection. Again, the signature file is always needed on the shared disk.
- The unmount command, may fail if there are active processes using the shared disk. To prevent this command failure, avoid accessing this disk from other applications while this disk resource is in online state.
- For complex mount and unmount commands, encapsulate the logic in executable scripts and specify the full path of these scripts as the mount and unmount commands. A proper unmount script is capable of killing other processes that are using this disk to ensure a successful and clean disk unmount. If the unmount command is in a script, do some basic file system checking, such as running a fsck command.
- A shared disk resource is a system resource. Create, update, or delete commands generate scripts that must be executed as root to complete the create operation. Follow the instructions from the screen output.
- If the signature file is at the mount point of the shared disk, the start/stop operation may fail. Having the signature file on the mount point signals ASCRS that the disk is mounted, even if its not.
- Validate the mount/unmount command before using it in the `mc` or `umc` parameters or in the script file. There is no validation from ASCRS for the commands.

11.4.1.3 Creating an Oracle Database Listener Resource

To create an Oracle database listener resource, the following is required:

- A valid Oracle database home
- The listener name
- The virtual IP resource name for the listen address
- The disk resource name for the Oracle home

To create an Oracle database listener resource, use the following ascrsctl command:

```
ascrsctl create -n mydblsnr -type dblsnr -loh /cfcdb -ln LISTENER -disk ohdisk
-vip myvip
```

Before creating the database listener resource, carefully consider the following:

- The database listener home is installed on a shared disk. A CRS resource has been created for the shared disk with an ascrsctl command, and the resource is started.
- A CRS resource has been created for the virtual IP with an ascrsctl command, and the resource is started.
- The listener Cold Failover Cluster (CFC) enabled. See Section 10.2.4, "Transforming an Oracle Database" for details.
- Ensure that the listener name and Oracle home are valid database sid and database home. ASCRS does not do exhaustive validation on this type of information.

11.4.1.4 Creating an Oracle Database Resource

Creating an Oracle database resource requires the following:

- A valid Oracle database home
- The database sid name
- Disk resource names for the Oracle home
- Data files
- The listener resource name

To create an Oracle database resource, run the following ascrsctl command:

```
ascrsctl create -name string -type db -oraHome string -oraSID string -disk string
[string...]

-lsnr string [-pfile string] [options]
```

For online help information for creating an Oracle database resource, use the following command:

```
ascrsctl help -c create -t db
```

Before creating the Oracle database resource, carefully consider the following:

- The database is installed on a shared disk. The data files of this database are on the same or different shared disk(s). CRS resources have been created for all these shared disks with ascrsctl and started.
- A CRS resource has been created for the database listener with an ascrsctl command, and the resource is started.
- The database needs to be CFC enabled. See Section 10.2.4, "Transforming an Oracle Database" for details.
- Ensure the database sid and Oracle home are valid. ASCRS does not do extensive validation of this information.

11.4.2 Updating Resources

You can update resources created with ascrsctl using the update command. Depending on the resource type, you can update the resource profile by specifying the appropriate parameter through the update command line. Some updates can only be performed when the resource is taken offline.

For example, to update the virtual IP resource created in the last section with a new IP address and a different interface, use the following command:

```
ascrsctl update -n myvip -type vip -ip 192.168.1.20 -if eth1
```

11.4.3 Starting Up Resources

When a resource is started, it is put under the control of CRS and its runtime status is monitored continuously by CRS. If the resource depends on other resources, starting this resource automatically starts the dependent resources. Refer to Oracle Clusterware documentation for information about the role of resource placement policy during resource start up. The ascrsctl start command maps to the CRS command `crs_start`.

For example, to start the virtual IP resource, use the following command:

```
ascrsctl start -n ora.myvip.cfcvip
```

Note: If a resource depends on more than one resource, while starting that resource, be sure that the resources, if online, are targeted on the same node.

For example, if `AS` resource depends on `VIP` and `Disk` resource. While starting `AS` resource, make sure that if `VIP` and `Disk` resources are online, they are targeted to the same node.

11.4.4 Shutting Down Resources

When a resource is stopped, it is brought down and put in offline state and CRS stops monitoring its runtime status. If the resource is depended on other resources that are in online status, the dependent resources are not stopped unless the -f option is specified. This option forces the resource and all of its dependents to stop. Refer to Oracle Clusterware documentation for more information about the implications of resource dependency during resource stop. The ascrsctl stop command maps to CRS command `crs_stop`.

For example, to stop the virtual IP resource, run the following command:

```
ascrsctl stop -n ora.myvip.cfcvip
```

11.4.5 Resource Switchover

Resource switchover is a process of shutting down the resource on the node on which it is running and restarting it on another node. The new node, if not specified, is determined by CRS, based on the placement policy. If the resource to be switched depends on other resources, or there are resources that are online and depend on it, this resource must be switched with -f flag.

To switch over a resource to another available node in the cluster, run the following command:

```
ascrsctl switch -n ora.myvip.cfcvip
```

11.4.6 Deleting Resources

You can remove a resource from CRS control. Once this association is removed, the corresponding application or component's functionality are not affected. CRS no longer monitors that resource’s state. If a resource has dependent resources, it can not be removed.

To delete a resource from CRS control, run the following command:

```
ascrsctl delete -n ora.myvip.cfcvip
```

11.4.7 Checking Resource Status

You can check resource status with the ascrsctl status command. With this command, you can view the states of all resources and their dependents. If a particular resource is specified, the status command show its CRS profile, its direct and indirect dependency relationships, and its current state information.

For example, to check the status of a resource, run the following command:

```
ascrsctl status -n ora.myvip.cfcvip
```

Assuming the virtual IP resource is used by a database listener resource and the listener resource is in turn required by a database resource, all the dependency information is shown in a tree structure, along with other status information in the following status output:

```
Basic information
------------------------+-----------------------
    Name                 |  ora.myvip.cfcvip
    Target state         |  OFFLINE
    Resource state       |  OFFLINE
    Hosting members      |  stajz11, stajz12
------------------------+-----------------------
  Common CRS parameters
------------------------+-----------------------
    Auto start           | No
    Check interval       | 60 sec
    Failover delay       | 10 sec
    Script timeout       | 50 sec
    Start timeout        | 100 sec
    Stop timeout         | 100 sec
------------------------+-----------------------
  Resource specific parameters
------------------------+-----------------------
    Interface(s)         | eth2
    Netmask              | 255.255.252.0
    Virtual IP address   | 140.87.27.48
 ------------------------+-----------------------
  Resource dependency tree(s)
 ------------------------------------------------
ora.mydb.cfcdb
|
+->ora.mydblsnr.cfcdblsnr
| |
| +->ora.mydisk.cfcvip
| |
| +->ora.myvip.cfcvip
|
+->ora.mydisk.cfcdisk
```

11.4.8 Configuring Oracle WebLogic Administration Server

Creating a WebLogic Administration Server CRS resource requires more preparation than other resource types. Due to its complexity, the procedure for creating a WebLogic Administration Server CRS resource is divided into the following sections:

- Basic Setup
- Node Manager Setup
- Administration Server Setup
- Creating a Resource

Basic Setup

Before starting the basic setup, be sure that WebLogic Server is installed on shared disk(s). WebLogic Server software and the administration server domain instance can be installed on the same or separate shared disk.

In addition, ensure that the WebLogic Server is CFC enabled. See Section 10.2.2.1, "Administration Server Topology 1" for details on enabling WebLogic Server for CFC. Once CFC is enabled, you can manually start and stop the Administration Server, the original node, and the failover node(s) without noticeable difference.

The steps for the basic setup section of the procedure for creating a WebLogic Server CSR resource:

1. Create a CRS resource for each shared disk and start it on the node on which it was created.
2. Create a CRS resource for the virtual IP with the ascrsctl command and start it on the same cluster node.

Node Manager Setup

To set up the Node Manager

1. If you have not yet done so, change Node Manager's username and password. The initial password is randomly generated. To change the Node Manager password, in the WebLogic Server Administration Console, select **Domain**, **Security**, **General**, and then **Advanced**. Enter the new password and click **Save**.
2. If you have changed anything in steps 1 or 2, restart the Node Manager using the following command from the *WLS_HOME*/server/bin directory:

   ```
   startNodemanager.sh
   ```

3. Start the WebLogic scripting tool in the *WLS_HOME*/common/bin/wlst.sh directory. To persist Node Manager's user login information in the `ascrscf.dat` and `ascrskf.dat` files, use the following commands:

   ```
   nmConnect('nmUser','nmPasswd','hostname','nmPort','domainName','domainDir')
   storeUserConfig(WLS_HOME/common/nodemanager/ascrscf.dat,
                   WLS_HOME/common/nodemanager/ascrskf.dat,'true')
   nmDisconnect()
   exit()
   ```

4. Copy *CRS_HOME*/ascrs/public/cfcStartNodeManager.sh to the *WLS_HOME*/server/bin directory, and make the script executable.

Note: To keep the setup consistently in sync, step 3 must be performed whenever the Node Manager passwords or usernames are changed.

After you have started Node Manager for the first time, you can edit the `nodemanager.properties` file to set the `StartScriptEnabled` property. The `nodemanager.properties` file does not exist until Node Manager is started for the first time.

In the WLS_HOME/common/nodemanager directory, set the `StartScriptEnabled` property in the `nodemanager.properties` file to `true`.

```
StartScriptEnabled=true
```

Check the `nodemanager.properties` file to ensure no value is assigned to ListenAddress, and that a valid port number is assigned to ListenPort.

When this property is set in the `nodemanager.properties` file, you no longer need to define it in the JAVA_OPTIONS environment variable.

Administration Server Setup

1. The administration server listens on the virtual IP. To ensure this is configured correctly, log in to the WebLogic Server Administration Console and navigate to the administration server listen address page and verify that the virtual IP and the port number are both set correctly and click **Save**.
2. The administration server must also listen on the localhost. To ensure this is configured correctly, login WebLogic Server Administration Console and do the following:
    a. In the Domain tree, select **Environment**, **Servers**, **AdminServer**, **Protocols**, and then **Channels**.
    b. Click the **Lock and Edit** button.
    c. Click **New**, select a channel name, and protocol **t3**, and continue to the next screen.
    d. Enter the localhost for both the **Listen Address** and **External ListenAddress**.
    e. Enter the port number to the **Listen Port** and **External ListenPort**. This port number must be exactly the same as the port number used for the virtual IP.
    f. Continue to the next screen and verify that **Enabled** is selected.
    g. Click **Finish**.
    h. Click **Activate Changes**.
3. Ensure the *DOMAIN_HOME*/servers/AdminServer/security directory exists. This directory should contain the `boot.properties` file. If this file does not exist, create it and include the following properties:

    ```
    username=<admin server user name>
    password=<admin server user password>
    ```

4. If *DOMAIN_HOME*/servers/AdminServer/data/nodemanager/startup.properties exists, ensure the property `AutoRestart` defined in this file is set to `false`.
5. Restart the administration server.

6. Shutdown all WebLogic processes and kill the Node Manager.

Note: To keep the setup consistently in sync, Step 3 must be performed whenever the administration server password is changed.

Create the Resource

After Basic Setup, Node Manager Setup, and Administration Server Setup, create the CRS resource using the following command:

```
ascrsctl create -n adminserver -type as -ch /cfcas -disk sharedisk -vip myvip
```

Note: The WebLogic component home argument (-ch) must be valid. ASCRS does not perform extensive validation for this information.

Note: For Oracle Portal, Forms, Reports, and Discoverer, the administration server cannot be separated from the Managed Server. ASCRS supports management of only the administration server in this release. As a result, the Managed Server running in the same MWOH and Domain Directory as the administration server must be down for ASCRS managed deployments.

11.5 Example Topologies

The following two examples illustrate how to use ASCRS to manage Fusion Middleware resources.

Figure 11–1 CRS Topology Example 1

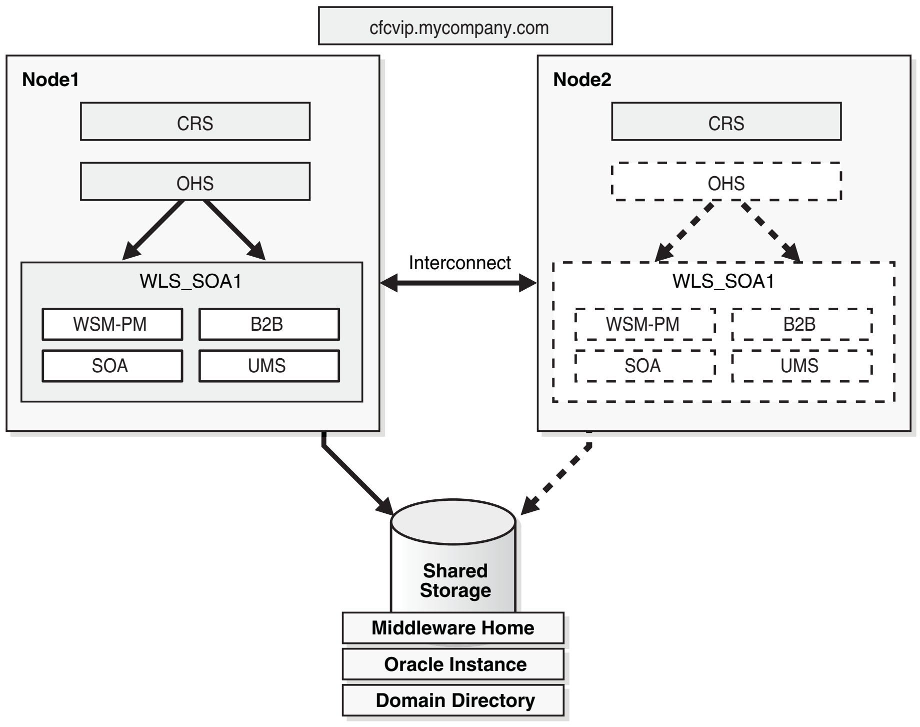


Figure 11–1 illustrates the CRS Example 1 topology. In the example topology, Oracle WebLogic Server is installed on a two-node cluster. Of all the WebLogic servers in the same domain, WebLogic Administration Server is the only one running on this cluster. The goal is to provide a failover solution for the WebLogic Administration Server that runs in a Java EE container, as well as Oracle Enterprise Manager. Both the WebLogic software and the Domain home reside on the same shared disk.

Assumptions:

- Operating Environment: This is a Linux, two-node cluster with node1.company.com and node2.company.com as its members. Node 1 is designated as the primary node and node2 is the failover node. CRS has been installed on both nodes in the /crshome directory and is started. ASCRS has been installed on both nodes and has been configured.
- One shared disk has been allocated for the WebLogic software and the Domain home of the administration server. It is a SCSI drive identified with /dev/sda1 and has ext2 file system on it. It is mounted on /sharedisk1.
- The WebLogic Server uses virtual IP 192.168.1.10 for its public listen address and 7001 for its listen port. One each node, two network interface controllers, eth0 and eth1 are available for binding the virtual IP. The netmask is 255.255.255.0.
- WebLogic Server is installed on the shared disk on the /sharedisk1/fmw directory and the domain directory is /sharedisk1/fmw/user_projects/domains/asdomain.

After creating resources for the shared disk as described in Section 11.4.1.2, "Creating a Shared Disk Resource,"and configuring a resource for the WebLogic Administration Server as described in Section 11.4.8, "Configuring Oracle WebLogic Administration Server,", perform the following steps to create all the resources:

1. Shutdown WebLogic server and its node manager completely.
2. Unmount the shared disk with command `/bin/unmount`.
3. Unbind the virtual IP with command `/sbin/ifconfig`.
4. CD to the /*CRS_HOME*/ascrs/bin directory.
5. Create the virtual IP resource using the following command:

   ```
   ascrsctl create -n asvip -t vip -if "eth0|eth1" -ip 192.168.1.10 -nm
   255.255.255.0
   ```

6. Create the shared disk resource using the following command:

   ```
   ascrsctl create -n asdisk -t disk -path /sharedisk1
   -mc "/bin/mount /dev/sda1 /sharedisk1"
   -umc "/bin/umount /sharedisk1"
   ```

7. Create the administration server resource using the following command:

   ```
   ascrsctl create -n wlas -t as -vip asvip -disk asdisk
   -ch /sharedisk1/fmw/user_projects/domains/asdomain
   ```

8. Start all the database related resources using the following command:

   ```
   ascrsctl start -n ora.wlas.cfcas
   ```

Figure 11–2 CRS Topology Example 2

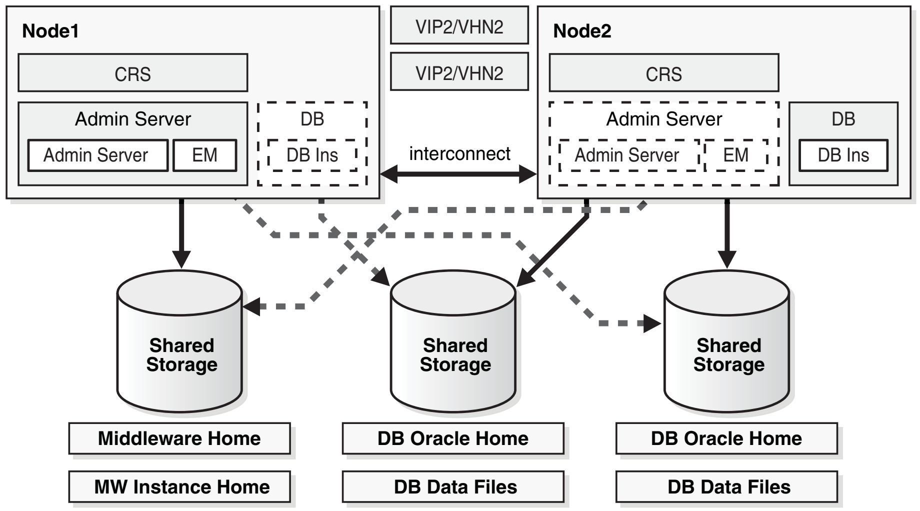


Figure 11–2 illustrates the CRS Example 2 topology. In this example topology, WebLogic Server and the Oracle database are installed on a two-node cluster with the following characteristics:

- The WebLogic Administration Server is the only server running on this cluster. The WebLogic software and the Domain home reside on the same shared disk.
- The administration server, along with Oracle Enterprise Manager, run in a WebLogic Java EE container with the first node as its primary node.
- The database software and its data files reside on two other shared disks with the second node as its primary node.

The goal of this topology is to provide a failover solution for both the WebLogic Administration Server and the database instance.

Assumptions:

- Operating Environment: This is a Linux two-node cluster with node1.company.com and node2.company.com as its members. Node 1 is designated as the primary node for WebLogic Server and Node 2 as its failover node. Node 2 is designated as the primary node for the Oracle database, and node 1 as its failover node. CRS is installed on both nodes in the /crshome directory and is started. ASCRS is installed on both nodes and is configured.
- Three shared disks are allocated. They are all SCSI drives identified with /dev/sda1, /dev/sda2, /dev/sda3 and have ext2 file system on it. They are used for WebLogic software and the Domain home, the Oracle database software and the data files. They are mounted on /sharedisk2, /sharedisk2 and /sharedisk2 respectively.
- The WebLogic server uses virtual IP 192.168.1.10 for its public listen address, and 7001 for its listen port. On each node, two network interface controllers, eth0 and eth1 are available for binding this virtual IP.

  The database listener uses virtual IP 192.168.1.20 for its listen address, and 1521 for its listen port. On each node, network interface controller eth2 is available for binding this virtual IP.

  The netmask is 255.255.255.0 for both virtual IPs.

- WebLogic Server is installed on the shared disk in the /sharedisk1/fmw directory, and the domain directory is /sharedisk1/fmw/user_projects/domains/asdomain.
- The Database is installed on the shared disk in the /sharedisk2/dbhome directory, and the data files are located in the /sharedisk3/dbdata directory. Assume orcl is the Oracle SID name and LISTENER is the listener name.

  The WebLogic resource is created just as the Example 1 topology. The database resource is created with the following steps:

After creating resources for the shared disk as described in Section 11.4.1.2, "Creating a Shared Disk Resource,"and configuring a resource for the WebLogic Administration Server as described in Section 11.4.8, "Configuring Oracle WebLogic Administration Server,", perform the following steps to create all the resources:

1. Shutdown the database and its listener.
2. Unmount the shared disks /sharedisk2 and /sharedisk3 with the /bin/unmount command.
3. Unbind the virtual IP with the /sbin/ifconfig command.
4. Create the virtual IP resource for the listener using the following command:

   ```
   ./ascrsctl create -n dbvip -t vip -if eth2
   -ip 192.168.1.20 -nm 255.255.255.0
   ```

5. Create the shared disk resource for the Oracle home using the following command:

   ```
   ./ascrsctl create -n dbdisk -t disk -path /sharedisk2
   -mc "/bin/mount /dev/sda2 /sharedisk2"
   -umc "/bin/umount /sharedisk2"
   ```

6. Create the shared disk resource for the data files using the following command:

   ```
   ./ascrsctl create -n dfdisk -t disk -path /sharedisk3
   -mc "/bin/mount /dev/sda3 /sharedisk3"
   -umc "/bin/umount /sharedisk3"
   ```

7. Create the database listener resource using the following command:

   ```
   ./ascrsctl create -n asdblsnr -t lsnr -vip dbvip
             -disk dbdisk -loh /sharedisk2/dbhome -ln LISTENER
   ```

8. Create the database resource using the following command:

   ```
   ./ascrsctl create -n asdb -t db -lsnr asdblsnr -sid orcl
            -oh /sharedisk2/dbhome -disk dbdisk dfdisk
   ```

9. Start all the database related resources on node2 using the following command

   ```
   ./ascrsctl start -n ora.asdb.cfcdb
   ```

11.6 Troubleshooting Oracle CRS

ASCRS relies on logging for diagnosing unexpected issues. To get more diagnostic information, you can increasing the verbosity of the log level by changing the ASCRS configuration file `config.xml`.

In addition, you can also check CRS daemon logs for basic CRS issues.

12 Configuring High Availability for Oracle Portal, Forms, Reports, and Discoverer

This chapter describes high availability concepts and configuration procedures for Oracle Portals, Forms, Reports, and Discoverer. This chapter includes the following topics:

- Section 12.1, "Overview of Oracle Portal, Forms, Reports, and Discoverer"
- Section 12.2, "Oracle Portal and High Availability Concepts"
- Section 12.3, "Oracle Reports and High Availability Concepts"
- Section 12.4, "Oracle Forms and High Availability Concepts"
- Section 12.5, "Oracle Discoverer and High Availability Concepts"
- Section 12.6, "Configuring Oracle Portal, Forms, Reports, and Discoverer for High Availability"

12.1 Overview of Oracle Portal, Forms, Reports, and Discoverer

Oracle Portal offers a complete portal framework for building, deploying, and managing portals that are tightly integrated with Oracle Fusion Middleware. Oracle Portal provides a rich, declarative environment for creating a portal Web interface and accessing dynamic data with an extensible framework for Java EE-based enterprise application access.

With Oracle Portal, you can enhance your deployments with enterprise content management capabilities through Oracle Universal Content Management. In addition, you can add Enterprise 2.0 capabilities to existing portals using Oracle WebCenter Services. Oracle Portal has certified interoperability with both of these complementary solutions, as well as with Oracle enterprise applications and other Oracle Fusion Middleware components.

When Oracle Portal is implemented, it often becomes the entry point into an organization internet or intranet, and as such, it is crucial that it is as highly available as possible.

Oracle Forms is Oracle's long established technology to design and build enterprise applications quickly and efficiently.

Oracle Reports Services is the reports publishing component of Oracle Fusion Middleware. It is an enterprise reporting service for producing high quality production reports that dynamically retrieve, format, and distribute any data, in any format, anywhere. You can use Oracle Reports Services to publish in both Web-based and non-Web-based environments.

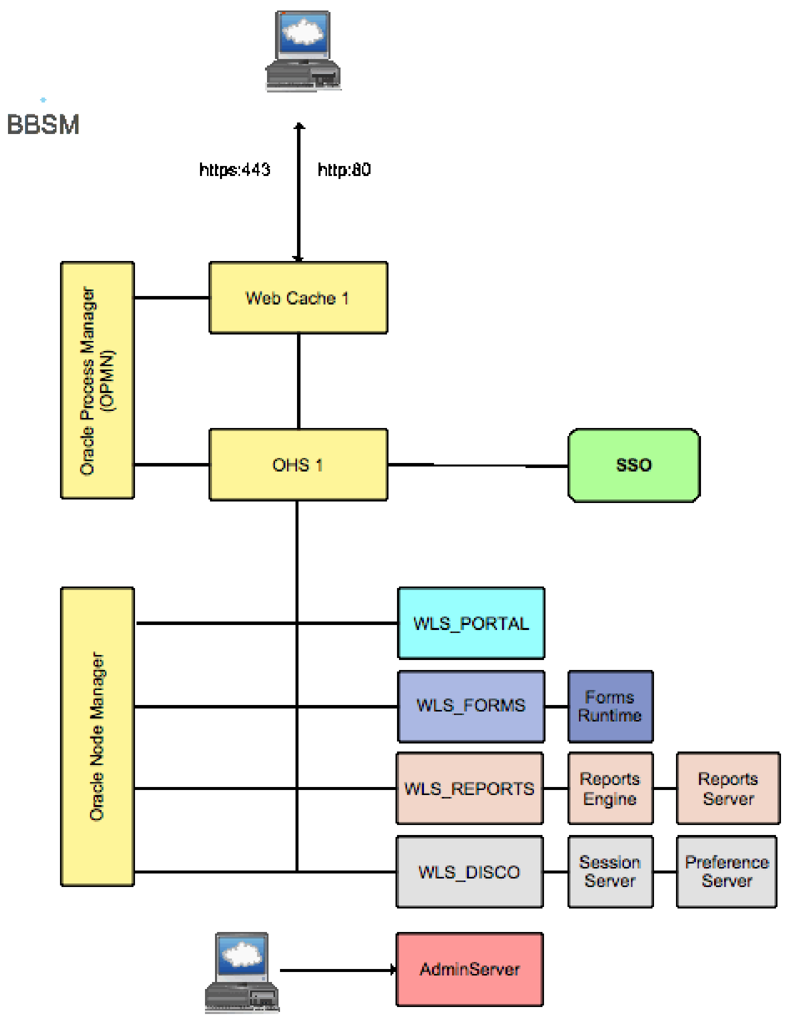
Oracle Discoverer is a business intelligence tool for analyzing data. It is a key component of Oracle Fusion Middleware. Discoverer provides an integrated business intelligence solution that comprises intuitive ad-hoc query, reporting, analysis, and Web publishing functionality. These tools enable non-technical users to gain immediate access to information from data marts, data warehouses, multidimensional (OLAP) data sources, and online transaction processing systems. Discoverer integrates seamlessly with Oracle Portal and Oracle WebCenter, enabling rapid deployment of Discoverer workbooks and worksheets to Web portals.

Oracle Portal, Forms, Reports and Discoverer can be installed individually or collectively.

12.1.1 Oracle Portal, Forms, Reports, and Discoverer Architecture

Figure 12–1 Illustrates a single-instance Oracle Portal, Forms, Reports, and Discoverer deployment.

Figure 12–1 Oracle Portal, Forms, Reports, and Discoverer Architecture



Oracle Portal, Forms, Reports, and Discoverer Common Components

Figure 12–1 includes the following common components:

- **Oracle Web Cache** performs two functions. Its primary function is to serve static Web content from its cache, much faster than could be achieved by the Oracle HTTP Server alone. If Oracle Web Cache does not have a cacheable page in its cache, or that page is not current, it requests the page from the attached Oracle HTTP Servers.

  The second function of Oracle Web Cache is used in high availability environments. It receives a request, and if it cannot service that request from its own cache, it can load balance it between several Oracle HTTP Servers.

  Oracle Web Cache is optional, but recommended when used in conjunction with Oracle Forms, Reports and Discoverer.

- **Oracle HTTP Server** is responsible for assembling requested pages. Page assembly is not always straightforward. Depending on how the page is made up, the Oracle HTTP Server performs one of the following:

  – If the page is a simple HTML document, then the Web tier finds and returns the document.

  – If the Web page needs to be assembled by executing a Java EE application, the Oracle Web Tier routes the request to Oracle WebLogic Server, which, after processing the request, sends the result back to the user through the Oracle Web Tier.

  – If the Web page needs to be assembled by executing some other application such as PLSQL or CGI, the Oracle Web Tier routes the request to the appropriate application, and once that application has processed the request, it sends the result back to the user through the Oracle Web Tier.

  – If the requested page is security controlled, the Oracle Web Server invokes Oracle Identity Management to ensure that the user is authorized to view the page.

  The Oracle HTTP server can be used as stand-alone or in conjunction with Oracle Web Cache.

  In Oracle Portal, Forms, Reports and Discoverer deployments, the Oracle HTTP Server uses an Apache module called mod_wl_ohs to route requests to the Oracle WebLogic Managed Servers. When WebLogic Managed Servers are clustered together, mod_wl_ohs load balances requests among all Managed Servers in a given cluster.

- **Oracle WebLogic Managed Servers** are the Java EE runtime containers for Oracle Portal, Forms, Reports and Discoverer Applications.

- **Oracle Process Manager and Notification Server (OPMN)** is used to start, stop, and monitor Oracle Web Cache and Oracle HTTP Server. It periodically polls Oracle Web Cache and the Oracle HTTP Server to ensure that they are functioning. If they are not functioning, OPMN takes appropriate action to restart the failed component to ensure that service is maintained.

- **Oracle Node Manager** is used to start, stop, and monitor Oracle WebLogic Managed Servers including the administration server. It periodically polls the WebLogic Managed Servers to ensure that they are functioning. If they are not functioning, the Oracle Node Manager takes appropriate action to restart the failed component to ensure that service is maintained.

- The **Oracle WebLogic Administration Server** contains both the WebLogic Administration Console and the Oracle Fusion Middleware Enterprise Manager. It is active only once within a WebLogic domain. In the event of the failure of the server hosting the administration server, it must be manually restarted elsewhere.

- **Oracle Single Sign-On** is Oracle's Enterprise authentication mechanism. Oracle Single Sign-On is integrated into Oracle HTTP Server for each of the product components. When a request which requires authentication comes in to the Oracle HTTP Server, it determines whether the user has been authenticated through the Oracle Single Sign-On server. If the user has been authenticated, the request is processed. If however, the user has not been authenticated, the Oracle Single Sign-On server is contacted to gain authorization.

12.1.2 Common Log Files

The following table lists, describes, and provides the location for generic log files used by Oracle Portal, Forms, Reports, and Discoverer:

File	Location	Description
access_log	*ORACLE_INSTANCE*/diagnostics/logs/OHS/ohs1	Lists each access to the Oracle HTTP Server and the HTTP Return code
ohs1.log	*ORACLE_INSTANCE*/diagnostics/logs/OHS/ohs1	HTTP Server error log
access_log	*ORACLE_INSTANCE*/diagnostics/logs/WebCache/web1	Lists each access to Oracle WebCache and the HTTP Return code
access_log	*DOMAIN_HOME*/servers/WLS_PORTAL/logs	Lists access requests being received by the WebLogic Managed Server
opmn.log	*ORACLE_INSTANCE*/diagnostics/logs/OPMN	OPMN log files

12.1.3 Common Component Failures and Expected Behaviors

This section describes high availability concepts that apply to Oracle Portal, Forms, Reports, and Discoverer.

12.1.3.1 Oracle Web Cache and Oracle HTTP Server Process Failures

Oracle HTTP Server, Oracle Web Cache, and Oracle Discoverer preference server processes are protected by the Oracle Process Manager and Notification system (OPMN). If one of these processes fails, OPMN automatically restarts the process.

Oracle WebLogic Managed Servers are started and monitored by Oracle Node Manager. If an Oracle WebLogic Managed Server fails, Oracle Node Manager restarts the process.

12.1.3.2 Common Component Node Failures

If a node that is not fronted by Oracle Web Cache fails, the Load balancer automatically reroutes requests to a surviving node.

If Oracle Web Cache fails, the Load Balancer automatically reroutes requests to a surviving web cache node.

If a node with Oracle HTTP Server fronted by Oracle Web Cache fails, Oracle Web Cache reroutes the request to a surviving Oracle HTTP Server.

If Oracle WebLogic Server fails, Oracle HTTP Server reroutes requests to another WebLogic cluster member.

Oracle Portal, Forms, Reports and Discoverer requests being processed by the failed node must be restarted.

12.1.3.3 Common Component WebLogic Managed Server Failures

In a high availability configuration, Oracle WebLogic Managed Servers are clustered together. If one of the managed servers fails, mod_wl_ohs automatically redirects requests to one of the surviving cluster members. If the application stores state, state replication is enabled within the cluster, which allows redirected requests access to the same state information.

12.1.3.4 Common Component Database Failures

Databases are recommended to be implemented using high availability technologies such as Oracle Real Application Clusters. If one of the database nodes fails, the database as a whole remains available. In some cases you may have to resubmit the request.

In a multi database node environment, if a user session is connected to the database node that fails, the following occurs:

- Oracle Portal: The user is required to resubmit the request.
- Oracle Forms: The user is required to resubmit the request.
- Oracle Reports: If the database connect string is configured using Oracle Transparent Application Failover, no action is required (unless the report writes to a database during its execution).

  If the database containing the reports queue loses a node, then the user is required to resubmit the report request.
- Oracle Discoverer: The user is required to resubmit the request.

12.1.4 Oracle Portal, Forms, Reports, and Discoverer Cluster-Wide Configuration Changes

Oracle Web Cache instances are clustered. Once a configuration change is made through the Oracle Fusion Middleware Console or through the Oracle Web Cache Administration utility, these changes are propagated to other cluster members. Propagation is done manually using these tools.

Oracle HTTP Servers are not clustered. The Oracle HTTP server configuration is file-based. As a result, changes made to one Oracle HTTP Server must be manually copied to other Oracle HTTP Servers in the configuration. This also applies to static HTML files stored in the htdocs directory.

Configure Oracle Portal, Forms, Reports and Discoverer using a series of configuration files. Any changes to these files must be manually applied to all members in the architecture.

WebLogic Managed Servers are clustered and share resources at the cluster level. Changes to these resources can be made once without the need for propagation. These resources include:

- Data sources
- Application redeployments

- State replication

12.1.5 Common Component Log File Information

Cluster wide log consolidation is not offered for Oracle Web Cache, Oracle HTTP Server, OPMN, and WebLogic Managed Servers. For information about the status of an Oracle HTTP Server application, refer to the log files on each Oracle HTTP Server node. To For information about the status of an failed application, refer to the log files on each Server node.

12.2 Oracle Portal and High Availability Concepts

This section describes single-instance information, as well as high availability concepts specific to Oracle Portal. This section guides you through the concepts and considerations necessary for creating a successful high availability Oracle Portal deployment.

12.2.1 Oracle Portal Single-Instance Characteristics

For information about the single-instance architecture of Oracle Portal, see the following sections in the *Oracle Portal Administrator's Guide*.

- Understanding the Oracle Portal Components - this section introduces the components of the Oracle Fusion Middleware. It describes how these components work with Oracle Portal.
- Understanding the Oracle Portal Architecture - this section describes the Portal architecture. Read the following topics in this section:
  - How Does Oracle Portal Integrate with Other Components?
  - How Are Pages Assembled in Oracle Portal?
  - How Does Communication Flow in Oracle Portal?
- Understanding Caching in Oracle Portal - this section describes the caching configurations you can implement to increase the availability and scalability of medium to large deployments.
- Understanding WSRP and JPS - this section provides an introduction to the Web Services for Remote Portlets (WSRP) specifications and Java Portlet Specification (JPS). These two standards enable the development of portlets that interoperate with different Portal products, thereby increasing the availability of portlets within an organization.

12.2.1.1 Oracle Portal Request Flow

For information about request flow in Oracle Portal, see the following sections in the *Oracle Fusion Middleware Administrator's Guide for Portal Guide*:

- How Are Pages Assembled in Oracle Portal?
- How Does Communication Flow in Oracle Portal?

12.2.1.2 Oracle Portal Component Characteristics

The following lists the characteristics of Oracle Portal components:

- Oracle Portal application runs inside a WebLogic container (WLS_PORTAL).

- Oracle Portal repository stores metadata, documents, customizations, and personalizations.
- Oracle Portal has a strong dependency on Oracle Web Cache to process and cache content.
- Oracle Portal depends on Oracle HTTP server to route traffic to WebLogic Server, service product images, and rewrite URLs.

12.2.1.3 Oracle Portal Startup and Shutdown of Processes and Lifecycle

Oracle Portal does not have any process of its own. It relies on standard Oracle tools and utilities to manage Oracle Web Cache, Oracle HTTP Server, WebLogic Managed Server WLS_PORTAL, and the database.

The following lists the configuration and monitoring interfaces for Oracle Portal components:

- Enterprise Manager for managing Oracle Web Cache, Oracle HTTP Server, Oracle Portal, and WLS_PORTAL
- Oracle WebLogic Administration Console for managing WLS_PORTAL
- Oracle WebLogic Scripting Tool (WLST) for adding or editing Portal-specific configuration parameters
- The `opmnctl` command line interface for managing Oracle Web Cache and Oracle HTTP Server

12.2.1.4 Oracle Portal Deployment Artifacts

The Portal application is deployed as a staged application (`portal.ear`) to WebLogic Server (WLS_PORTAL). The configuration files are available in *DOMAIN_HOME*/servers/WLS_PORTAL/stage/portal/portal/configuration.

12.2.1.5 Oracle Portal Configuration Information

Some of the Portal configuration is stored in the Portal repository. This information includes site details (site host and port), Web Cache details (webcache host, invalidation port, and invalidation password), and Oracle Internet Directory details (Oracle Internet Directory host and port). Such information can be changed by accessing Enterprise Manager or by using WLST commands for any of the containers.

The remaining Portal configuration is available in configuration files that are located inside the staged location of the Portal application. Such configuration files are located in `$DOMAIN_HOME/servers/WLS_PORTAL/stage/portal/portal/configuration`. In the HA environment, if a configuration file is modified, the same change must be repeated on each WLS_PORTAL instance either by repeating the action on each node or by copying over the configuration files from one node to another.

Table 12–1 Portal Configuration Files

Configuration File	Description
`appConfig.xml`	Portal Page Engine configuration

Table 12–1 (Cont.) Portal Configuration Files

Configuration File	Description
`portal_dads.conf`	Portal Database Access Descriptor (DAD) configuration This file contains properties related to High Availability (HA). It contains connect string information to the database. If RAC nodes are added or removed, this file must be updated to reflect the final set of nodes. The file contains a configuration property, which can be enabled to test a pooled database connection before using it. At a minimal cost, this property can be enabled to ensure better results when a database node goes down.
`portal_cache.conf`	Portal Cache configuration
`portal_plsql.conf`	Portal global settings

12.2.1.6 Oracle Portal Logging and Log Configuration

Portal log files are generated in the `DOMAIN_HOME/servers/WLS_PORTAL/logs` directory. The log file is named `WLS_PORTAL-diagnostic.log`. Log rotation ensures that older logs are archived with a similar name. The configuration of log files in Portal is controlled by the `logging.xml` file, which is located in `DOMAIN_HOME/config/fmwconfig/servers/WLS_PORTAL`.

12.2.1.6.1 Oracle Portal Log Files The following table lists and describes log files used by Oracle Portal:

File	Location	Description
WLS_PORTAL.log	*DOMAIN_HOME*/servers/WLS_PORTAL/logs	Log file for the WebLogic Managed Server
WLS_PORTAL.out	*DOMAIN_HOME*/servers/WLS_PORTAL/logs	Supplemental log file for the WebLogic Managed Server. This is generally the first place to look for WebLogic issues.

12.2.1.7 Oracle Portal External Dependencies

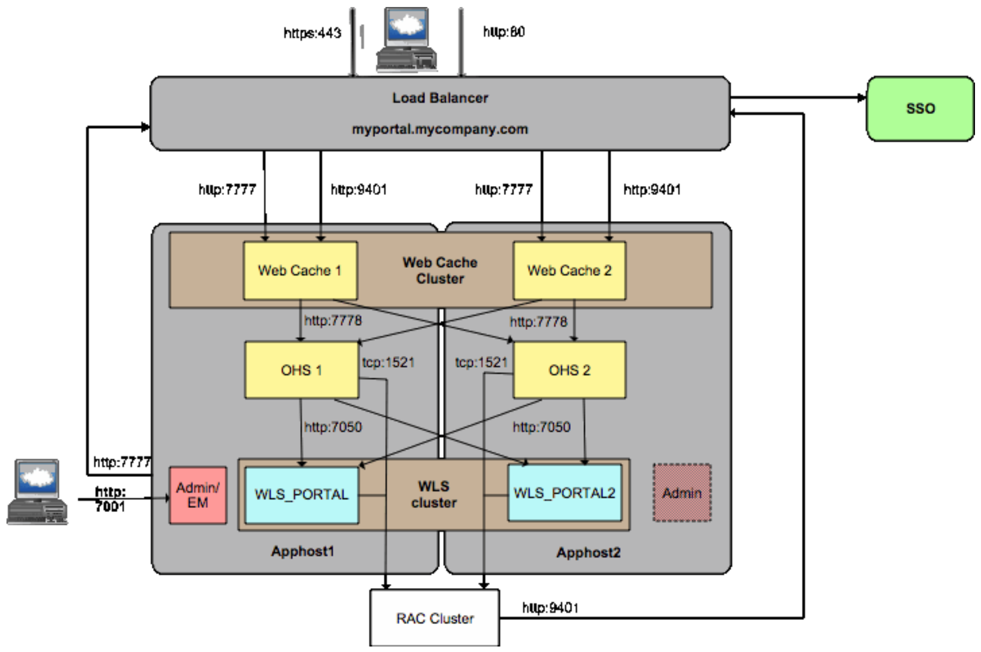
Oracle Portal requires an Oracle HTTP Server to service requests. Optionally these requests may be cached by Oracle Web Cache, in which case it also acts in the capacity of a load balancer.

Oracle Portal requires a database to store information. This database is pre-seeded with schemas using the Oracle Repository Creation Assistant.

12.2.2 Oracle Portal Protection from Failures and Expected Behavior

Figure 12–2 illustrates a high availability configuration for Oracle Portal.

Figure 12–2 Oracle Portal High Availability Deployment



The Oracle Portal high availability setup includes two or more Oracle Web Cache instances. These Oracle Web Cache instances are clustered together to provide cache consistency and failover. Each Oracle Web Cache is configured to route traffic to two or more Oracle HTTP servers, and these servers load balance incoming requests.

Each Oracle HTTP Server is configured to route Oracle Portal requests to two or more Oracle WebLogic managed servers, which host the Oracle Portal application. Oracle HTTP Server is a web server provided by Oracle Fusion Middleware. It incorporates an OpenSSL module to support Secure Sockets Layer (SSL) and HTTP Secure Sockets Layer (HTTPS). Oracle HTTP Server also provides a mod connector to route traffic to Oracle WebLogic Server, which runs Java servlets and J2EE applications. Oracle Portal relies on Oracle HTTP Server to service product images and rewrite URLs rewriting, in some scenarios.

WLS_PORTAL is the WebLogic Server that runs the Oracle Portal application. WebLogic Server provides a complete Java EE environment that includes a JSP translator, a JSP servlet engine (OJSP), and an Enterprise JavaBeans (EJB) container. It provides a fast, lightweight, highly scalable, easy-to-use, complete Java EE environment. It is written entirely in Java and executes on the standard Java Development Kit (JDK) Virtual Machine (JVM).

The Portal application running inside WLS_PORTAL works closely with Oracle Web Cache, using its Edge Side Includes (ESI) processing, to service dynamic Portal pages in a secure fashion. Oracle Portal uses ESI to securely cache various pieces of metadata and content in Oracle Web Cache. As content changes, Oracle Portal invalidates any cached copies in Oracle Web Cache. Such content is regenerated in a subsequent request. ESI allows content to be cached at a more granular level, thereby increasing the cache hit ratio and enabling a more granular invalidation of Portal content. For most content/metadata, Oracle Portal also backs up the in-memory content in Web Cache into the Portal Cache, which is based on file system.

Oracle Portal stores its content and metadata information in a database. To ensure that the database is not a single point of failure, the database is built using Oracle Real Application Clusters.

Finally, a load balancer front ends the Oracle Web Cache instances to provide a single virtual access point to Oracle Portal. Besides load balancing external traffic coming from the browser, the load balancer also provides load balancing for internal traffic generated by Oracle Portal (referred to as Portal Loopback Requests). The load balancer also provides load balancing of invalidation messages generated from the Portal metadata repository. Such invalidation requests are sent to Oracle Web Cache to invalidate any stale content cached in Oracle Web Cache. The clustering feature in Oracle Web Cache ensures cache consistency across the Oracle Web Cache instances.

Note: If you are deploying your own producers, ensure that there is redundancy for the Producers and the producers are front-ended by a load balancer. This step ensures that there is no single-point of failure in the system.

12.2.2.1 Oracle Portal Process Failures

Due to redundancy in each component, if a particular process goes down, Oracle Portal continues to work. Note that Oracle Portal does not attempt to retry a request. Therefore, if a failure occurs while processing a particular request, that request fails.

12.2.2.2 Oracle Portal Node Failures

For information about Oracle Portal Node failure, see Section 12.1.3.2, "Common Component Node Failures."

12.2.2.3 Oracle Portal WebLogic Managed Server Failures

For information about Oracle Portal WebLogic Managed Server failure, see Section 12.1.3.3, "Common Component WebLogic Managed Server Failures."

12.2.2.4 Oracle Portal Protection from Database Failures

For information about Oracle Portal database failure, see Section 12.1.3.4, "Common Component Database Failures."

12.3 Oracle Reports and High Availability Concepts

This section describes single-instance information, as well as high availability concepts specific to Oracle Reports. This section guides you through the concepts and considerations necessary for creating a successful high availability Oracle Reports deployment.

12.3.1 Oracle Reports Single-Instance Characteristics

Oracle Reports Services is the reports publishing component for Oracle Fusion Middleware. It is an enterprise reporting service for producing high quality production reports that dynamically retrieve, format, and distribute any data, in any format, anywhere. Oracle Reports Services can be used to publish in both Web-based and non-Web-based environments.

The following components are specific to Oracle Reports:

Oracle Reports Server

The Reports Server (rwserver) processes client requests, including ushering them through its various services, such as authentication and authorization checking, scheduling, caching, and distribution (including distribution to custom-or pluggable-output destinations). The Reports Server also spawns runtime engines for generating requested reports, fetches completed reports from the Reports Server cache, and notifies the client that the job is ready.

The reports server can be run stand-alone or in-process.

Oracle Reports Engine

The Reports Engine includes components for running SQL-based and pluggable data source-based reports, fetches requested data from the data source, formats the report, sends the output to cache, and notifies the Reports Server that the job is complete.

Oracle Reports Cache

The Reports Cache is used to store the output of reports, which have been successfully run. By using the reports cache it is not necessary to generate the same report repeatedly.

Oracle Reports Queue

The Reports Queue contains a list of the report requests. When processing capacity becomes available, the next report in the queue is executed.

Oracle Reports Customer Databases

Typically, reports are compiled using information stored in a variety of databases. These are referred to as customer databases.

Oracle Single Sign On

Although advisable it is not a mandatory requirement to restrict access to Oracle Reports using Oracle Single Sign-On (SSO).

12.3.1.1 Oracle Reports State Information

Oracle Reports only state information is job metadata. For example, which reports are to be run, and the parameters with which they are run. This is often referred to as the reports queue. This information is stored in operating system files `servername.dat`, or alternatively in a database.

12.3.1.2 Oracle Reports External Dependencies

Oracle Reports requires Oracle HTTP Server to service requests. Optionally these requests may be cached by Oracle Web Cache, in which case it also acts in the capacity of a load balancer.

12.3.1.3 Oracle Reports Specific Configuration Files

The following table lists and locates configuration files used by Oracle Reports:

File	APPHOST1 Location	APPPHOST2 Location
rwserver.conf	*DOMAIN_HOME*/servers/WLS_REPORTS/stage/reports/configuration	*DOMAIN_HOME*/servers/WLS_REPORTS1/stage/reports/configuration
reports_ohs.conf	*ORACLE_INSTANCE*/config/OHS/ohs1/moduleconf	*ORACLE_INSTANCE*/config/OHS/ohs1/moduleconf

12.3.1.4 Oracle Reports Connection Retry

This section describes the following connection retries:

- Oracle Portal Database Connection Retry:

  If the connection from the Reports Server to the Oracle Portal database schema is dropped, the Reports Server tries to reestablish the connection before generating an error. If reconnection is successful, there is no need to restart the Reports Server.

- Oracle Internet Directory Connection Retry:

  If the Oracle Internet Directory connection becomes stale, the Reports servlet and the Reports Server try to reestablish the connection before generating errors. If reconnection is successful, there is no need to restart the Reports Server.

- Oracle Metadata Repository and Oracle Identity Management Outage:

  The outage of the Oracle Metadata Repository (which stores security metadata) does not bring down the Reports Server. If the Oracle Metadata Repository is unavailable, the Reports Server rejects new requests as a result of the component being unavailable. When the Oracle Metadata Repository is brought back on-line, the Reports Server recovers itself and begins to receive and process new requests.

  If Oracle Identity Management components become unavailable, the Reports Server also reject new requests, much like the outage of the Oracle Metadata Repository.

- Reports Server Timeout:

  The Reports Server has a configurable timeout for waiting for requests to be returned from the database. This timeout must be set to a high enough value to allow valid reports to run but not so high as to cause excessively long waits.

12.3.1.5 Oracle Reports Process Flow

The various components of Oracle Reports Services contribute to the process of running a report as follows:

1. The client requests a report by contacting a server through a URL (Web)
2. The Reports Server processes the request as follows:

   If the request includes a TOLERANCE option, the Reports Server checks its cache to determine whether it already has output that satisfies the request. If it finds acceptable output in its cache, then it immediately returns that output rather than rerunning the report.

   If the request is the same as a currently running job, it reuses the output from the current job rather than rerunning the report.

   If neither of these conditions is met:

   - If Oracle Reports Servlet (rwservlet) is SSO-enabled, it checks for authentication. A secure Reports Server then authorizes the user using Oracle Internet Directory. If Oracle Reports Servlet (rwservlet) is not SSO-enabled, a secure Reports Server authorizes and authenticates the user.
   - If the report is scheduled, the Reports Server stores the request in the scheduled job queue, and the report is run according to schedule. If the report is not scheduled, it is queued in the current job queue for execution when a Reports Engine becomes available.
3. At runtime, the Reports Server spawns a Reports Engine and sends the request to that engine to be run.

4. The Reports Engine retrieves and formats the data.
5. The Reports Engine populates the Reports Server cache.
6. The Reports Engine notifies the Reports Server that the report is ready.
7. The Reports Server accesses the cache and sends the report to output according to the runtime parameters specified in either the URL, the command line, or the keyword section in the `cgicmd.dat` file (URL requests only).

If a report server dies while running a request the request must be resubmitted. Once resubmitted, one of the surviving reports servers processes the request.

12.3.1.6 Oracle Reports Log Files

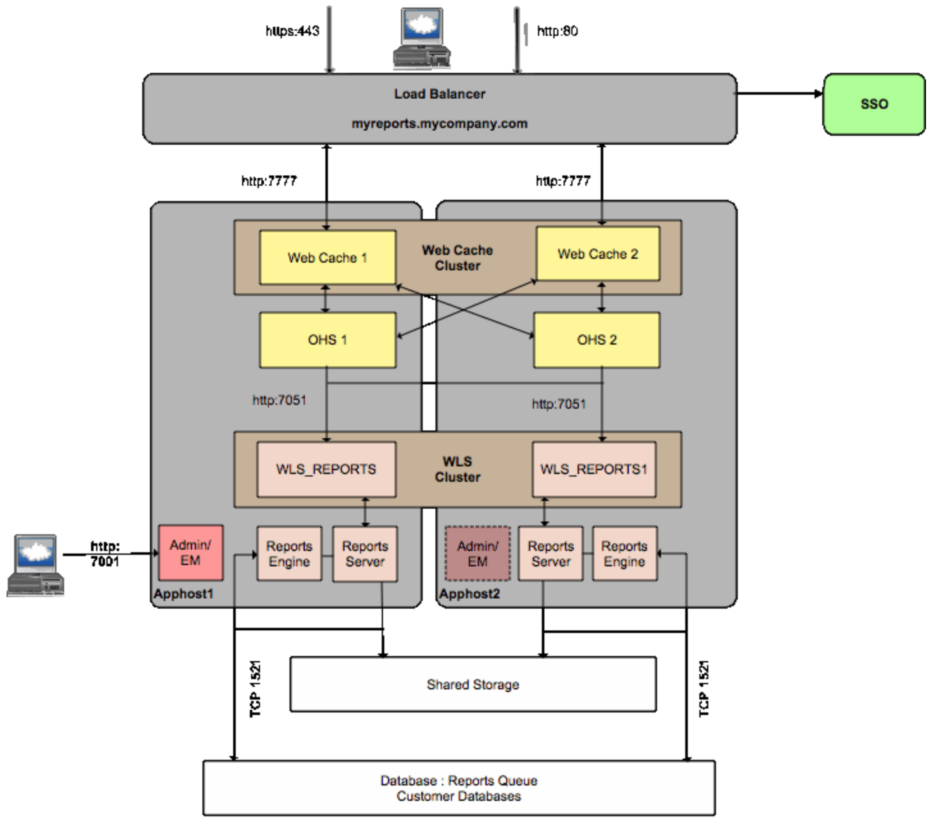
The following table shows log files used by Oracle Reports:

File	Location	Description
WLS_REPORTS.log	*DOMAIN_HOME*/servers/WLS_REPORTS/logs	Log file for the WebLogic Managed Server
WLS_REPORTS.out	*DOMAIN_HOME*/servers/WLS_REPORTS/logs	Supplemental log file for the WebLogic Managed Server. This is generally the first place to look for WebLogic issues.
rwservlet_diagnostic.log	*DOMAIN_HOME*/servers/WLS_REPORTS/logs	Log information relating to the Reports servlet.
rwserver_diagnostic.log	*DOMAIN_HOME*/servers/WLS_REPORTS/logs	Log information relating to the Reports server

12.3.2 Oracle Reports Protection from Failure and Expected Behavior

Figure 12–3 illustrates an example Oracle Reports high availability deployment.

Figure 12–3 Oracle Reports High Availability Deployment



As shown in Figure 12–3, the Oracle Real Application Clusters database provides a high availability repository for the reports queue. In high availability configurations, each Reports server has access to the same reports queue. A shared reports queue ensures that any request is only processed once, but it can be processed by any Reports server in the deployment.

Typically, reports are compiled using information stored in a variety of databases, the diagram above refers to these databases as customer databases.

In order to make full use of the Reports cache, the cache should be available to any of the Reports servers in the implementation. This way reports generated by one Reports server can be reused or served by any Reports server. In Figure 12–3, the Reports cache is made available using a shared directory.

Single Sign-On (SSO) is not required to restrict access to Oracle Reports, however, many organizations do restrict access to Oracle Reports using Oracle Single Sign-on. This chapter includes the steps required to protect the deployment using Oracle Single Sign On.

The in-process Reports server is recommended for high availability reports deployments. The in-process reports server runs inside Oracle WebLogic Server and the Reports servers themselves can be clustered to ensure that high availability is maintained.

When using Web Cache in a highly available distributed configuration, it is important that contents of all of the distributed caches are consistent. For example, the same cached copy of a given report is available from any Oracle Web Cache instance. When cached content becomes invalid, it is essential that it becomes invalid in all of the web caches. To achieve this goal web cache clustering is used.

The Diagram above includes the Web Logic Administration server and shows how it can be positioned on APPHOST2. To determine how to do this refer to the appropriate chapter elsewhere in this guide.

12.3.2.1 Oracle Reports Process Failures

If the Reports Server hangs or fails to respond, but the WebLogic Managed Server does not, the WebLogic Managed Server must be restarted manually.

12.3.2.2 Oracle Reports Node Failures

For information about Oracle Reports Node failure, see Section 12.1.3.2, "Common Component Node Failures."

12.3.2.3 Oracle Reports WebLogic Managed Server Failures

For information about Oracle Reports WebLogic Managed Server failure, see Section 12.1.3.3, "Common Component WebLogic Managed Server Failures."

12.3.2.4 Oracle Reports Database Failures

For information about Oracle Reports database failure, see Section 12.1.3.4, "Common Component Database Failures."

12.4 Oracle Forms and High Availability Concepts

This section describes single-instance information, as well as high availability concepts specific to Oracle Forms. This section guides you through the concepts and considerations necessary for creating a successful high availability Oracle Forms deployment.

12.4.1 Oracle Forms Single-Instance Component Characteristics

In the Oracle Fusion Middleware Forms Services architecture there is only one connection between the client and the HTTP Listener, much like any Web-based application. The HTTP Listener routes the request to the Forms Listener Servlet, which controls routing the requests from the Forms client to the Forms runtime.

The communication between the Forms client and the Forms runtime always goes through the HTTP Listener, leaving the application with only one port open to the network.

The client sends HTTP requests and receives HTTP responses from the HTTP Listener process. With the HTTP Listener acting as the network endpoint for the client, the other server machines and ports are not exposed at the firewall.

12.4.1.1 Oracle Forms State Information

Oracle Forms employs a stateful architecture. Each Runtime process keeps the state for the client it serves in working memory. No state is ever, or can ever be, serialized or shared between runtime processes

12.4.1.2 Oracle Forms Database Requirements

Oracle Forms only requires access to the databases with which it will interact. There are no special requirements for Oracle Forms itself.

12.4.1.3 Oracle Forms Request Flow

The Oracle Forms request flow is as follows:

1. The user chooses a Link from a Web page, or types a URL directly in the Browser.
2. The HTTP Listener interprets the URL that is passed and displays an HTML page containing an `<EMBED>` or `<OBJECT>` tag (depending on which browser is used) that describes the Forms Java Client to the Browser. The URL that is passed calls the Forms Servlet to create an HTML page dynamically based on the base.html files located on the web server.
3. The Client receives the HTML file served by the HTTP Listener. The tag in the HTML file supples the information required to locate the Java Class files that make up the Forms Java Client. Within the tag in the HTML file you supply information about the Form that should run, and any other parameters that you want to pass to your Forms session, such as the Login information. The tag definition also contains instructions on what Forms Services to run and many parameters which help to customize aspects of the Java Client, including the look-and-feel, and color schemes.

   The HTML file might also contain other HTML attributes such as those to tell the browser to run this particular applet using a particular version of the JRE on the client.
4. The Browser then asks the HTTP Listener for the Java Class files from the location specified in the HTML file. The CODEBASE parameter in the HTML file is used to define this. The files may be downloaded individually or as an "Archive". This archive has an extension of .JAR and can be best thought of as a .ZIP file containing all of individual CLASS files required by the Applet. The use of a JAR file speeds up the download of the Java Client and enables caching on the client for subsequent calls. The ARCHIVE parameter defines which (if any) .JAR file should be used. The JRE plug-in carries out the additional step of checking the version of the Forms Client Java code available on the HTTP Listener and only downloads it if it turns out to be newer that any version that the plug-in currently has cached.
5. The CLASS or JAR files are downloaded (if not already present) to the Browser and the Java applet starts.
6. The Java Client applet sends a request to start a Forms session through the HTTP Listener to the Forms Listener Servlet. The Forms Listener Servlet is defined by the serverURL parameter in the HTML file's tag.
7. After receiving the connection request from the Java Client, the Forms Listener Servlet starts a new Forms Runtime process for this client. You can define user specific environments for each runtime process by setting the Servlet initialization envFile parameter in the `formsweb.cfg` configuration file to a specific environment file.
8. The Forms Runtime process allocated to this 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 Client and the Forms Runtime process is passed through the Forms Listener Servlet.
9. The user is prompted for database login information, if this had not already been supplied, and the connection to the database server is established.
10. The user is now ready to work.

12.4.1.4 Oracle Forms Configuration Persistence

Oracle Forms uses several files for startup configuration:

- `formsweb.cfg` holds the Forms specific startup parameters for the runtime process that used to be expressed on the command line. It consists of a default section that is used if no specific section is utilized and one or more user sections that holds parameters specific to that section. These sections correspond to what is referred to as an application, a set of forms that belong to the same logical unit.

  `formsweb.cfg` is located in the following directory:

  ```
  DOMAIN_HOME/WLS_FORMS/stage/formsapp/11.1.1/formsapp/config
  ```

- `default.env` holds startup parameters that used to be expressed in environment variables. On Windows these variables can also be put in the system registry.

  This file is located in the same directory as `formsweb.cfg`:

  ```
  DOMAIN_HOME/WLS_FORMS/stage/formsapp/11.1.1/formsapp/config
  ```

  It is possible to create customized .env files with different names. If such files exist, they are specified in the `formsweb.cfg` and are thus discoverable.

- `base.html` and `basejpi.htm` or a customized version of either, holds the template structure for the HTML code that launches the Forms client applet.

  These files are co-located with the formsweb.cfg and the default.env files.

- `registry.dat` holds the default location and search paths for fonts, icons, and images. Find this file in:

  ```
  ORACLE_INSTANCE/config/FormsComponent/forms/registry/oracle/forms/registry
  ```

- `frmweb.res` holds key binding definitions for the Forms client. The bindings define the relationships between keyboard keys and the internal Forms functions they can trigger. For UNIX this bindings file is located at:

  ```
  ORACLE_INSTANCE/config/FormsComponent/forms/admin/resource/<language>
  ```

  Where `<language>` is a two character language denominator. For Windows this file is available at:

  ```
  ORACLE_INSTANCE/config/FormsComponent/forms
  ```

  On Windows, the language is defined by putting the language abbreviation at the end of the file name: `frmwebf.res` for French for example.

- `jvmcontroller.cfg` is the file for configuring JVM Controller(s). Any controllers configured for the system will have been defined in this file. This file is located in the following directory:

  ```
  ORACLE_INSTANCE/config/FRComponent/frcommon/tools/jvm/
  ```

12.4.1.5 Oracle Forms Runtime Considerations

The Oracle Fusion Middleware Forms Services is made up of three components: a Forms Client that is downloaded automatically to the end user's client machine and cached, the Forms Listener Servlet, and the Forms Runtime, on the middle tier.

Oracle Forms Client (Java Applet)

When a user runs a Forms session, the Forms Client, a thin 100 percent Java Applet, dynamically downloads from the Oracle Fusion Middleware Application Server. This generic Java Applet provides the classes for rendering the user interface for the associated Forms Runtime process on the middle tier, and handles user interaction and visual feedback, such as that generated by navigating between items or checking a checkbox. The same Java applet is used for all Forms application, therefore it is downloaded only once and cached on the client and so is available for subsequent Forms applications.

In order to run a Java applet in a browser, it is necessary to have a Java Virtual Machine (JVM) installed. The JVM is installed on the client and is platform dependent.

Oracle Forms Runtime Process

The Forms Runtime process is the process that maintains a connection to the database on behalf of the Forms Client. The process is created when a user accesses a page containing a Forms application. The process is automatically stopped as soon as the user closes the Forms application or terminates the browser window.

Oracle Forms Listener Servlet

The Forms Listener Servlet manages:

- The creation of a Forms runtime process for each client when a user requests to run a Forms application.
- The Forms Listener Servlet is also in charge of stopping the runtime process as the user closes the Forms application or terminates the browser window.
- Network communications between the client and its associated Forms runtime process

12.4.1.6 Oracle Forms Process Flow

The various components of Oracle Forms Services contribute to the process of running and Oracle Form as follows:

1. Client requests a form by contacting a server through a URL.
2. Oracle HTTP Server routes the request to Oracle WebLogic Server
3. Oracle WebLogic Server creates a forms Runtime process to run the form.

If a middle tier server crashes or a servlet session is interrupted, recover from either failure by restarting the application. A new Forms Runtime is created by doing so and any unsaved data can be reentered. The unsaved data does not cause database corruption since Forms uses atomic transactions which guarantees that database saves happen in an orderly and defined manner.

12.4.1.7 Oracle Forms Configuration Files

The following table shows configuration files used by Oracle Forms:

File	APPHOST1 Location	APPHOST2 Location
formsweb.cfg	*DOMAIN_HOME*/servers/WLS_FORMS/stage/formsapp/11.1.1/formsapp/config	*DOMAIN_HOME*/servers/WLS_FORMS/stage/formsapp/11.1.1/formsapp/config
default.env	*DOMAIN_HOME*/servers/WLS_FORMS/stage/formsapp/11.1.1/formsapp/config	*DOMAIN_HOME*/servers/WLS_FORMS/stage/formsapp/11.1.1/formsapp/config
base(jpi).htm	*DOMAIN_HOME*/servers/WLS_FORMS/stage/formsapp/11.1.1/formsapp/config	*DOMAIN_HOME*/servers/WLS_FORMS/stage/formsapp/11.1.1/formsapp/config

File	APPHOST1 Location	APPHOST2 Location
registry.dat	*DOMAIN_HOME*/config/FormsComponent/forms/registry/oracle/forms/registry	*ORACLE_INSTANCE*/config/FormsComponent/forms/registry/oracle/forms/registry
frmweb.res	UNIX: *ORACLE_INSTANCE*/config/FormsComponent/forms/admin/resource/<language> Windows: *ORACLE_INSTANCE*/config/FormsComponent/forms	UNIX: *ORACLE_INSTANCE*/config/FormsComponent/forms/admin/resource/<language> Windows: *ORACLE_INSTANCE*/config/FormsComponent/forms
jvmcontroller.cfg	*ORACLE_INSTANCE*/config/FRComponent/frcommon/tools/jvm/	*ORACLE_INSTANCE*/config/FRComponent/frcommon/tools/jvm/
forms_ohs.conf	*ORACLE_INSTANCE*/config/OHS/ohs1/moduleconf	*ORACLE_INSTANCE*/config/OHS/ohs1/moduleconf

12.4.1.8 Oracle Forms External Dependencies

Oracle Forms requires an Oracle HTTP Server to service requests. Optionally these requests may be cached by Oracle Web Cache, in which case it also acts as a load balancer.

Oracle Forms uses Sun's Java plugin (JRE) on the client to run the Forms Java Client.

12.4.1.9 Oracle Forms Log Files

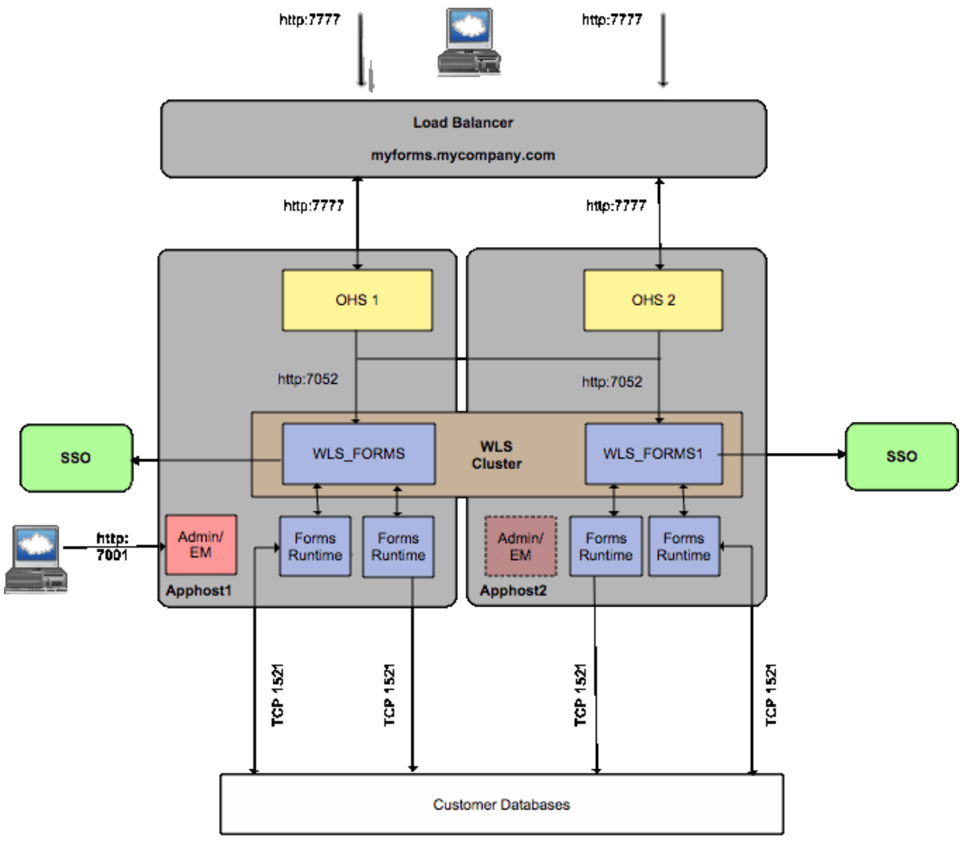
The following table shows log files used by Oracle Forms:

File	Location	Description
WLS_FORMS.log	*DOMAIN_HOME*/servers/WLS_FORMS/logs	Log file for the WebLogic Managed Server
WLS_FORMS.out	*DOMAIN_HOME*/servers/WLS_FORMS/logs	Supplemental log file for the WebLogic Managed Server. This is generally the first place to look for WebLogic issues.
Various	*ORACLE_INSTANCE*/FormsComponent/forms/trace	Forms trace files, generated in the event of a Forms runtime process crash. The filename has the format: <forms_runtime_process>_dump_<process_id> Forms runtime process does not write to a log file under normal operation
Various	*DOMAIN_HOME*/servers/WLS_FORMS/logs	Listener Servlet logs

12.4.2 Oracle Forms Protection from Failover and Expected Behavior

Oracle Forms has no serializable state, which means that transparent failover is not possible. If a runtime process fails, the client process served by that server process also fails. Oracle Forms employs atomic database transactions. With atomic transactions, a set of data, a record, or set of records, is either fully saved or not saved at all. As a result, the failure causes data not yet saved to be lost in a defined and precise manner. The data must be re-entered by the user when the application is restarted. If the process failed as a result of the machine it ran on failing, a substantial part, or all of the existing capacity to service the user base may be unavailable. In this scenario the application as a whole may be unavailable. To ensure continuity one option is to have redundant capacity either in an active-active or active-passive configuration.

Figure 12–4 Oracle Forms High Availability Deployment



12.4.2.1 Oracle Forms N+1 Redundancy

N+1 refers to an approach to redundant capacity that is based on the idea that hardware tends to break in units rather than in groups, meaning that a network of computers is more likely to lose one of its components rather than lose several at the same time. The principal of N+1 is to have as many machines as needed to service the entire user base at peak load plus one additional unit of equal capacity as the machine with the largest capacity in the set. This is often referred to as active-passive failover, since the failover happens on the machine level.

If you had six servers in your deployment that can handle the entire user base and are identically configured, they all have an HTTP Server and a Java Runtime and the Forms Runtime installed. There is a hardware (or software) load balancer to ensure that no single server is over utilized. The load balancer is aware of one machine labeled Standby, which has the same capacity as the machine with the largest capacity among the rest of the set, and is identically configured to all of them, but does not route to it. In an active-passive scenario, the standby machine doesn't have to be running. In an active-active scenario it could be an active part of the set, and provide spare capacity in an ongoing basis.

Now let's assume one of the machines fails:

If one of the machines fails, the standby machine is brought online and the load balancer has been reconfigured to route new requests to that machine as well, and to ignore the failed machine (some hardware load balancers do these two steps automatically). If the standby machine was already running and serving the user base in an active-active deployment, the downtime may be as small as the time it takes to restart the browser on the client. If the Standby machine was not running, in an active-passive deployment, and required must be started, and the load balancer had to be reconfigured manually the downtime would be longer.

12.4.2.2 Oracle Forms N+M Redundancy

For added failover capacity more than one standby machine can be used. That is usually referred to as N+M. The chance of more than one machine failing at the same time (or close to the same time) is significantly smaller, but if there is a requirement to be able to handle that situation, a N+M setup is possible and perhaps called for.

For redundancy in the case of a condition that affects all the machines in the deployment, perhaps a natural disaster that destroys all machines in one location, this deployment can be duplicated in two different geographical locations.

12.4.2.3 Oracle Forms Virtual Machines

In a datacenter that utilizes virtual machines, the standby machine can be brought online using any spare hardware and thus be implemented very cheaply.

12.4.2.4 Oracle Forms Configuration Cloning

The significance of cloning the environment becomes apparent when studying the scenarios described in the previous sections. All changes done to one machine's environment must also be present on all other machines, including the standby machine. This can present a practical problem, especially if the spare machine is virtual. The image that is the virtual machine must be kept in sync with the changes made to the other machines.

12.4.2.5 Oracle Forms Process Failures

For information about Oracle Forms process failure, see Section 12.1.3.1, "Oracle Web Cache and Oracle HTTP Server Process Failures."

12.4.2.6 Oracle Forms Node Failures

For information about Oracle Forms Node failure, see Section 12.1.3.2, "Common Component Node Failures."

12.4.2.7 Oracle Forms WebLogic Managed Server Failures

For information about Oracle Forms WebLogic Managed Server failure, see Section 12.1.3.3, "Common Component WebLogic Managed Server Failures."

12.4.2.8 Oracle Forms Database Failures

For information about Oracle Forms database failure, see Section 12.1.3.4, "Common Component Database Failures."

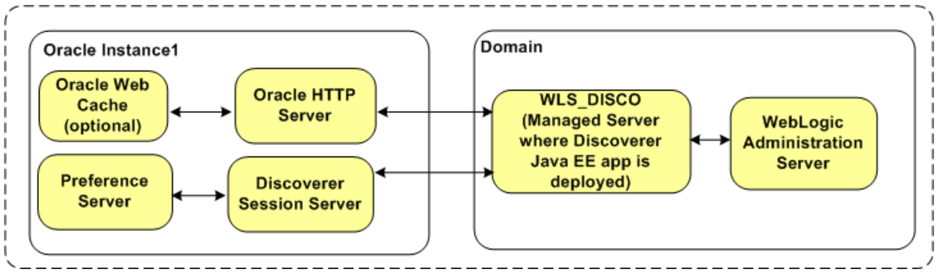
12.5 Oracle Discoverer and High Availability Concepts

This section describes single-instance information, as well as high availability concepts specific to Oracle Discoverer. This section guides you through the concepts and considerations necessary for creating a successful high availability Oracle Discoverer deployment.

12.5.1 Oracle Discoverer Single-Instance Characteristics

When you install Discoverer, you create a Discoverer topology within a single instance, as shown in Figure 12–5. After installation, you can configure other types of topologies for Discoverer.

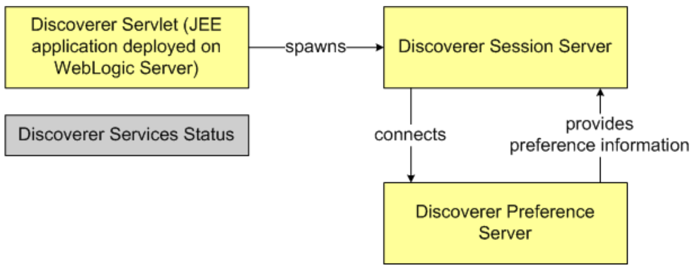
Figure 12–5 Discoverer Topology for a Single Instance


Oracle Instance1
Oracle Web Cache (optional)
Oracle HTTP Server
Preference Server
Discoverer Session Server
Domain
WLS_DISCO (Managed Server where Discoverer Java EE app is deployed)
WebLogic Administration Server

12.5.1.1 Oracle Discoverer Runtime Considerations

Figure 12–6 illustrates the runtime interaction between Discoverer components.

Figure 12–6 Runtime Interaction Between Discoverer Components


Discoverer Servlet (JEE application deployed on WebLogic Server)
spawns
Discoverer Session Server
Discoverer Services Status
connects
provides preference information
Discoverer Preference Server

The **Discoverer Servlet** is a Java EE application that is deployed on a managed server, which can be started and shutdown through the Oracle WebLogic Server administration console and by using the Oracle Enterprise Manager Fusion Middleware Control. The Discoverer Servlet is a stateless process.

The **Discoverer Session Server** is an OPMN-managed CORBA component that performs Discoverer operations such as connecting to the database or opening a workbook. The Session Server provides the link between the Discoverer Servlet and the database. There is one Session Server component per active user login session. The Discoverer Session Server is a stateful process; the state is stored only in memory.

The **Discoverer Preference Server** is an OPMN-managed component that provides preference settings (both default and user-defined) for all Discoverer users. These preferences control the Discoverer behavior. For information about starting and stopping the Preference Server, see the *Oracle Business Intelligence Discoverer Configuration Guide*.

The **Discoverer Services Status** is a dummy process managed by OPMN. This process must be running for the Session Server to be spawned.

Note: The Discoverer Servlet and Session Server must run on the same machine.

Oracle Discoverer External Dependencies

Discoverer requires an Oracle HTTP Server and Enterprise Manager Fusion Middleware Control.

The database schema for Discoverer and the portlet schema must be loaded (before installing Discoverer) by using the Repository Creation Utility (RCU).

Discoverer interacts with customer databases that contain Discoverer workbooks and the Discoverer End User Layer and other data sources. For more information, see the "About the Discoverer database tier" section in the *Oracle Business Intelligence Discoverer Configuration Guide*.

Oracle Web Cache can be used to configure load balancing for Discoverer.

The Portlet Provider component of Discoverer can provide portlets to Oracle Portal and Oracle Web Center.

The Web Services component of Discoverer can be used by external clients (such as Oracle BI Publisher and Oracle BI Enterprise Edition) to obtain Discoverer connections and workbooks.

It is recommended (but not mandatory) to restrict access to Oracle Discoverer by using Oracle Single Sign-On (SSO).

Discoverer Process Flow

See the "How does Discoverer work?" section in the *Oracle Business Intelligence Discoverer Configuration Guide*.

Connection Protocols

Client browsers send HTTP requests to the Discoverer Java EE application.

The Discoverer Java EE application accesses the database schema by using WebLogic Server data sources.

The Discoverer Session Server (non-Java EE component) uses the OCI layer to connect to data sources.

The Discoverer Java EE application and the Discoverer Session Server processes communicate by using CORBA.

12.5.1.2 Oracle Discoverer Viewer and Web Cache

You can improve Discoverer Viewer performance and availability by using Oracle Web Cache. For information about when and how to use Web Cache, see the "Using Discoverer Viewer with Oracle Web Cache" chapter in the *Oracle Business Intelligence Discoverer Configuration Guide*.

12.5.1.3 Oracle Discoverer Configuration Considerations

The environment and behavior of Oracle Discoverer are controlled by Discoverer preferences and configuration parameters.

- For information about the list of preferences and the procedure to change preference settings, see Managing Discoverer Preferences in the *Oracle Business Intelligence Discoverer Configuration Guide*
- Configuration parameters are stored in the `configuration.xml` file. For information about configuration parameters and how you can define settings for those parameters, see Managing and Configuring Discoverer in the *Oracle Business Intelligence Discoverer Configuration Guide*.

For information about the location of the files in which preferences and configuration parameters are stored, see the "Discoverer Configuration Files" section in the *Oracle Business Intelligence Discoverer Configuration Guide*.

12.5.1.4 Oracle Discoverer Deployment Considerations

The Discoverer Java EE application is an externally staged deployment and contains basic deployment descriptors.

- The `discoverer.ear` file consists of the following files: `application.xml`, `jazn-data.xml`, and `weblogic-application.xml`.

  These files are located in the `*DOMAIN*/servers/WLS_DISCO/stage/discoverer/11.1.1.1.0/discoverer/META-INF` directory, where *`DOMAIN`* is the name of the domain directory.
- The discoverer.war file contains the following files: `custom-laf.xml`, `plus_versions.properties`, `uix-config.xml`, `portlet.xml`, `weblogic.xml`, `struts-config.xml`, and `web.xml`.

  These files are located in the `*DOMAIN*/servers/WLS_DISCO/stage/discoverer/11.1.1.1.0/discoverer/discoverer.war/WEB-INF` directory, where *`DOMAIN`* is the name of the domain directory.

12.5.1.5 Oracle Discoverer Log File Locations

You can search for and view Discoverer log files in the Enterprise Manager Fusion Middleware Control.

For more information, see the Enterprise Manager Fusion Middleware Control online help.

12.5.1.6 Discoverer Log Files

The following table shows log files used by Oracle Discoverer:

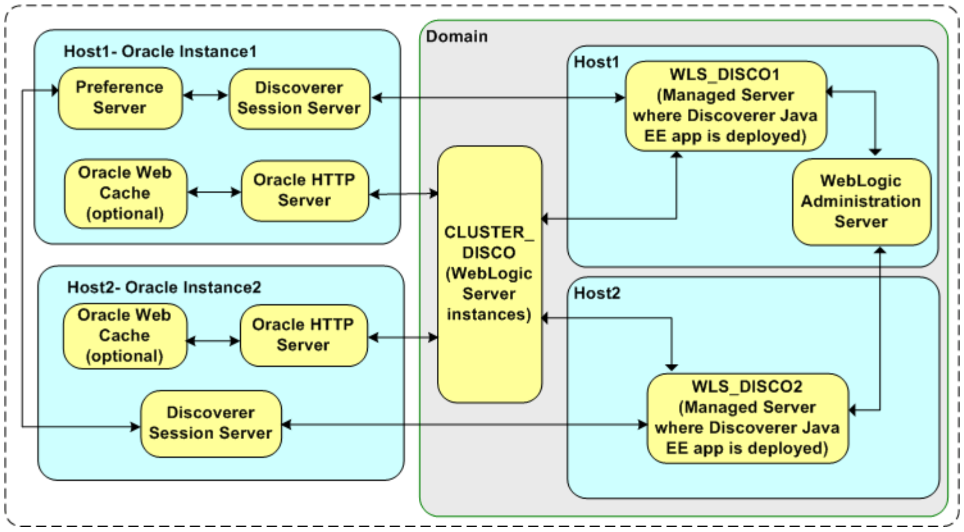
File	Location	Description
WLS_DISCO.log	*DOMAIN_HOME*/servers/WLS_DISCO/logs	Log file for the WebLogic Managed Server
WLS_DISCO.out	*DOMAIN_HOME*/servers/WLS_DISCO/logs	Supplemental log file for the WebLogic Managed Server. This is generally the first place to look for WebLogic issues.
Various	*ORACLE_INSTANCE*/diagnostics/logs/Discoverer/Discoverer_*Instance*	Supplemental Discoverer log files including Preference store.

12.5.2 Oracle Discoverer Protection from Failures and Expected Behavior

Figure 12–7 shows a Discoverer topology that consists of two Discoverer instances.

The Discoverer Java EE application in each managed server communicates with a single Preference Server. Therefore, the same preferences are applied to both the Discoverer instances. The Discoverer instances can exist either on the same machine or on different machines.

Figure 12–7 Discoverer Topology for Multiple Instances


Host1- Oracle Instance1
Preference Server
Discoverer Session Server
Oracle Web Cache (optional)
Oracle HTTP Server
Host2- Oracle Instance2
Oracle Web Cache (optional)
Oracle HTTP Server
Discoverer Session Server
Domain
CLUSTER_DISCO (WebLogic Server instances)
Host1
WLS_DISCO1 (Managed Server where Discoverer Java EE app is deployed)
WebLogic Administration Server
Host2
WLS_DISCO2 (Managed Server where Discoverer Java EE app is deployed)

The following sections discuss specific high-availability considerations relevant for Discoverer. After making configuration changes to provide high availability, you must restart the Oracle HTTP Server and the managed servers in which the Discoverer Java EE applications are deployed.

12.5.2.1 Preference Server Failover

Failure detection and restart of the Discoverer Java EE application in a cluster is managed by WebLogic Server.

The Preference Server is a singleton process and runs only one instance in the entire cluster.

If the Preference Server process goes down, OPMN brings it up again automatically. If OPMN too is down, the administrator must either start the Preference Server manually or move the files containing the preferences (`reg_key.dc` and `pref.txt`) to another machine where the Preference Server is configured and then start the Preference Server on that machine.

If the Preference Server is started in another node in the cluster, the Discoverer Java EE application must be configured to recognize the new Preference Server.

12.5.2.2 Session State Replication and Failover

When a server or machine that is handling requests pertaining to a particular Discoverer HTTP session is down, subsequent request are diverted to other managed servers in the cluster. The session state can be replicated in the new server. The necessary information to achieve this replication is available in the HTTP request (as GET/POST).

Note: Discoverer requires a Session Server (C++ process) to be available to complete any request. Therefore, in a failover situation, the response time would be marginally higher because a C++ process must be spawned and brought to the required state.

12.5.2.3 Performance Recommendation

The Discoverer Java EE Servlet spawns a Session Server process for each HTTP request that it receives. If multiple Discoverer instances (managed server and the associated oracle instance) exist in a single machine, the load on the CPU might become very high because all Discoverer instances might spawn C++ processes simultaneously. Adding Discoverer instances on different machines yields better performance than adding them on the same machine.

12.5.2.4 Propagation of Configuration Changes Across the Cluster

Any configuration change must be implemented individually in each managed server in the cluster. You can achieve this by copying the `configuration.xml` file to all the managed servers.

For information about the location of the configuration files, see the "Discoverer Configuration Files" section in the *Oracle Business Intelligence Discoverer Configuration Guide*.

12.5.2.5 Cluster-Wide Application Deployment

Each Discoverer Java EE application instance (managed server) should be associated with an Oracle instance that contains Discoverer components. These components are created during the configuration phase of the installation process. So adding managed servers to a cluster and deploying the Discoverer Java EE application from the WebLogic Server administration console are not supported.

All the managed servers defined in the cluster are targets for deployment of the Discoverer Java EE application.

12.5.2.6 Online Application Deployment

Most of the Discoverer Java EE application-dependent jar files are available through shared libraries. Patching these shared libraries does not require the Discoverer Java EE application to be restarted.

Changes to the following jar files, which are available through the system classpath, require the Discoverer Java EE application to be restarted.

```
WL_HOME/server/lib/weblogic.jar
ORACLE_HOME/modules/oracle.jrf_11.1.1/jrf.jar
ORACLE_HOME/opmn/lib/nonj2eembeans.jar
ORACLE_HOME/jdbc/lib/ojdbc6.jar
ORACLE_HOME/opmn/lib/optic.jar
ORACLE_HOME/opmn/lib/iasprovision.jar
ORACLE_HOME/opmn/lib/ons.jar
ORACLE_HOME/modules/oracle.adf.share_11.1.1/oracle-el.jar
ORACLE_HOME/jlib/share.jar
ORACLE_HOME/jlib/jewt4.jar
```

12.5.2.7 Oracle Discoverer Process Failures

For the Discoverer Java EE application to access databases, an entry for each database must be created in the `ORACLE_INSTANCE/config/tnsnames.ora` file. For more information, see the "About configuring the tnsnames.ora file" section in the *Oracle Business Intelligence Discoverer Configuration Guide*.

12.5.2.8 Oracle Discoverer Node Failures

For information about Oracle Discoverer Node failure, see Section 12.1.3.2, "Common Component Node Failures."

12.5.2.9 Oracle Discoverer WebLogic Managed Server Failures

For information about Oracle Discoverer WebLogic Managed Server failure, see Section 12.1.3.3, "Common Component WebLogic Managed Server Failures."

12.5.2.10 Oracle Discoverer Database Failures

For information about Oracle Discoverer database failure, see Section 12.1.3.4, "Common Component Database Failures."

12.6 Configuring Oracle Portal, Forms, Reports, and Discoverer for High Availability

This section describes procedures for configuring Oracle Portal, Forms, Reports, and Discoverer for a high availability deployment. This section contains information on the following topics

Note: Oracle strongly recommends reading the release notes for any additional installation and deployment considerations prior to starting the setup process.

- Section 12.6.1, "Prerequisites"
- Section 12.6.2, "Assumptions"
- Section 12.6.3, "Creating the Metadata Repository"
- Section 12.6.4, "Install and Configure Application Tier on APPHOST1"
- Section 12.6.5, "Install and Configure Application Tier on APPHOST2"
- Section 12.6.6, "Scaling Out the Deployment"

Note: The hostnames and ports listed in these sections are, for illustrating this example deployment. When configuring your actual deployment, you can specify your own hostnames and ports.

12.6.1 Prerequisites

Before installing Oracle Portal, Forms, Reports and Discoverer, review the prerequisites described in this section.

12.6.1.1 Dependencies

If you are using Oracle Single Sign-On, it is assumed that a high availability Identity Management Framework is available. Oracle Portal, Forms, Reports, and Discoverer uses Single Sign-On, which is available in Oracle Identity Management 10*g*.

12.6.1.2 Network Requirements

This section describes network requirements.

12.6.1.2.1 Load Balancer In order to distribute requests across the Oracle Web servers, a load balancer is required. This external load balancer should have the following features:

- Virtual server name and port configuration
- Process failure detection
- Monitoring of ports (HTTP, HTTPS) for Oracle HTTP and HTTPS
- SSL Translation (if required)

12.6.1.2.2 Load Balancer Configuration - Virtual Server Names and Ports If you are using a load balancing router, it must be configured to enable the following:

If using SSL traffic terminated at the load balancer

A virtual IP address (VIP1) that listens for requests to `mysite.mycompany.com` on port 443 (an HTTPS listening port), and balances them to the application tier Oracle Web Cache, running on WEBHOST1 and WEBHOST2, port 7777 (an HTTP listening port). You must configure the Load Balancing Router to perform protocol conversion.

If using site SSL

A virtual IP address (VIP1) that listens for requests to `mysite.mycompany.com` on port 443 (an HTTPS listening port), and balances them to the application tier, Oracle Web Caches running on WEBHOST1 and WEBHOST2 port 443 (an HTTPS listening port).

If using HTTP traffic terminated at the load balancer

A virtual IP address (VIP1) that listens for requests to mysite.mycompany.com on port 80 (an HTTP listening port), and balances them to the application tier, Oracle Web Caches running on WEBHOST1 and WEBHOST2 port 7777 (an HTTP listening port).

Web Cache

- The virtual IP address VIP1 listens for requests to mysite.mycompany.com on port 9401 (an HTTP listening port), and balances them to the application tier Oracle Web Cache on WEBHOST1 and WEBHOST2 port 9401 (an HTTP listening port).

> **Note:** For security reasons, ports 9401, 9402 and 9403 on the load balancing router should not be visible to external users.

- HTTP/HTTPS monitoring of Oracle Web Cache: The Load Balancing Router must be configured to detect an inoperative computer and stop routing requests to it until it is functioning again. Two Oracle Web Cache ports must be monitored: the HTTP request port and the invalidation port.

To monitor port 7777, use the following URL in the Load Balancing Router configuration:

```
hostname:port/_oracle_http_server_webcache_static_.html
```

For example:

```
http://webhost1.mycompany.com:7777/_oracle_http_server_webcache_static_.html
```

If the Load Balancing Router receives a response from this URL, then the Oracle Web Cache instance is running. If it does not receive a response, the process or the server is down, and the load balancing router forwards all requests to the active computer.

To monitor port 9401, use the following URL in the Load Balancing Router configuration:

```
http://hostname.domain.com:9401
```

For example:

```
http://apphost1.mycompany.com:9401
```

The load balancing router sends an HTTP request to this URL; the response header resembles the following:

```
HTTP/1.0
```

The load balancing router must be configured to detect the string HTTP in the first line of the response header. Therefore, when the load balancing router detects HTTP in the first line of the response header, the invalidation port is available. If it does not, all invalidation requests are routed to the active computer.

Note: The `sqlnet.ora` file must be updated to prevent connection timeouts related to the load balancing router and firewall.

To summarize, the load balancer requires the following configuration:

Virtual Host	Virtual Port	Server Pool	Server	Port	Comments
mysite.mycompany.com	443/80	UserRequest	WEBHOST1	7777	Protocol conversion required if terminating SSL at the load balancer
mysite.mycompany.com	443/80	UserRequest	WEBHOST2	7777	Protocol conversion required if terminating SSL at the load balancer
mysite.mycompany.com		Cache Invalidation	WEBHOST1	9401	Invisible to the external clients

Virtual Host	Virtual Port	Server Pool	Server	Port	Comments
mysite.mycompany.com		Cache Invalidation	WEBHOST2	9401	Invisible to the external clients

12.6.1.3 Databases

Different products use databases in different ways. Below is a summary of the database requirements for Oracle Portal, Forms, Reports and Discoverer:

Oracle Portal

Oracle Portal stores both its own metadata and user content in a database. The majority of the Portal application logic is also placed into this database in the form of PLSQL. Using a combination of the metadata, user content, and PLSQL, Portal generates Web pages.

Oracle Portal requires a high availability database, which has been pre-seeded with database objects required by the Portal application. For information about creating the Meta Data Repository, see Section 12.6.3, "Creating the Metadata Repository."

Oracle Forms

Oracle Forms interacts with the database and therefore requires access to it. There are no special database requirements of Oracle Forms itself.

Oracle Reports

Oracle Reports requires a highly available database, which contains the reports queue. It also requires access to various customer databases, which it uses to compile report content.

Oracle Discoverer

Oracle Discoverer requires a database for portlets. This should be a high availability database, and should be seeded using the Oracle Repository Creation Utility (RCU).

In addition, Oracle Discoverer interacts with a number of customer databases. These databases are used to store Discoverer End User Layers, and work books as well as the data on which it reports.

12.6.1.4 Shared Directories

Shared directory requirements vary depending on the product. These requirements are described in the following table:

Product	Shared Directory Requirement
Oracle Portal	None
Oracle Forms	Not mandatory, but useful to have a shared directory for Oracle Forms executables.
Oracle Reports	Reports Cache
Oracle Discoverer	None

12.6.1.5 Managed Port Numbers

Many Oracle Fusion Middleware components and services use ports. As an administrator, it is important to know the port numbers used by these services, and to ensure that the same port number is not used by two services on the same host.

Port numbers can either be automatically or manually assigned at installation time.

12.6.1.6 Site Names

In order to configure a Web site a site name is required. This site name, for example mysite.mycompany.com, must be defined in DNS and be associated with the Virtual IP address assigned to the load balancer.

A site name is also required for the Single Sign-On server, which is set up as part of the high availability Single Sign-On installation.

12.6.2 Assumptions

For the example high availability configuration described in this chapter, consider the following assumptions:

12.6.2.1 Ports

The following table lists the typical ports required by an Oracle Portal, Forms, Reports, and Discoverer implementation.

Purpose	Host(s)	Port	Comment
mysite.mycompany.com	Load balancer	443	SSL port on the load balancer
mysite.mycompany.com	Load balancer	80	HTTP port on the load balancer
Web Cache HTTP	APPHOST1 APPHOST2	7777	Web Cache HTTP port
Web Cache HTTPS	APPHOST1 APPHOST2	4443	Web Cache HTTPS port
Web Cache Invalidation	APPHOST1 APPHOST2	9401	Web Cache invalidation port
Web Cache Admin	APPHOST1 APPHOST2	9400	Web Cache Administration port
HTTP Server (OHS) - HTTP	APPHOST1 APPHOST2	7778	OHS HTTP listening port
HTTP Server (OHS) - HTTPS	APPHOST1 APPHOST2	4444	OHS HTTPS listening port
WebLogic Admin port	APPHOST1	7001	WebLogic Administration Server port
WLS_PORTAL	APPHOST1	7050	WebLogic Managed Server port
WLS_PORTAL1	APPHOST2	7050	WebLogic Managed Server port
WLS_REPORTS	APPHOST1	7051	WebLogic Managed Server port

Purpose	Host(s)	Port	Comment
WLS_REPORTS1	APPHOST2	7051	WebLogic Managed Server port
WLS_DISCO	APPHOST1	7052	WebLogic Managed Server port
WLS_DISCO1	APPHOST2	7052	WebLogic Managed Server port
WLS_FORMS	APPHOST1	7053	WebLogic Managed Server port
WLS_FORMS1	APPHOST2	7053	WebLogic Managed Server port
Internet Directory	SSOHOST	389	Oracle Internet Directory HTTP port
Single Sign On	SSOHOST	7777	Single Sign On listening port

12.6.3 Creating the Metadata Repository

Before installing Oracle Portal and Oracle Discoverer, create and "seed" a repository with metadata that these applications require to function, by running the Repository Creation Utility (RCU).

Note: A metadata repository is not required for Oracle Forms and Reports.

12.6.3.1 Install the Repository Creation Utility (RCU)

Install the Repository Creation Assistant as described in the *Oracle Fusion Middleware Repository Creation Utility User's Guide*.

12.6.3.2 Run Repository Creation Utility

Having installed the RCU use it to populate the database. Start the Repository Creation Utility using the following commands:

For UNIX:

```
rcu
```

For Windows:

```
rcu.exe
```

These commands are located in the RCU `ORACLE_HOME/bin` directory.

1. In the Create Repository screen, select **Create** and then click **Next**.
2. In the Database Connection Details screen, specify the following values:
   - **Database Type**: `Oracle Database`
   - **Host Name**: Enter *one* of the RAC nodes (use the VIP name), if using Oracle RAC.
   - **Port**: Enter the listener port.
   - **Service Name**: Enter the service name of the Oracle Real Application Clusters database.

- **User Name**: Enter `sys`.
- **Password**: Enter the `sys` user password.
- **Role**: Select `SYSDBA`.

3. In the Check Prerequisites screen, click **OK** when the prerequisites have been validated.
4. In the Select Components screen, specify the following values:
   - **Create New Prefix**: Enter a prefix to be added to database schemas, for example: `MYS`
   - **Components**: `Portal and BI > Portal and Discoverer`

   Click **Next**.
5. In the Check Prerequisites screen, click **OK** when the prerequisites have been validated.
6. In the Schema Passwords screen, enter passwords for each of the portal schemas or use the same password for all schemas.

   Click **Next**.
7. In the Map Tablespaces screen, click **Next** to accept the defaults.
8. In the Create Tablespaces screen, click **Yes** to allow RCU to create any missing tablespaces.
9. In the Creating Tablespaces screen, click **OK** to acknowledge tablespace creation.
10. In the Summary screen, click **Create** to begin the creation process.

12.6.4 Install and Configure Application Tier on APPHOST1

The following steps are applicable for Oracle Portal, Forms, Reports and Discoverer installations.

- Section 12.6.4.1, "Install Oracle WebLogic Server"
- Section 12.6.4.2, "Install Oracle Portal, Forms, Reports, and Discoverer Software"
- Section 12.6.4.3, "Validation"
- Section 12.6.4.4, "Generic Configuration"
- Section 12.6.4.5, "Configure Oracle Portal for High Availability"
- Section 12.6.4.6, "Configure Oracle Forms for High Availability"
- Section 12.6.4.7, "Configure Oracle Reports for High Availability"
- Section 12.6.4.8, "Configure Oracle Discoverer for High Availability"

12.6.4.1 Install Oracle WebLogic Server

To install Oracle WebLogic Server binaries, see the *Oracle Fusion Middleware Installation Guide for Oracle WebLogic Server*.

12.6.4.2 Install Oracle Portal, Forms, Reports, and Discoverer Software

Install the Oracle binaries into the FMW_HOME created in the previous section.

For UNIX:

```
runInstaller
```

For Windows:

```
seutp.exe
```

Note: Before starting the install ensure that the following environment variables (UNIX) are not set:

- LD_ASSUME_KERNEL
- ORACLE_BASE
- LD_LIBRARY_PATH

Continue with the following steps:

1. In the Welcome screen, click **Next**.
2. In the Installation Type screen, choose **Install Software and Configure**, then click **Next**.
3. In the Prerequisite Checks screen, once all the prerequisites have been validated, click **Next**.
4. In the Create Domain screen, select **Create New Domain** and enter these values:
   - **User Name**: Name of the user to log into the WebLogic domain.
   - **User Password**: Password for the domain.
   - **Confirm Password**: Type the same password again.
   - **Domain Name**: Name for the domain. For example: `MyDomain`

   Click **Next**.
5. In the Specify Oracle Configuration Manager Details screen, if required, enter a username and password to receive Oracle Security updates for Oracle Support.
6. In the Specify Installation Location screen, enter these values:
   - **Middleware Home**: Enter the value for the FMW_HOME. For example:

     ```
     /u01/app/oracle/product/FMW
     ```

   - Oracle Home: Enter the installation directory. Note that this directory is placed under the FMW_HOME directory. For example:

     ```
     FMWHome
     ```

   - **WebLogic Server Directory**: Enter the installation directory for Oracle WebLogic Server. This directory should be `FMW_HOME/wlserver_10.3`, for example:

     ```
     /u01/app/oracle/product/FMW/wlserver_10.3
     ```

   - **Oracle Instance Location**: Enter the directory where the Oracle configuration files are to be placed. This should be outside of the Oracle home directory. This directory will be known as the ORACLE_INSTANCE. For example:

     ```
     /u01/app/oracle/admin/fmw1
     ```

   - **Oracle Instance Name**: `fmw1`

Click **Next**.

7. In the Configure Components screen, choose which products to install and configure.

   Ensure that **Management Components - Enterprise Manager** is also selected.

   Select **Clustered**.

   Click **Next**.

8. In high availability implementations, it is not mandatory for all of the ports used by the various components to be are synchronised across hosts, but Oracle strongly recommends it. Oracle Allows the bypassing of Automatic Port Configuration by specifying ports to be used in a file.

   Create a file with the ports you wish to assign by Select a file name, and click **View/Edit**.

   You can find a sample `staticports.ini` file on installation Disk1 in the `stage/Response` directory.

   This file should have entries for the following:

   ```
   [Domain Port No = 7001
   Oracle HTTP Server Port No = 7778
   WebCache Port No = 7777
   WebCache Administration Port No = 9400
   WebCache Statistics Port No = 9402
   Web Cache Invalidation Port = 9401
   Portal Managed Server Port = 7050
   Reports Managed Server Port = 7051
   Discoverer Managed Server Port = 7052
   Forms Managed Server Port = 7053]
   ```

   Save the file and click **Next**.

9. In the Specify Proxy Details screen (Reports only), if a proxy server is in use, enter the details in this screen.

   Otherwise, click **Next**.

10. In the Specify Schema screen (Portal and Discoverer only), specify the following values:

    - **Database Connect String** in the format:

      For an Oracle RAC database:

      ```
      racnode1-vip:ListenerPort:racnode2-vip:ListenerPort@mydb.mycompany.com
      ```

      For other databases:

      ```
      host:port:mydb.mycompany.com
      ```

    - **Portal Schema Name**: `MYS_PORTAL`
    - **Portal Schema Password**: Enter password entered in RCU.
    - **Discoverer Schema Name**: `MYS_DISCOVERER`
    - **Discoverer Schema Password**: Enter password entered in RCU.

    Click **Next**.

11. In the Specify Portlet Schema screen (Portal and Discoverer only), specify the following values:

- **Portlet Schema Name**: `MYS_PORTLET`
- **Portlet Schema Password**: Enter password entered in RCU.

Click **Next**.

12. In the Specify Application Identity Store screen, specify the following values:
    - **Hostname**: Name of the Oracle Internet Directory server, for example: `myoid.mycompany.com`
    - **Port**: Oracle Internet Directory port, for example: `389`
    - **User Name**: `cn=orcladmin`
    - **Password**: Password for Oracle Internet Directory's `orcladmin` account
13. In the Summary screen, click **Install** to begin the creation process.

12.6.4.3 Validation

Validate the initial installation by performing the following tests:

Test	URL	Result
Portal	http://apphost1.mycompany.com:7777/portal/pls/portal	Portal Home page is displayed
Forms	http://apphost1.mycompany.com:7777/forms/frmservlet	Forms Servlet Home page is displayed
Reports	http://apphost1.mycompany.com:7777/reports/rwservlet/showjobs	Job Queue is displayed
Discoverer	http://apphost1.mycompany.com:7777/discoverer/viewer	Discoverer Viewer Home page is displayed

12.6.4.4 Generic Configuration

The following steps are required regardless of the product being configured:

12.6.4.4.1 Configure Virtual Hosts For the site to function properly with the load balancer, you must add two virtual hosts. You can configure the virtual hosts using Oracle Enterprise Manager Fusion Middleware Control, or you can edit the file *ORACLE_INSTANCE*/config/OHS/ohs1/httpd.conf or create a file called virtual_hosts.conf in the directory *ORACLE_INSTANCE*/config/OHS/ohs1/moduleconf.

To add two virtual hosts to the `virtual_hosts.conf` file, in a text editor, add the following entries to the file:

```
NameVirtualHost *:7778
<VirtualHost *:7778>
    ServerName https://mysite.mycompany.com:443 ** If using SSL Termination/site
at LB
    ServerName http://mysite.mycompany.com:80 ** If not using SSL
    RewriteEngine On
    RewriteOptions inherit
    UseCanonicalName On
</VirtualHost>

<VirtualHost *:7778>
    ServerName apphost1.mycompany.com:7777
    RewriteEngine On
    RewriteOptions inherit
    UseCanonicalName On
</VirtualHost>
```

Note: Where:

7778 is the Oracle HTTP Server Listening Port

443/80 are Load Balancer Listening Ports

7777 is the Web Cache Listening Port.

Name of the server on which the Oracle HTTP server resides: apphost1

Use ServerName HTTPS if your site is SSL enabled or you are terminating SSL at the Load balancer.

Use ServerName HTTP if your site works only in the http protocol.

Restart the Web Tier using these commands:

```
opmnctl stopall
opmnctl startall
```

12.6.4.4.2 Create boot.properties File Create a `boot.properties` file for the administration server on APPHOST1. The `boot.properties` file enables the administration server to start without prompting you for the administrator username and password.

In a text editor, create a file called `boot.properties` in the directory *DOM_HOME*/servers/AdminServer/security, and enter the following lines in the file:

```
username=adminuser
password=password
```

Restarting the administration server encrypts the values in this file, for that reason it is recommended that the administration server is restarted on each node that can host it.

Stop the administration server using the script `stopWebLogic.sh`, which is located in *DOMAIN_HOME*/bin, or by using the script `startWebLogic`, also located in *DOMAIN_HOME*/bin.

12.6.4.4.3 Configure sqlnet.ora Create a file called `sqlnet.ora` in the directory `ORACLE_INSTANCE/config/` and add the following entry to the file:

```
TCP.CONNECT_TIMEOUT=10
```

This ensures that database connections time out after a reasonable time.

12.6.4.4.4 Configure Web Cache Web Cache is optional for Oracle Forms, Reports and Discoverer, but mandatory for Oracle Portal. If the environment being configured does not use Web Cache, ignore the following Web Cache configuration steps:

Log Into Enterprise Manager Console

Log into the Oracle Fusion Middleware Enterprise Manager Console using the following URL:

```
http://apphost1.mycompany.com:7001/em
```

The default **User Name** and **Password** are the same as the WebLogic domain username and password entered during the installation.

Create Site

1. In the Navigator window, expand the Web Tier tree.
2. Click on the component `wc1`.
3. From the list at the top of the page, select **Administration - Sites**.
4. Select **Create Site**.
5. Enter the following information to add a site.

Field	Value
Host Name	mysite.mycompany.com
Port	Specify the port number on the load balancer where requests are received, for example, 80 for HTTP requests and 443 for HTTPS request. If using a load balancer, this is the listening port on the load balancer.
Default site	Yes
Site Wide Compression	Yes
Site Alias - Host Name	mysite.mycompany.com
Site Alias - Port	7777
Site Alias - Host Name	mysite.mycompany.com
Site Alias - Port	80 [1]

[1] Required when the loadbalancer can send requests to port 80 as well as port 443.

6. Do not change any other defaults.
7. Select **Submit**.
8. Select **OK** to save each entry.

Create Site to Server Mapping

1. On the same page, select **Create** in the Site-to-Server Mapping section.
2. Enter the following information to add the site:

Field	Value
Host Pattern	mysite.mycompany.com
Port Pattern	443 or 80, depending on whether SSL is being used or not.
Selected Origin Servers	apphost1.mycompany.com:7778

3. Click **OK** to store the site.

4. Ensure that the site mysite.mycompany.com:443/80 appears first in the list of site-to-server mappings.
5. Click **Apply** to save the changes.

Enable Session Binding

The session binding feature in Oracle Web Cache is used to bind user sessions to a given origin server to maintain state for a period of time. Although almost all components running in a default Oracle Fusion Middleware mid-tier are stateless, session binding is required for Oracle Portal.

Enabling session binding forces all the user requests to go to a specific Oracle Portal middle tier, resulting in a better cache hit ratio for the portal cache.

Follow these steps to enable session binding:

1. Select **Administration - Session Configuration** at the top of the page.
2. Select the site `mysite.mycompany.com:443/80` from the drop down list.
3. In the Session Binding section, select **Cookie based Session Binding with any Set Cookie**.
4. Click **Apply** to save the changes.

12.6.4.4.5 Change the Web Cache Passwords Web Cache passwords are randomly generated, however they are required in later stages. It is therefore recommended that the Web Cache passwords be changed from the default value to a new known value.

To change the default password, follow these steps:

1. In the Navigator window, expand the **Web Tier** tree.
2. Click on the component `wc1`.
3. From the drop down list at the top of the page, select **Administration - Passwords**.
4. Enter new invalidation and administration passwords, confirm them, and click **Apply**.

12.6.4.4.6 Restart Web Tier (Oracle HTTP Server and Web Cache) After making the previous changes, restart the Web Tier components using these commands:

```
opmnctl stopall
opmnctl startall
```

12.6.4.4.7 Register with Single Sign On Server Perform these steps from the Single Sign-On server:

1. Set the ORACLE_HOME variable to the Single Sign-On Server ORACLE_HOME location.
2. Execute ORACLE_HOME/sso/bin/ssoreg.sh (ssoreg.bat on Windows) with the following parameters:

   Site SSL/Termination:

   ```
   -site_name mysite.mycompany.com
   -mod_osso_url https://mysite.mycompany.com
   -config_mod_osso TRUE
   -oracle_home_path ORACLE_HOME
   -config_file /tmp/osso.conf
   -admin_info cn=orcladmin
   ```

```
-virtualhost
-remote_midtier
```

No SSL:

```
-site_name mysite.mycompany.com
-mod_osso_url http://mysite.mycompany.com
-config_mod_osso TRUE
-oracle_home_path ORACLE_HOME
-config_file /tmp/osso.conf
-admin_info cn=orcladmin
-virtualhost
-remote_midtier
```

3. Copy `/tmp/osso.conf` to the mid-tier home location, which is:

   ```
   ORACLE_INSTANCE/config/OHS/ohs1
   ```

4. Restart Oracle HTTP server on APPHOST1 using the following command:

   ```
   opmnctl restartproc process-type=OHS
   ```

5. Log into the Single Sign-On Server using the following URL:

   ```
   http://login.mycompany.com/pls/orasso
   ```

6. Go to the Administration page and then Administer Partner applications. Delete the entry for `apphost1.mycompany.com`.

12.6.4.4.8 Change Host Assertion in WebLogic Because the Oracle HTTP server acts as a proxy for WebLogic, by default certain CGI environment variables are not passed through to WebLogic. These include the host and port. WebLogic must be told that it is using a virtual site name and port so that it can generate internal URLs appropriately.

1. Log into the Oracle WebLogic Server Administration Console using the following URL:

   ```
   http://apphost1.mycompany.com:7001/console
   ```

2. Select **Clusters** from the home page or choose **Environment -> Clusters** from the `Domain structure` menu.
3. Click **Lock and Edit** in the Change Center window to enable editing.
4. Click on the Cluster Name (`cluster_portal`, `cluster_forms`, `cluster_reports`, `cluster_disco`).
5. Select **HTTP** and enter the following values:

Parameter	Value
Frontend Host	mysite.mycompany.com
Frontend HTTP Port	80
Frontend HTTPS Port	443

   This ensures that any HTTPS URLs created from within WebLogic are directed to port 443 or 80 on the load balancer.

6. Click **Activate Changes** in the Change Center window to save the changes.

7. Restart the WLS_PORTAL, WLS_FORMS, WLS_REPORTS, WLS_DISCO Managed Servers using the following steps:

   a. Select **Servers** from the Home page or **Environment > Servers** from the **Domain structure** menu.

   b. Select the **Control** tab.

   c. Select the box next to each Managed Server.

   d. Choose **Shutdown > Force Shutdown Now**.

   e. Click **Yes** to shut down the Managed Server.

   f. Once the Managed Server is shut down, select the box next to the Managed Server.

   g. Click **Start**.

   h. Click **Yes** to start the Managed Server.

12.6.4.5 Configure Oracle Portal for High Availability

The following steps are required to configure Oracle Portal only.

12.6.4.5.1 Rewire Portal Repository Follow these steps to rewire the Oracle Portal repository:

1. Log into the domain using Oracle Fusion Middleware Enterprise Manager using the following URL:

   ```
   http://apphost1.us.oracle.com:7001/em
   ```

2. Expand the **Fusion Middleware** menu on the left hand side.

3. Expand the **Portal** menu (under **Fusion Middleware** menu).

4. Click **Portal**.

   The Portal Domain information page is displayed.

5. Right-click **Portal** and select settings **Wire Configuration**.

6. Enter the following information for Portal midtier:

Parameter	Value
Host	Enter the DNS name of the load balancer. For example: mysite.mycompany.com
Port	Enter the port that the load balancer is listening on. For example, 443 for SSL Termination/Site Enabled SSL or 80 for HTTP.
SSL Protocol	Select this if SSL Termination/Site Enabled SSL is being used. This will ensure that when Portal needs to generate URLs that it generates them in these formats: https://mysite.mycompany.com:443/ (If selected) or: http://mysite.mycompany.com:80/ (If not selected)

7. Enter the following information for Web Cache:

Parameter	Value
Host	Enter the DNS name of the load balancer. For example: mysite.mycompany.com
Invalidation Port	Enter the Portal Invalidation port as configured at the load balancer, for example: 9401
Invalidation User Name	Enter the user name to be used for Portal invalidations, for example: invalidator
Invalidation Password	Password for the above account. If you do not know the value of this password, it can be changed in Enterprise Manager as described above.

8. Click **Apply** to start the rewire.
9. After the rewire completes, click the **Portal** menu option again.
10. Ensure that the Portal URL now shows:

    ```
    https://mysite.mycompany.com:443/portal/pls/portal
    ```

    or:

    ```
    http://mysite.mycompany.com:80/portal/pls/portal
    ```

12.6.4.5.2 Configure Parallel Page Engine Loop-Back with Load Balancer The purpose of the Parallel Page Engine (PPE Servlet) is to construct pages that have been requested by users. It does this by receiving the page request from a user, making its own new requests to fetch all the pieces of the page "in parallel", assembling these pieces into a single page file and then sending the page content back to the end user (or back to the client browser).

These internal requests should be kept inside of the organization, and be served using the HTTP protocol.

The following steps are required only if you are using a load balancer.

1. Log into the Oracle Fusion Middleware Enterprise Manager using this URL:

    ```
    http://apphost1.us.oracle.com:7001/em
    ```

2. Select **Fusion Middleware > Classic -> Portal** from the object browser on the left.
3. Right-click **Portal**, and select **Settings > Page Engine**.
4. In the Advanced Properties section, add the following information:

Parameter	Value
UsePort	Select the internal loopback port number, for example: 7777. This port will have been configured at the load balancer for internal requests.
Use Scheme	Change to the value of HTTP
HTTPS Ports	443. Set this to the SSL listening port on the load balancer.

5. Click **Apply** to save the settings.
6. Restart the WebLogic Managed Server from the Oracle WebLogic Administration Console as follows:

a. Select **Servers** from the Home page or **Environment > Servers** from the **Domain structure** menu.

b. Select the **Control** tab.

c. Select the box next to the **WLS_PORTAL** Managed Server.

d. Choose **Shutdown > Force Shutdown Now**.

e. Click **Yes** to shut down the Managed Server.

f. Once the Managed Server is shut down, select the box next to the Managed Server.

g. Click **Start**.

h. Click **Yes** to start the Managed Server.

12.6.4.5.3 Database Wallets and Portal If you are using SSL, Oracle Portal requires that the database in which the portal schema resides has a wallet. In this wallet is stored the certificate of the load balancer or the site. Once the wallet has been created, Oracle Portal must be informed of its location.

Note: This step is required only if you are using SSL Termination or Site SSL.

Create a Database Wallet

Before starting this process, copy the certificate to the database servers.

Each browser does this in a slightly different way. For Firefox browsers:

1. Use a browser to access the URL `https://mysite.mycompany.com`.
2. Go to **Firefox > Preferences > Advanced > Encryption > view certificates**.
3. Highlight the certificate for `mysite.mycompany.com` select export and give the file a name.
4. Copy this file to the database server.
5. Save the certificate if requested.
6. Now that you have obtained a copy of the certificate, create a wallet on each of the database servers, and import this certificate using the Oracle Wallet Manager from the database server. This must be done for all of the Oracle RAC nodes by following these steps:

   a. Type **owm** to invoke the Oracle Wallet Manager.

   b. Select **Wallet > New**.

   c. Select **No** so that you do not create the wallet in the default location.

   d. Enter a password for the wallet (remember this password - it is needed later)

   e. Set the Wallet Type to **Standard**.

   f. Select **No** to the question Do you want to create a certificate at this time?.

   g. In Oracle Wallet Manager, select **Operations > Import Trusted certificate**.

   h. Select **Select a file that contains the certificate**, and click **OK**.

   i. Select the certificate file selected above and click **Import**.

   j. Select **Wallet** and **Save As**.

k. Select a location for the wallet. For example:

```
ORACLE_BASE/admin/DB_NAME/wallet
```

l. Repeat this procedure for other Oracle RAC database nodes.

You must use the same directory paths on all database nodes.

Identify the Wallet to Portal

Now that the certificate is stored inside the database wallet, store the location of the wallet within the portal repository by following these steps:

1. Run the SQL*Plus script `secwc.sql`, which is located in the following directory

```
ORACLE_HOME/portal/admin/plsql/wwc
```

It may be necessary to create a database entry in the file `tnsnames.ora` located in `ORACLE_HOME/network/admin`:

```
SQL> @secwc 'file:$ORACLE_BASE/admin/DB_NAME/wallet' 'walletpassword'
```

In the command above:

- Use the absolute path to the wallet. Do not use environment variables.
- *walletpassword* is the password for the wallet.
- Use the path to the wallet directory, not the wallet file itself. The keyword file is required.

12.6.4.5.4 Restart All Components After making the changes in previous sections, the Web tier components must be restarted. This section describes the procedure for restarting the components.

Restart Web Components

Restart the Oracle HTTP Server using these commands:

```
opmnctl stopall
opmnctl startall
```

Restart WLS_Portal

Restart the Managed Server WLS_PORTAL by logging into the WebLogic Administration Console and following these steps:

1. Select **Servers** from the Home page or **Environment > Servers** from the **Domain structure** menu.
2. Select the **Control** tab.
3. Select the box next to the `WLS_PORTAL` Managed Server.
4. Choose **Shutdown > Force Shutdown Now**.
5. Click **Yes** to shut down the Managed Server.
6. Once the Managed Server is shut down, select the box next to the `WLS_PORTAL` Managed Server.
7. Click **Start**.
8. Click **Yes** to start the Managed Server.

12.6.4.5.5 Post-installation Step for Portal Installation with Oracle RAC Oracle recommends setting initial-capacity for individual data sources created for Portal and Portlet components to **0**. Please use Administration Console to set initial-capacity to 0 for these data sources.

12.6.4.5.6 Validate Configuration To validate the configuration, perform the following tests for SSL:

Test	URL	Result
Test Portal using the load balancer	https://mysite.mycompany.com/portal/pls/portal	Portal Home page is displayed
Test Portal Login using the load balancer	https://mysite.mycompany.com/portal/pls/portal	Should be able to log in using account `orcladmin`

To validate the configuration, perform the following tests for non-SSL:

Test	URL	Result
Test Portal using the load balancer	http://mysite.mycompany.com/portal/pls/portal	Portal Home page is displayed

Troubleshooting Single Sign-On Errors

If a Single Sign-On error message appears on the Portal front screen at this stage in the configuration, the database wallet may not have been created properly, or Oracle Portal may not have been notified of its correct location.

If this error appears, repeat the steps to create the database wallet and the subsequent identification to portal.

12.6.4.6 Configure Oracle Forms for High Availability

Use the following steps to configure Oracle Forms only.

12.6.4.6.1 Create TNSNAMES Entries for Customer Databases If the application will access one or more databases, an entry for each database being accessed must be placed into the file `tnsnames.ora`.

to place an entry for each database into the file, follow these steps:

1. Edit the file `ORACLE_INSTANCE/config/tnsnames.ora` and add an entry similar to this:

```
mydb.mycompany.com =
  (DESCRIPTION =
    (ADDRESS = (PROTOCOL = TCP)(HOST = mydbnode1-vip)(PORT = 1521))
    (ADDRESS = (PROTOCOL = TCP)(HOST = mydbnode2-vip)(PORT = 1521))
    (LOAD_BALANCE = yes)
    (CONNECT_DATA =
      (SERVER = DEDICATED)
      (SERVICE_NAME = mydb.mycompany.com)
    )
  )
```

Note: This is an Oracle RAC database connect string.

2. Save the file and test that the database connection is configured correctly using the command:

```
tnsping mydb.mycompany.com
```

Note: Set the environment variable TNS_ADMIN before issuing the **tnsping** command above.

12.6.4.6.2 Restart WLS_FORMS Restart the Managed Server WLS_FORMS by logging into the WebLogic Administration Console and following these steps:

1. Select **Servers** from the Home page or **Environment > Servers** from the **Domain structure** menu.
2. Select the **Control** tab.
3. Select the box next to the `WLS_FORMS` Managed Server.
4. Choose **Shutdown > Force Shutdown Now**.
5. Click **Yes** to shut down the Managed Server.
6. Once the Managed Server is shut down, select the box next to the `WLS_FORMS` Managed Server.
7. Click **Start**.
8. Click **Yes** to start the Managed Server.

12.6.4.6.3 Validate Configuration To validate the configuration, the following tests should be performed:

Test	URL	Result
Test load balancer	http://mysite.mycompany.com/	Home page is displayed
Test load balancer using SSL	https://mysite.mycompany.com/	Home page is displayed
Forms	https://mysite.mycompany.com/forms/frmservlet	Forms Servlet Home page is displayed

12.6.4.7 Configure Oracle Reports for High Availability

The following steps are required to configure Oracle Reports only.

12.6.4.7.1 Create Reports Queue in Database To maintain a consistent reports queue across multiple Reports server instances, and to be resilient to the failure of a reports server, create the reports queue in a high availability Real Application Clusters database.

To create the Reports queue, run the SQL*Plus script `rw_server.sql` against the database.

The script is located in this directory:

```
ORACLE_HOME/reports/admin/sql
```

Follow these steps:

1. Create a user to hold the report queue in the database using the following commands:

   ```
   sql> create user report_queue identified by MYSpassword;
   sql> grant connect, resource,create view to report_queue;
   ```

2. Connect to the Reports user and execute the following script:

   ```
   sqlplus report_queue/MYSpasswd
   sql> @ORACLE_HOME/reports/admin/sql/rw_server.sql
   ```

12.6.4.7.2 Create a TNSNAMES Entry for Reports Queue Oracle Reports uses entries in the `tnsnames.ora` file to determine database connection information. Place an entry in this file for the reports queue database by following these steps:

1. Edit the file `ORACLE_INSTANCE/config/tnsnames.ora` file and add an entry similar to this:

   ```
   myrepq.mycompany.com =
     (DESCRIPTION =
       (ADDRESS = (PROTOCOL = TCP)(HOST = mydbnode1-vip)(PORT = 1521))
       (ADDRESS = (PROTOCOL = TCP)(HOST = mydbnode2-vip)(PORT = 1521))
       (LOAD_BALANCE = yes)
       (CONNECT_DATA =
         (SERVER = DEDICATED)
         (SERVICE_NAME = myrepq.mycompany.com)
       )
     )
   ```

   Note: This is an Oracle RAC database connect string.

2. Save the file and test that the database connection is configured correctly using the following command:

   ```
   tnsping myrepq.mycompany.com
   ```

   Note: Set the environment variable TNS_ADMIN before using the **tnsping** command.

12.6.4.7.3 Create a Security Key for the Reports Queue Oracle Reports security is performed using an indirect model. Before the reports server can be configured to use the database reports queue, make an entry in WebLogic to hold the reports queue password by following these steps:

1. Log into Oracle Fusion Middleware Enterprise Manager using the following URL, entering the WebLogic administrator user and password when prompted:

   ```
   http://apphost1.mycompany.com:7001/em
   ```

2. In the navigation tree on the left hand side, expand **WebLogicDomain**, and click on the name of the domain, for example: `ReportsDomain`.

   The ReportsDomain overview page appears.

3. From the drop-down menu at the top of the page select **Security > Credentials**.

4. Click **Create Key**.
5. Provide the following information:

Parameter	Value
Select Map	reports
Key	queuePassword
Type	Password
User Name	report_queue
Password	Password for the reports queue account
Description	Description for the reports queue account

6. Click **OK** when finished.

12.6.4.7.4 Configure the Database Job Repository for In-Process Reports Servers After the reports queue has been placed into the database, the reports server needs to be told how to access it.

To do this, edit the `rwserver.conf` file.

The file is located in the following directory:

```
DOM_HOME/servers/WLS_REPORTS/stage/reports/reports/configuration
```

There is a fixed order for entries in the server configuration file. Add the following element immediately after the `<jobStatusReppository>` element in the `rwserver.conf` file:

```
<jobRepository>
 <property name="dbuser" value="dbuser"/>
 <property name="dbpassword" value="csf:reports:dbpasswdKey"/>
 <property name="dbconn" value="dbconn"/>
</jobRepository>
```

where:

- *dbuser* is the name of the schema where the reports queue resides
- *dbconn* references a database entry in the tnsnames.ora file in the *ORACLE_INSTANCE*/config directory

For example:

```
<jobRepository>
 <property name="dbuser" value="report_queue"/>
 <property name="dbpassword" value="csf:reports:queuePassword"/>
 <property name="dbconn" value="myrepq.mycompany.com"/>
</jobRepository>
```

12.6.4.7.5 Configure the Reports Server to Access Shared Output Directory Add the CacheDir or JOCCacheDir property to the `<cache>` element the file `rwserver.conf` file, located in the following directory:

```
DOM_HOME/servers/WLS_REPORTS/stage/reports/reports/configuration
```

For example:

```
<property name="JOCCacheDir" value="folder_name"/>
```

```
<property name="CacheDir" value="folder_name"/>
```

On UNIX:

```
<cache class="oracle.reports.cache.RWCache">
 <property name="cacheSize" value="50"/>
 <!--property name="cacheDir" value="your cache directory"/-->
 <!--property name="maxCacheFileNumber" value="max number of cache files"/-->
 <property name="JOCCacheDir" value="/share/reports"/>
 <property name="CacheDir" value="/share/reports"/>
</cache>
```

12.6.4.7.6 Restart WLS_REPORTS To restart the Managed Server WLS_REPORTS, log into the WebLogic Administration Console, and follow these steps:

1. Select **Servers** from the Home page or **Environment > Servers** from the **Domain structure** menu.
2. Select the **Control** tab.
3. Select the box next to the `WLS_REPORTS` Managed Server.
4. Choose **Shutdown > Force Shutdown Now**.
5. Click **Yes** to shut down the Managed Server.
6. Once the Managed Server is shut down, select the box next to the `WLS_REPORTS` Managed Server.
7. Click **Start**.
8. Click **Yes** to start the Managed Server.

12.6.4.7.7 Validate Configuration Validate the initial Reports configuration by performing these tests:

Test	URL	Result
Reports Queue - SSL	https://mysite.mycompany.com/reports/rwservlet/showjobs	Single Sign On screen and then Reports Queue is displayed
Reports Queue - Non SSL	http://mysite.mycompany.com/reports/rwservlet/showjobs	Single Sign On screen and then Reports Queue is displayed

For an alternate test, download a sample report from Oracle Technology Network, and run it. The URL for Oracle Technology Network is:

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

12.6.4.8 Configure Oracle Discoverer for High Availability

Follow these steps to configure high availability for Oracle Discoverer only.

12.6.4.8.1 Create TNSNAMES Entries for Customer Databases If the application accesses one or more databases, place an entry for each database being accessed into the `tnsnames.ora` file, by following these steps:

1. Edit the file `ORACLE_INSTANCE/config/tnsnames.ora`, and add an entry similar to this:

   ```
   mydb.mycompany.com =
   ```

```
(DESCRIPTION =
  (ADDRESS = (PROTOCOL = TCP)(HOST = mydbnode1-vip)(PORT = 1521))
  (ADDRESS = (PROTOCOL = TCP)(HOST = mydbnode2-vip)(PORT = 1521))
  (LOAD_BALANCE = yes)
  (CONNECT_DATA =
    (SERVER = DEDICATED)
    (SERVICE_NAME = mydb.mycompany.com)
  )
)
```

Note: This is an Oracle RAC database connect string.

2. Save the file and test that the database connection is configured correctly using the following command:

```
tnsping mydb.mycompany.com
```

Note: Set the environment variable TNS_ADMIN before issuing the **tnsping** command.

12.6.4.8.2 Update configuration.xml The configuration.xml stores information about the Discoverer configuration. The file is located in the following directory:

```
DOM_HOME/servers/WLS_DISCO/stage/discoverer/11.1.1.1.0/discoverer/configuration
```

Update the line beginning with `applicationURL=` and change the URL to:

```
http://mysite.mycompany.com/discoverer
```

For example:

```
applicationURL="http://mysite.mycompany.com/discoverer">
```

Note: If you choose SSL the application URL is as follows:

https://mysite.mycompany.com/discoverer

For non-SSL the application URL is as follows:

http://mysite.mycompany.com/discoverer

12.6.4.8.3 Discoverer Viewer and Web Cache By default, Discoverer Viewer is not configured to make full use of Oracle Web Cache. When enabled, significant performance gains can be attained. However, it is not always appropriate to enable this functionality.

For details on when and how to enable Discoverer Viewer with Oracle Web Cache, see the "Using Discoverer Viewer with Oracle Web Cache" section of *Oracle Business Intelligence Discoverer Configuration Guide*.

12.6.4.8.4 Enable Single Sign On By default, Discoverer is not protected by single sign-on. To secure Discoverer Plus and Viewer, use the following steps:

1. Edit the file `mod_osso.conf` file, located in the following directory:

```
ORACLE_INSTANCE/config/OHS/ohs1/moduleconf
```

2. Add the following lines to the file before the line that begins with `</IfModule>`:

```
<Location /discoverer/plus>
 require valid-user
 AuthType Osso
</Location>

<Location /discoverer/viewer>
 require valid-user
 AuthType Osso
</Location>

<Location /discoverer/app>
 require valid-user
 AuthType Osso
</Location>
```

12.6.4.8.5 Restart All Components After making the changes in previous sections, restart the Web tier components using the procedures described in this section

Restart Web Components

Restart the Oracle HTTP Server using the following commands:

```
opmnctl stopall
opmnctl startall
```

Restart WLS_DISCO

Restart the Managed Server WLS_DISCO by logging into the WebLogic Administration Console and following these steps:

1. Select **Servers** from the Home page or **Environment > Servers** from the **Domain structure** menu.
2. Select the **Control** tab.
3. Select the box next to the `WLS_DISCO` Managed Server.
4. Choose **Shutdown > Force Shutdown Now**.
5. Click **Yes** to shut down the Managed Server.
6. Once the Managed Server is shut down, select the box next to the `WLS_DISCO` Managed Server.
7. Click **Start**.
8. Click **Yes** to start the Managed Server.

12.6.4.8.6 Validate Configuration Validate the initial Discoverer configuration by performing these tests:

Test	URL	Result
Test load balancer	http://mysite.mycompany.com/	Home page is displayed

Test	URL	Result
Test load balancer using SSL	https://mysite.mycompany.com/	Home page is displayed
Discoverer (SSL)	https://mysite.mycompany.com/discoverer/viewer	Single Sign-On and then Discoverer Viewer is displayed
Discoverer (Non-SSL)	http://mysite.mycompany.com/discoverer/viewer	Single Sign-On and then Discoverer Viewer is displayed

12.6.5 Install and Configure Application Tier on APPHOST2

The following sections describe how to install and configure the Application Tier on APPHOST2.

- Section 12.6.5.1, "Install Oracle WebLogic Server"
- Section 12.6.5.2, "Install Oracle Portal, Forms, Reports, and Discoverer Software"
- Section 12.6.5.3, "Generic Configuration"
- Section 12.6.5.4, "Configure Oracle Portal for High Availability"
- Section 12.6.5.5, "Configure Oracle Forms for High Availability"
- Section 12.6.5.6, "Configure Oracle Reports for High Availability"
- Section 12.6.5.7, "Configure Oracle Discoverer for High Availability"

12.6.5.1 Install Oracle WebLogic Server

To install Oracle WebLogic Server binaries, see the *Oracle Fusion Middleware Installation Guide for Oracle WebLogic Server*.

12.6.5.2 Install Oracle Portal, Forms, Reports, and Discoverer Software

Install the Oracle binaries into the MW_HOME created in the previous procedure.

For UNIX:

```
runInstaller
```

For Windows:

```
setup.exe
```

Note: Before starting the install ensure that the following environment variables (UNIX) are not set:

- LD_ASSUME_KERNEL
- ORACLE_BASE
- LD_LIBRARY_PATH

Continue with these steps:

1. In the Welcome screen, click **Next**.

2. In the Installation Type screen, choose **Install Software and Configure**, then click **Next**.
3. In the Prerequisite Checks screen, once all the prerequisites have been validated, click **Next**.
4. In the Create Domain screen, select **Expand Cluster** and enter these values:
   - **Host Name**: Name of the host running the WebLogic Administration Server, for example: `APPHOST1.mycompany.com`
   - **Port**: The Administration Server port. For example: `7001`
   - **User Name**: Administration Server administrator account name.
   - **Password**: Password for the Administration Server administrator account.

   Click **Next**.
5. In the Specify Oracle Configuration Manager Details screen, if required, enter a username and password to receive Oracle Security updates for Oracle Support.
6. In the Specify Installation Location screen, enter these values:
   - **Middleware Home**: Enter the value for the FMW_HOME. For example:

     ```
     /u01/app/oracle/product/FMW
     ```
   - **Oracle Home**: Enter the installation directory for Reports. Note that this will be placed under the FMW_HOME directory. For example:

     ```
     FMW_HOME
     ```
   - **WebLogic Server Directory**: Enter the installation directory for Oracle WebLogic Server. This should be `FMW_HOME/wlserver_10.3`, for example:

     ```
     /u01/app/oracle/product/FMW/wlserver_10.3
     ```
   - **Oracle Instance Location**: Enter the directory where the Oracle configuration files will be placed. This should be outside of the Oracle home directory. This directory will be known as the ORACLE_INSTANCE. For example:

     ```
     /u01/app/oracle/admin/fmw2
     ```
   - **Oracle Instance Name**: `fmw2`

   Click **Next**.
7. In the Configure Components screen, choose which products to install and configure. If placing Oracle Web Cache or the Oracle HTTP Server onto this node, ensure that they are selected.

   Note: This should be the same list as selected on APPHOST1.

   Click **Next**.
8. Select the same file used for APPHOST1.
9. In the Specify Application Identity Store screen, specify the following values:
   - **Hostname**: Name of the Oracle Internet Directory server, for example: `login.myoid.com`
   - **Port**: Oracle Internet Directory port, for example: `389`

- **User Name**: `cn=orcladmin`
- **Password**: Password for Oracle Internet Directory's `orcladmin` account

10. In the Summary screen, click **Install** to begin the creation process.

12.6.5.3 Generic Configuration

The following steps are required regardless of the component you are configuring.

12.6.5.3.1 Copy Configuration Information from APPHOST1 After the Expand Cluster operation has created a new WebLogic Managed Server and associated machine, it is necessary to copy some configuration information from APPHOST1 to APPHOST2.

Copy the following files located on APPHOST1 to the APPHOST2 locations shown in the following table:

File	APPHOST1 Location	APPHOST2 Location
mod_oradav.conf	*ORACLE_INSTANCE*/config/OHS/ohs1/moduleconf	*ORACLE_INSTANCE*/config/OHS/ohs1/moduleconf
mod_osso.conf	*ORACLE_INSTANCE*/config/OHS/ohs1/moduleconf	*ORACLE_INSTANCE*/config/OHS/ohs1/moduleconf
plsql.conf	*ORACLE_INSTANCE*/config/OHS/ohs1/moduleconf	*ORACLE_INSTANCE*/config/OHS/ohs1/moduleconf
portal.conf	*ORACLE_INSTANCE*/config/OHS/ohs1/moduleconf	*ORACLE_INSTANCE*/config/OHS/ohs1/moduleconf
forms_ohs.conf	*ORACLE_INSTANCE*/config/OHS/ohs1/moduleconf	*ORACLE_INSTANCE*/config/OHS/ohs1/moduleconf
reports_ohs.conf	*ORACLE_INSTANCE*/config/OHS/ohs1/moduleconf	*ORACLE_INSTANCE*/config/OHS/ohs1/moduleconf
module_disco.conf	*ORACLE_INSTANCE*/config/OHS/ohs1/moduleconf	*ORACLE_INSTANCE*/config/OHS/ohs1/moduleconf
virtual_hosts.conf	*ORACLE_INSTANCE*/config/OHS/ohs1/moduleconf	*ORACLE_INSTANCE*/config/OHS/ohs1/moduleconf
osso.conf	*ORACLE_INSTANCE*/config/OHS/ohs1/	*ORACLE_INSTANCE*/config/OHS/ohs1/
sqlnet.ora	*ORACLE_INSTANCE*/config	*ORACLE_INSTANCE*/config
tnsnames.ora	*ORACLE_INSTANCE*/config	*ORACLE_INSTANCE*/config

12.6.5.3.2 Configure Virtual Hosts For the site to function properly with the load balancer, you must add two virtual hosts. You can configure the virtual hosts using Oracle Enterprise Manager Fusion Middleware Control, edit the file *ORACLE_INSTANCE*/config/OHS/ohs1/httpd.conf, or edit the `virtual_hosts.conf` file located in the in *ORACLE_INSTANCE*/config/OHS/ohs1/moduleconf directory (copied from APPHOST1).

In a text editor, update the file to change APPHOST1 to APPHOST2:

```
NameVirtualHost *:7778
<VirtualHost *:7778>
     ServerName https://mysite.mycompany.com:443 ** If using SSL
Termination/site at LB
     ServerName http://mysite.mycompany.com:80 ** If not using SSL
    RewriteEngine On
    RewriteOptions inherit
    UseCanonicalName On
</VirtualHost>

<VirtualHost *:7778>
    ServerName apphost2.mycompany.com:7777
    RewriteEngine On
    RewriteOptions inherit
    UseCanonicalName On
</VirtualHost>
```

12.6.5.3.3 Update Oracle HTTP Server Configuration to be Cluster Aware

When the installation was first created, it was configured so that all Web Logic requests were directed to the initially created Managed Servers residing on APPHOST1. Now that APPHOST2 exists, both the Oracle HTTP Servers on APPHOST1 and APPHOST2 must be made aware of all of the Oracle WebLogic Managed Servers.

To make the Oracle HTTP Server aware of APPHOST2, edit files located in the following directory:

```
ORACLE_INSTANCE/config/OHS/ohs1/moduleconf
```

There are different files to edit for Portal, Forms, Reports, and Discoverer, as shown in the following table:

Product	File
Portal	portal.conf
Forms	forms_ohs.conf
Reports	reports_ohs.conf
Discoverer	module_disco.conf

Edit the files as follows:

1. Remove the lines that begin with `WebLogicHost` and `WebLogicPort` and add the following lines:

   ```
   WebLogicCluster apphost1:9001,apphost2:9001
   ```

2. Change any lines that begin with `WebLogicCluster` to look similar to this:

   ```
   WebLogicCluster apphost1:9001,apphost2:9001
   ```

   For example, change the following:

   ```
   <Location /portal>
       SetHandler weblogic-handler
       WebLogicHost apphost1.mycompany.com
       WebLogicPort PORT
   </Location>
   ```

   to this:

```
<Location /portal>
    SetHandler weblogic-handler
    WebLogicCluster apphost1.mycompany.com:PORT,apphost2.mycompany.com:PORT
</Location>
```

Change the following:

```
<Location /Forms>
    SetHandler weblogic-handler
    WebLogicCluster apphost1.mycompany.com:PORT
</Location>
```

to this:

```
<Location /Forms>
    SetHandler weblogic-handler
    WebLogicCluster apphost1.mycompany.com:PORT,apphost2.mycompany.com:PORT
</Location>
```

Repeat these steps on APPHOST2.

12.6.5.3.4 Change the Web Cache Passwords Web Cache passwords are randomly generated, however they are required in later stages. It is therefore recommended that the Web Cache passwords be changed from the default value to a new known value.

To change the default password, follow these steps:

1. In the Navigator window, expand the **Web Tier** tree.
2. Click on the component `wc1`.
3. From the drop down list at the top of the page, select **Administration - Passwords**.
4. Enter new invalidation and administartion passwords, confirm them, and click **Apply**.

Note: The passwords used in this procedure must be the same as those used on APPHOST1. This is required for the web caches to be clustered together in later procedures.

12.6.5.3.5 Configure Web Cache Web Cache is optional for Oracle Forms, Reports and Discoverer, but mandatory for Oracle Portal. If Web Cache was not configured for APPHOST1, ignore the following Web Cache configuration steps:

Log into the Enterprise Manager Console

Log into the Enterprise Manager Console using the following URL:

```
http://apphost1.mycompany.com:7001/em
```

The default User Name and Password are the same as the WebLogic domain user name and password entered during the installation.

Create the Origin Server

To create the origin server:

1. In the Navigator Window, expand the Web Tier tree.
2. Click on the component `wc1` (make sure it is the one associated with APPHOST1).

3. From the drop down list at the top of the page select **Administration - Origin Servers**.
4. Click **Create**.
5. Enter the following information to add the origin server:

Field	Value
Host	APPHOST2.mycompany.com
Port	7778
Capacity	100
Protocol	HTTP
Failover Threshold	5
Ping URL	/
Ping Interval	10

6. Click **OK** to save the changes.
7. Click **Apply** to save the changes.

Add Origin Server to existing Site to Server Mapping

To add the origin server site to server mapping:

1. In the Navigator Window, expand the Web Tier tree.
2. Click on the component `wc1` (make sure it is the one associated with APPHOST1).
3. From the drop down list at the top of the page select **Administration - Sites**.
4. In the Site to Server Mapping section, click on the Host:Port, for example:

   ```
   mysite.mycompany.com:443/80
   ```

5. Click **Edit**.
6. Select the origin server `APPHOST2.mycompany.com:7778` and move it to the selected Origin servers list.
7. Click **OK** to save the changes.
8. Click **Apply** to save the changes.

Cluster Web Cache on APPHOST1 and APPHOST2

Follow these steps to cluster Oracle Web Cache on APPHOST1 and APPHOST2:

Note: For Oracle Web Cache clustering to function properly, the administration password of each of the Web Caches clustered together must be the same.

1. In the Navigator Window, expand the Web Tier tree.
2. Click on the component `wc1` (make sure it is the one associated with APPHOST1).
3. From the drop down list at the top of the page select **Administration - Cluster**.
4. Click **Add**.

The Web Cache from APPHOST2 is automatically added.

5. Click **Apply** to apply the changes.
6. Click on the newly created Web Cache entry (be sure not to click on the URL part of the entry).
7. Click **Synchronize** to copy the configuration to the Web Cache on APPHOST2.
8. Click **Yes** when prompted to confirm that you wish you perform the operation.
9. Click **Apply** to apply the new configuration.

12.6.5.3.6 Restart Web Processes on APPHOST1 and APPHOST2

After making the previous changes, restart the Web Tier components on APPHOST1 and APPHOST2 using the following commands on each of the servers:

```
opmnctl stopall
opmnctl startall
```

12.6.5.3.7 Start Oracle Node Manager on APPHOST2

To start the newly created Managed Server, start WebLogic Node Manager on APPHOST2 using the following commands on APPHOST2:

```
export ORACLE_HOME=oracle home path
export LD_LIBRARY_PATH=ORACLE_HOME/lib
MW_HOME/wlserver_10.3/server/bin/startNodeManager.sh
```

12.6.5.4 Configure Oracle Portal for High Availability

This section describes the procedure for configuring only Oracle Portal for high availability.

12.6.5.4.1 Copy Configuration Information from APPHOST1

Even though the Expand Cluster has created a new WebLogic Managed Server and associated machine, it is still necessary to copy some configuration information from APPHOST1 to APPHOST2.

Copy the following files located on APPHOST1 to the APPHOST2 locations shown in the following table:

File	APPHOST1 Location	APPHOST2 Location
appConfig.xml	MW_HOME/user_projects/domains/PortalDomain/servers/WLS_PORTAL/stage/portal/portal/configuration	MW_HOME/user_projects/domains/PortalDomain/servers/WLS_PORTAL/stage/portal/portal/configuration
portal_cache.conf	MW_HOME/user_projects/domains/PortalDomain/servers/WLS_PORTAL/stage/portal/portal/configuration	MW_HOME/user_projects/domains/PortalDomain/servers/WLS_PORTAL/stage/portal/portal/configuration
portal_dads.conf	MW_HOME/user_projects/domains/PortalDomain/servers/WLS_PORTAL/stage/portal/portal/configuration	MW_HOME/user_projects/domains/PortalDomain/servers/WLS_PORTAL/stage/portal/portal/configuration

File	APPHOST1 Location	APPHOST2 Location
portal_plsql.conf	MW_HOME/user_projects/domains/PortalDomain/servers/WLS_PORTAL/stage/portal/portal/configuration	MW_HOME/user_projects/domains/PortalDomain/servers/WLS_PORTAL/stage/portal/portal/configuration

12.6.5.4.2 Create Portal Directories Create the following directories on APPHOST2 to allow the storage of the Oracle Portal Cache:

```
ORACLE_INSTANCE/portal/cache
ORACLE_INSTANCE/diagnostics/logs/portal
```

12.6.5.4.3 Update Instance Paths Edit the hard coded entries in the following two copied files to reflect the paths above.

- In the `portal_cache.conf` file, change `PlsqlCacheDirectory`.
- In the `portal_plsql.conf` file, change `PlsqlLogDirectory`.

These files are located in the *DOMAIN_HOME*/servers/WLS_PORTAL/stage/portal/portal/configuration directory.

12.6.5.4.4 Restart the Web Process Restart the Web Tier components on APPHOST2 using the following commands on each of the servers:

```
opmnctl stopall
opmnctl startall
```

12.6.5.4.5 Start WLS_PORTAL1 Now that the application files have been copied across. Start the managed server, WLS_PORTAL1:

1. Login to the administration server on APPHOST1 using the URL:

   ```
   http://APPHOST1.mycompany.com:7001/console
   ```

2. Provide the WebLogic Administration Console login credentials.
3. Select **Environment -> Servers** from the Domain structure menu.
4. Select the **Control** tab.
5. Select the box next to the Managed Server WLS_PORTAL1, and click **Shutdown - Force Shutdown Now**.
6. Click on **Yes** to confirm the operation.

   This resets the server's status.

12.6.5.4.6 Validate the Configuration To validate the configuration:

Test	URL	Result
Test Portal via Load Balancer	https://mysite.mycompany.com/portal/pls/portal	Portal Home Page Displayed
Test Portal Login via Load Balancer	https://mysite.mycompany.com/portal/pls/portal	Should be able to login using account orcladmin

With no SSL:

Test	URL	Result
Test Portal via Load Balancer	http://mysite.mycompany.com/portal/pls/portal	Portal Home Page Displayed
Test Portal Login via Load Balancer	http://mysite.mycompany.com/portal/pls/portal	Should be able to login using account orcladmin

12.6.5.4.7 Best Practices

By default configuration information is not automatically propagated to all server instances. If configuration files are changed on one portal server, propagate the configuration to all of the servers.

12.6.5.5 Configure Oracle Forms for High Availability

The section describes the procedure for configuring only Oracle Forms for high availability.

12.6.5.5.1 Create a TNSNAMES entries for Customer Databases

If the application is to access one or more databases, place an entry for each database being accessed into the file `tnsnames.ora`.

Edit the file `ORACLE_INSTANCE/config/tnsnames.ora`, and add an entry similar to the one below:

```
mydb.mycompany.com =
  (DESCRIPTION =
    (ADDRESS = (PROTOCOL = TCP)(HOST = mydbnode1-vip)(PORT = 1521))
    (ADDRESS = (PROTOCOL = TCP)(HOST = mydbnode2-vip)(PORT = 1521))
    (LOAD_BALANCE = yes)
    (CONNECT_DATA =
      (SERVER = DEDICATED)
      (SERVICE_NAME = mydb.mycompany.com)
    )
  )
```

This is an Oracle RAC database connect string.

12.6.5.5.2 Copy Forms Configuration Files

Even though the expand cluster has created a new WebLogic Managed Server and associated machine. It is still necessary to copy some configuration information from APPHOST1 to APPHOST2.

Copy the following files located on APPHOST1

File	Location of APPHOST1	Location of APPHOST2
ALL	*DOMAIN_HOME*/servers/WLS_FORMS/stage/Formsapp/11.1.1/formsapp/config	*DOMAIN_HOME*/servers/WLS_FORMS1/stage/Formsapp/11.1.1/formsapp/config
ALL	*ORACLE_INSTANCE*/config/FormsComponent/forms	*ORACLE_INSTANCE*/config/FormsComponent/forms
ALL	*ORACLE_INSTANCE*/config/FRComponent/frcommon	*ORACLE_INSTANCE*/config/FRComponent/frcommon

12.6.5.5.3 Update default.env

Having copied the above files, update the file `default.env`, located in the *DOMAIN_HOME*/servers/WLS_

FORMS/stage/formsapp/11.1.1/formsapp/config directory with the correct values for APPHOST2. In particular, the following entries must be changed:

- ORACLE_INSTANCE
- TNS_ADMIN
- FORMS_PATH
- WEBUTIL_CONFIG

Typically these entries have the following values:

```
ORACLE_INSTANCE=/u01/app/oracle/admin/Forms1
TNS_ADMIN=/u01/app/oracle/admin/Forms1/config
FORMS_
PATH=/u01/app/oracle/product/FMW/Forms/forms:/u01/app/oracle/admin/Forms1/FormsCom
ponent/forms
WEBUTIL_
CONFIG=/u01/app/oracle/admin/Forms1/config/FormsComponent/forms/server/webutil.cfg
```

And must be changed to:

```
ORACLE_INSTANCE=/u01/app/oracle/admin/Forms2
TNS_ADMIN=/u01/app/oracle/admin/Forms2/config
FORMS_
PATH=/u01/app/oracle/product/FMW/Forms/forms:/u01/app/oracle/admin/Forms2/FormsCom
ponent/forms
WEBUTIL_
CONFIG=/u01/app/oracle/admin/Forms2/config/FormsComponent/forms/server/webutil.cfg
```

12.6.5.5.4 Restart WLS_FORMS1 After making the changes in the previous section, restart the WebLogic managed servers WLS_FORMS and WLS_FORMS1 by logging into the Oracle WebLogic Administration Console using the following URL

```
http://apphost1.mycompany.com:7001/console
```

1. Select **Environment -> Servers** from the Domain Structure menu.
2. Select the **Control** tab.
3. Select the boxes next to the server WLS_FORMS1.
4. Select **Shutdown -> Force shutdown now**.
5. Select **Yes** when the confirmation dialogue is displayed.
6. When the server is shutdown, restart it by selecting the boxes next to the server WLS_FORMS1.
7. Click **Start**.

12.6.5.5.5 Validate the Configuration To validate the configuration, sue the following tests:

Test	URL	Result
Test Load Balancer	http://mysite.mycompany.com/	AS Home Page Displayed
Test Load Balancer using SSL	https://mysite.mycompany.com/	AS Home Page Displayed
Forms through SSL	https://mysite.mycompany.com/forms/frmservlet	Forms Servlet Home Page displayed.

Test	URL	Result
Forms	http://mysite.mycompany.com/forms/frmservlet	Forms Servlet Home Page displayed.

12.6.5.5.6 Best Practices

WebLogic Application Server does not replicate configuration information automatically between nodes. It is important to manually propagate any changes to any of the following to the other servers in the deployment:

- ORACLE_INSTANCE
- *DOMAIN_HOME*/servers/WLS_FORMS/stage/formsapp/11.1.1/formsapp/config

12.6.5.6 Configure Oracle Reports for High Availability

This section describes the procedures for configuring only Oracle Reports for high availability.

12.6.5.6.1 Create a TNSNAMES entries for Customer Databases

Oracle Reports uses entries in the `tnsnames.ora` file to determine database connection information. Place an entry into this file for the Oracle Reports queue database.

Edit the file `ORACLE_INSTANCE/config/tnsnames.ora` and add an entry similar to the following:

```
mydb.mycompany.com =
  (DESCRIPTION =
    (ADDRESS = (PROTOCOL = TCP)(HOST = mydbnode1-vip)(PORT = 1521))
    (ADDRESS = (PROTOCOL = TCP)(HOST = mydbnode2-vip)(PORT = 1521))
    (LOAD_BALANCE = yes)
    (CONNECT_DATA =
      (SERVER = DEDICATED)
      (SERVICE_NAME = mydb.mycompany.com)
    )
  )
```

Save the file and test that the database connection is configured correctly using the following command:

```
tnsping myreportq.mycompany.com
```

12.6.5.6.2 Configure the Reports Server to Access Shared output directory.

Add the CacheDir or JOCCacheDir property to the <cache> element of each of the `rwserver.conf` files, which are located in the *DOMAIN_HOME*/servers/WLS_REPORTS1/stage/reports/reports/configuration directory.

For example:

```
<property name="JOCCacheDir" value="folder_name"/>
<property name="CacheDir" value="folder_name"/>
```

For UNIX

```
<cache class="oracle.reports.cache.RWCache">
      <property name="cacheSize" value="50"/>
      <!--property name="cacheDir" value="your cache directory"/-->
      <!--property name="maxCacheFileNumber" value="max number of cache files"/-->
      <property name="JOCCacheDir" value="/share/reports"/>
      <property name="CacheDir" value="/share/reports"/>
   </cache>
```

12.6.5.6.3 Configure the database job repository for in-process Reports Servers Now that the reports queue has been placed into the database, notify the Reports server how to access it using the by editing the file rwserver.conf file, located in the *DOMAIN_HOME*/servers/WLS_REPORTS1/stage/reports/reports/configuration directory.

When editing the file, the order in which different entries appear inside the server configuration file is fixed. As a result, the following element must be added immediately after the <jobStatusRepository> element in the server configuration file:

```
 <jobRepository>
        <property name="dbuser" value="dbuser"/>
        <property name="dbpassword" value="csf:reports:dbpasswdKey"/>
        <property name="dbconn" value="dbconn"/>
      </jobRepository>
```

Where dbuser is the name of the schema where the reports queue resides.

Also, dbconn references a database entry in the `tnsnames.ora` file located in *ORACLE_INSTANCE*/config directory. `dbpasswdKey` is the name of the entry created.

For example:

```
<jobRepository>
        <property name="dbuser" value="reports_queue"/>
        <property name="dbpassword" value="csf:reports:queuePassword"/>
        <property name="dbconn" value="myrepq.mycompany.com"/>
</jobRepository>
```

The value of clusternodes is the value that appears in the <server> tag in the file `rwservlet.properties` file on APPHOST1.

The clusternodes parameter should list all of the reports servers in the cluster (comma separated) **EXCEPT** the local report server.

12.6.5.6.4 Restart WLS_REPORTS and WLS_REPORTS1 Restart the Web Logic managed servers WLS_REPORTS and WLS_REPORTS1 by logging into the WebLogic Administration Console using the following URL:

```
http://apphost1.mycompany.com:7001/console
```

1. Select **Environment -> Servers** from the Domain Structure menu.
2. Select the **Control** tab.
3. Select the boxes next to the server WLS_REPORTS and WLS_REPORTS1.
4. Select **Shutdown -> Force shutdown now**.
5. Select **Yes** when the confirmation dialogue is displayed.
6. When the server is shutdown, restart it by selecting the boxes next to the server WLS_REPORTS and WLS_REPORTS1.
7. Click **Start**.

12.6.5.6.5 Validate the Configuration To validate the configuration run the following tests:

Test	URL	Result
Test Load Balancer	http://mysite.mycompany.com/	AS Home Page Displayed
Test Reports Queue	https://mysite.mycompany.com/reports/rwservlet/showjobs	Single Sign On and Reports Queue pages are displayed

12.6.5.6.6 Managing Connection Availability for Oracle Reports Services You must tune the time out settings to manage idle connections for the Load Balancing Router, firewall, Oracle HTTP Server, and WebLogic Server according to their Oracle Reports Services application requirements. For example, if the Load Balancing Router and firewall connection timeout is set to ten minutes, and the execution time for a report exceeds ten minutes, then the browser connection is timed out by the Load Balancing Router, and the Oracle Reports Services output does not reach the browser. The time out value for the connection between Oracle Reports Services clients and the Oracle Reports server is governed by the idleTimeout attribute of the connection element in the Oracle Reports Services Server configuration file `rwserver.conf`.

12.6.5.7 Configure Oracle Discoverer for High Availability

This section describes the procedures for configuring high availability for only Oracle Discoverer.

12.6.5.7.1 Create a TNSNAMES entries for Customer Databases If the application is to access one or more databases, place an entry for each database being accessed into the `tnsnames.ora` file.

Edit the file *ORACLE_INSTANCE*/config/tnsnames.ora and add an entry as follows:

```
mydb.mycompany.com =
  (DESCRIPTION =
    (ADDRESS = (PROTOCOL = TCP)(HOST = mydbnode1-vip)(PORT = 1521))
    (ADDRESS = (PROTOCOL = TCP)(HOST = mydbnode2-vip)(PORT = 1521))
    (LOAD_BALANCE = yes)
    (CONNECT_DATA =
      (SERVER = DEDICATED)
      (SERVICE_NAME = mydb.mycompany.com)
    )
  )
```

This is an Oracle RAC database connect string.

Save the file and test that the database connection is configured correctly using the following command:

```
tnsping myreportq.mycompany.com
```

12.6.5.7.2 Copy Discoverer Configuration Files. The expand cluster procedure has created a new WebLogic Managed Server and associated machine. However, you must still copy some configuration information from APPHOST1 to APPHOST2.

Prior to performing this step make a copy of the `configuration.xml` file located in the `DOM_HOME/servers/WLS_DISCO/stage/Discoverer/11.1.1.0/Discoverer/configuration` directory.

Copy this file from APPHOST1 to the same directory in APPHOST2.

12.6.5.7.3 Update configuration.xml Update the `configuration.xml` file you just copied, replacing the following values with the values in the `configuration.xml.orig` file.

```
<oracleInstance>
<discovererComponentName>
```

12.6.5.7.4 Changing the Preference Store In an enterprise deployment, there should only be one Discoverer preference store active at one time. To change the preference store, change Oracle Discoverer on APPHOST2 to point to the preference store on APPHOST1.

Identify Preference Store Host Name and Port

The nominated preference server in this example is configured to run on APPHOST1. When installed onto the server, the Preference server is assigned a port. Locate this value before proceeding to the following next steps:

1. On APPHOST1, open the opmn.xml file is located in the *ORACLE_INSTANCE*/config/OPMN/opmn directory.
2. Search the file for the entry PREFERENCE_PORT the value= shows the port assigned to the preference server.

Specifying a Discoverer Preferences Server on other Machines

Having identified the host name and port number of the machine that is going to run the Preferences component (that is, the Discoverer Preferences server machine), you must now ensure that other machines use the Preferences component on the Discoverer Preferences server machine.

To modify the `opmn.xml` file of other machines to use the Discoverer Preferences server machine, perform the following steps on every other machine in the installation.

1. On each machine except the Preferences server machine, open the `opmn.xml` file in a text editor (or XML editor).
2. Locate the PREFERENCE_HOST variable, and change its value to the host name of the Discoverer Preferences server machine, as follows:

   ```
   <variable id="PREFERENCE_HOST" value="hostname">
   ```

3. Locate the PREFERENCE_PORT variable, and change its value to the port number of the Discoverer Preferences server machine, as follows:

   ```
   <variable id="PREFERENCE_PORT" value="port">
   ```

4. Locate the PreferenceServer process type, and change its status to disabled, as follows:

   ```
   <process-type id="PreferenceServer" module-id="Disco_PreferenceServer"
   working-dir="$DC_LOG_DIR" status="disabled">
   ```

5. Save the `opmn.xml` file.

12.6.5.7.5 Restart WLS_DISCO and WLS_DISCO1 Restart the Web Logic managed servers WLS_DISCO and WLS_DISCO1 by logging into the WebLogic Administration Console using the following URL:

```
http://apphost1.mycompany.com:7001/console
```

1. Select **Environment -> Servers** from the Domain Structure menu.
2. Select the **Control** tab.
3. Select the boxes next to the server WLS_DISCO and WLS_DISCO1.
4. Select **Shutdown -> Force shutdown now**.
5. Select **Yes** when the confirmation dialogue is displayed.
6. When the server is shutdown, restart it by selecting the boxes next to the server WLS_DISCO and WLS_DISCO1.
7. Click **Start**.

12.6.5.7.6 Validate the Configuration Validate the configuration using the following tests:

Test	URL	Result
Test Load Balancer	http://mysite.mycompany.com/	AS Home Page Displayed
Test Load Balancer using ssl	https://mysite.mycompany.com/	Single Sign-On and Reports Queue pages are displayed
Discoverer	http://mysite.mycompany.com /discoverer/viewer	Discoverer Servlet Home Page displayed.
Discoverer (SSL)	https://mysite.mycompany.com /discoverer/viewer	Discoverer Servlet Home Page displayed.

12.6.5.7.7 Failover of the Preference Server If the server hosting the preference server fails, the preference server must be started on one of the surviving nodes using the following procedure:

Identify Preference Store Host Name and Port

The nominated preference server in this example is configured to run on APPHOST1. When installed onto the server, the Preference server is assigned a port. Locate this value using the following procedure:

1. On APPHOST1, open the opmn.xml file is located in the *ORACLE_INSTANCE*/config/OPMN/opmn directory.
2. Search the file for the entry PREFERENCE_PORT the value= shows the port assigned to the preference server.

Enable the Preference Store on APPHOST2

Now that APPHOST2 is using the preference store on APPHOST1, disable the preference store on APPHOST2 by editing the `opmn.xml` file located in the *ORACLE_INSTANCE*/config/OPMN/opmn directory

Locate the following line:

```
<process-type id="PreferenceServer" module-id="Disco_PreferenceServer"
working-dir="$DC_LOG_DIR" status="disabled">
```

Change `status=disabled` to **status=enabled**.

For example:

```
<process-type id="PreferenceServer" module-id="Disco_PreferenceServer"
working-dir="$DC_LOG_DIR" status="enabled">
```

Start the Preference Server on APPHOST2

Start the preference server on APPHOST2 using the following command:

```
opmnctl startproc process-type=PreferenceServer
```

Change Surviving Servers to Use the Preference Store on APPHOST2

Now that the preference server has been started on APPHOST2, amend all discoverer instances to use this preference server, including APPHOST2.

The preference store being used is determined at the startup of the WebLogic Managed Server WLS_DISCO1. In order to get WLS_DISCO1 to use a different preference store the startup parameters must be changed, by logging into the WebLogic Administration Console using the following URL:

http://apphost1.mycompany.com:7001/console

Continue with the following steps:

1. Select **Environment -> Servers** from the Domain Structure menu.
2. Click on **Lock and Edit** in the change center window
3. Click on the server WLS_DISCO1
4. Click on the **Configuration** Tab and the **Server Start** sub tab.
5. In the arguments field append the following

   ```
   -Doracle.disco.activation.preferencePort=<portno>
   -Doracle.disco.activation.preferenceHost=<hostname>
   ```

   the port number is the value of PREFERENCE_PORT defined previously.

   The hostname is the name of the host on which the preference server is started, for example, APPHOST2.

12.6.5.7.8 Best Practices Oracle WebLogic Server does not replicate configuration information automatically between nodes. Therefore, it is important that any changes to any of the following be manually propagated to the other servers in the deployment:

- ORACLE_INSTANCE
- *DOMAIN_HOM*E/servers/WLS_ DISCO1/stage/discoverer/11.1.1.1.0/discoverer/configuration/configuration.xm l

Preference Server

The preference server contains details about user preferences, which can be updated on a frequent basis. In the event of the failure of the Oracle Discoverer preference server, the preference server must be started on another server. The preference information therefore must be copied to these potential hosts on a regular basis.

To this end the following file needs to be propagated to other servers, which could host the preference server on a regular basis:

`ORACLE_INSTANCE`/config/PreferenceServer/Discoverer_Instance/.reg_key.dc

12.6.6 Scaling Out the Deployment

To scale out the deployment to an additional node, follow the steps in Section 12.6.5, "Install and Configure Application Tier on APPHOST2." Be sure to use the same directory structure used for previous nodes.

A Setting Up Auditing with an Oracle RAC Database Store

With Oracle Fusion Middleware 11*g*, you have the option of setting up the Oracle Fusion Middleware Audit Framework service. Auditing provides a measure of accountability and answers "who has done what and when" types of questions.

The Oracle Fusion Middleware Audit Framework is designed to provide a centralized audit framework for middleware products. The framework provides audit service for the following:

- Middleware Platform - This includes Java components such as Oracle Platform Security Services (OPSS) and Oracle Web Services. These are components that are leveraged by applications deployed in the middleware. Indirectly, all the deployed applications leveraging these Java components will benefit from the audit framework auditing events that are happening at the platform level.
- JavaEE applications - The objective is to provide a framework for JavaEE applications, starting with Oracle's own Java components. JavaEE applications will be able to create application-specific audit events. In the current release, the Java EE components using the Oracle Fusion Middleware Audit Framework are internal Oracle components.
- System components - For system components in the middleware that are managed by Oracle Process Manager and Notification Server (OPMN), the audit framework also provides an end-to-end service similar to that for Java components.

See the "Introduction to Oracle Fusion Middleware Audit Framework" chapter in the *Oracle Fusion Middleware Security Guide* for more introductory information about Oracle Fusion Middleware Audit Framework.

Out of the box, the Audit Framework uses the file system to store audit records. In a production environment, however, Oracle recommends that you use a database audit store to provide scalability and high availability for the audit framework. In high availability configurations such as the configurations described in this chapter, Oracle recommends that you use an Oracle Real Application Clusters (RAC) database as the database audit store.

The "Configuring and Managing Auditing" chapter in the *Oracle Fusion Middleware Security Guide* includes the steps for configuring auditing. The "Managing the Audit Store" section in that chapter includes steps for setting up a database as the audit data store.

When you set up the Oracle Fusion Middleware Audit Framework with an Oracle RAC database audit store, you must manually configure the following:

- Data sources and multi data sources for the audit data source using WebLogic Server
- The JDBC string for the OPMN loader in the opmn.xml file

The following sections provide additional information specific to configuring auditing when an Oracle RAC database is used as the audit data store.

A.1 Using WebLogic Server to Configure Audit Data Sources and Multi Data Sources

To set up the audit data source and multi data sources for an Oracle RAC database, follow the instructions in the "Managing the Audit Store" section of the *Oracle Fusion Middleware Security Guide*. Use the information in the "Set Up Audit Data Sources" section to set up the audit data sources and the information in the "Multiple Data Sources" section to configure an Oracle RAC database as the audit data store.

Use the information in the "Set Up Audit Data Sources" section to set up the audit data sources. To use an Oracle RAC database as the audit data store, you must create two individual data sources pointing to each individual Oracle RAC instance where the audit schemas are installed. The following settings are required:

- The connection URL should be in the following format:

  ```
  jdbc:oracle:thin:@(DESCRIPTION=(ADDRESS=PROTOCOL=TCP)(HOST=host-vip)
  (PORT=1521))(CONNECT_DATA=(SERVICE_NAME=dbservice)(INSTANCE_NAME=inst1))
  ```

  Note that the service name and instance name are required, in addition to the host and port.

- The driver used is oracle.jdbc.OracleDriver
- The following property should be set:

  ```
  <property>
  <name>oracle.net.CONNECT_TIMEOUT</name>
  <value>10000</value>
  </property>
  ```

- The following settings are required for the individual data sources:
  - initial-capacity: **0**
  - connection-creation-retry-frequency-seconds: **10**
  - test-frequency-seconds: **300**
  - test-connections-on-reserve: **true**
  - test-table-name: **SQL SELECT 1 FROM DUAL**
  - seconds-to-trust-an-idle-pool-connection: **0**
  - global-transactions-protocol: **None**

Use the information in the "Multiple Data Sources" section to configure an Oracle RAC database as the audit data store. Create a multi data source with JNDI name jdbc/AuditDB. This multi data source should point to the individual data sources you created.

The following settings are required for the multi data source:

- test-frequency-seconds: **5**
- algorithm-type: **Load-Balancing**

- data-source-list: point to a list of comma separated child data sources **("JDBC Data Source-0,JDBC Data Source-1")**. This list is the same set of data sources that you created for each individual node of the Oracle RAC database.

A.2 Configuring the JDBC String for the Audit Loader

If you have an audit store configured, Oracle Process Manager and Notification Server (OPMN) manages several system components running in Oracle WebLogic Server. For these components, OPMN pushes the audit events to the database audit store.

The "Configure a Database Audit Store for System Components" section in the *Oracle Fusion Middleware Security Guide* describes how to set up the OPMN startup audit loader.

During the setup of the OPMN startup audit loader, you must modify the `rmd-definitions` element in the `opmn.xml` file. By default, the `rmd-definitions` element includes a JDBC string for a single instance database in this format:

```
jdbc:oracle:thin:@host:port:sid
```

When you are using an Oracle RAC database as the audit data store, you must use a JDBC string for an Oracle RAC database in the following format in the `rmd-definitions` element:

```
jdbc:oracle:thin@(DESCRIPTION=(ADDRESS_LIST=(LOAD_BALANCE=on)(ADDRESS=(PROTOCOL=
tcp)(HOST=node1-vip)(PORT=1521))(ADDRESS=(PROTOCOL=tcp)(HOST=node2-vip)(PORT=1521)
))(CONNECT_DATA=SERVICE_NAME=service-name.mycompany.com)))
```

If you also need to configure the Oracle RAC database audit store for Java components, refer to the instructions in the "Configure a Database Audit Store for Java Components" section in the *Oracle Fusion Middleware Security Guide*.

B Recommended Multi Data Sources

The following are examples of multi data source configuration files. Multi pool DS JDBC Multi Data Source-0 is made up of two data sources, JDBC Data Source-0, and JDBC Data Source-1. The property settings in bold text are recommended settings:

B.1 JDBC Multi Data Source-0

```
<?xml version='1.0' encoding='UTF-8'?>
<jdbc-data-source xmlns="http://www.bea.com/ns/weblogic/jdbc-data-source"
 xmlns:sec="http://www.bea.com/ns/weblogic/90/security"
 xmlns:wls="http://www.bea.com/ns/weblogic/90/security/wls"
 xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
 xsi:schemaLocation="http://www.bea.com/ns/weblogic/jdbc-data-source
 http://www.bea.com/ns/weblogic/jdbc-data-source/1.0/jdbc-data-source.xsd">
  <name>JDBC Multi Data Source-0</name>
  <jdbc-connection-pool-params>
    <test-frequency-seconds>5</test-frequency-seconds>
  </jdbc-connection-pool-params>
  <jdbc-data-source-params>
    <jndi-name>jdbc/OracleDS</jndi-name>
    <algorithm-type>Load-Balancing</algorithm-type>
    <data-source-list>JDBC Data Source-0,JDBC Data Source-1</data-source-list>
    <failover-request-if-busy>false</failover-request-if-busy>
  </jdbc-data-source-params>
</jdbc-data-source>
```

B.2 JDBC Data Source-0 (non-XA)

```
<?xml version='1.0' encoding='UTF-8'?>
<jdbc-data-source xmlns="http://www.bea.com/ns/weblogic/jdbc-data-source"
 xmlns:sec="http://www.bea.com/ns/weblogic/90/security"
 xmlns:wls="http://www.bea.com/ns/weblogic/90/security/wls"
 xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
 xsi:schemaLocation="http://www.bea.com/ns/weblogic/jdbc-data-source
 http://www.bea.com/ns/weblogic/jdbc-data-source/1.0/jdbc-data-source.xsd">
  <name>JDBC Data Source-0</name>
  <jdbc-driver-params>
    <url>jdbc:oracle:thin:@(DESCRIPTION=(ADDRESS=(PROTOCOL=TCP)(HOST=host-vip)
(PORT=1521))(CONNECT_DATA=(SERVICE_NAME=dbservice)(INSTANCE_NAME=inst1)))</url>
    <driver-name>oracle.jdbc.OracleDriver</driver-name>
    <properties>
      <property>
        <name>user</name>
        <value>username</value>
      </property>
```

```
        <property>
          <name>oracle.net.CONNECT_TIMEOUT</name>
          <value>10000</value>
        </property>
      </properties>
      <password-encrypted>{3DES}LMWCj6TOOuI=</password-encrypted>
    </jdbc-driver-params>
    <jdbc-connection-pool-params>
      <initial-capacity>0</initial-capacity>
      <max-capacity>20</max-capacity>
      <capacity-increment>1</capacity-increment>
      <shrink-frequency-seconds>900</shrink-frequency-seconds>
      <highest-num-waiters>2147483647</highest-num-waiters>

<connection-creation-retry-frequency-seconds>10</connection-creation-retry-frequen
cy-seconds>
      <connection-reserve-timeout-seconds>10</connection-reserve-timeout-seconds>
      <test-frequency-seconds>300</test-frequency-seconds>
      <test-connections-on-reserve>true</test-connections-on-reserve>
      <ignore-in-use-connections-enabled>true</ignore-in-use-connections-enabled>
      <inactive-connection-timeout-seconds>0</inactive-connection-timeout-seconds>
      <test-table-name>SQL SELECT 1 FROM DUAL</test-table-name>
      <login-delay-seconds>0</login-delay-seconds>
      <statement-cache-size>10</statement-cache-size>
      <statement-cache-type>LRU</statement-cache-type>
      <remove-infected-connections>true</remove-infected-connections>

<seconds-to-trust-an-idle-pool-connection>0</seconds-to-trust-an-idle-pool-connect
ion>
      <statement-timeout>-1</statement-timeout>
      <pinned-to-thread>false</pinned-to-thread>
    </jdbc-connection-pool-params>
    <jdbc-data-source-params>
      <jndi-name>jdbc/OracleDS0</jndi-name>
      <global-transactions-protocol>None</global-transactions-protocol>
    </jdbc-data-source-params>
</jdbc-data-source>
```

B.3 JDBC Data Source-0 (XA)

```
<?xml version='1.0' encoding='UTF-8'?>
<jdbc-data-source xmlns="http://www.bea.com/ns/weblogic/jdbc-data-source"
 xmlns:sec="http://www.bea.com/ns/weblogic/90/security"
 xmlns:wls="http://www.bea.com/ns/weblogic/90/security/wls"
 xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
 xsi:schemaLocation="http://www.bea.com/ns/weblogic/jdbc-data-source
 http://www.bea.com/ns/weblogic/jdbc-data-source/1.0/jdbc-data-source.xsd">
  <name>JDBC Data Source-0</name>
  <jdbc-driver-params>
    <url>
jdbc:oracle:thin:@(DESCRIPTION=(ADDRESS=(PROTOCOL=TCP)(HOST=host-vip)(PORT=1521))
(CONNECT_DATA=(SERVICE_NAME=dbservice)(INSTANCE_NAME=inst1)))</url>
    <driver-name>oracle.jdbc.xa.client.OracleXADataSource</driver-name>
    <properties>
      <property>
        <name>user</name>
        <value>username</value>
      </property>
      <property>
        <name>oracle.net.CONNECT_TIMEOUT</name>
        <value>10000</value>
      </property>
    </properties>
    <password-encrypted>{3DES}LMWCj6TOOuI=</password-encrypted>
    <use-xa-data-source-interface>true</use-xa-data-source-interface>
  </jdbc-driver-params>
  <jdbc-connection-pool-params>
    <initial-capacity>0</initial-capacity>
<connection-creation-retry-frequency-seconds>10</connection-creation-retry-frequen
cy-seconds>
    <max-capacity>15</max-capacity>
    <capacity-increment>1</capacity-increment>
    <shrink-frequency-seconds>900</shrink-frequency-seconds>
    <highest-num-waiters>2147483647</highest-num-waiters>
    <connection-reserve-timeout-seconds>10</connection-reserve-timeout-seconds>
    <test-frequency-seconds>300</test-frequency-seconds>
    <test-connections-on-reserve>true</test-connections-on-reserve>
    <ignore-in-use-connections-enabled>true</ignore-in-use-connections-enabled>
    <inactive-connection-timeout-seconds>0</inactive-connection-timeout-seconds>
    <test-table-name>SQL SELECT 1 FROM DUAL</test-table-name>
    <login-delay-seconds>0</login-delay-seconds>
    <statement-cache-size>10</statement-cache-size>
    <statement-cache-type>LRU</statement-cache-type>
    <remove-infected-connections>true</remove-infected-connections>
<seconds-to-trust-an-idle-pool-connection>0</seconds-to-trust-an-idle-pool-connect
ion>
    <statement-timeout>-1</statement-timeout>
    <jdbc-xa-debug-level>10</jdbc-xa-debug-level>
    <pinned-to-thread>false</pinned-to-thread>
  </jdbc-connection-pool-params>
  <jdbc-data-source-params>
    <jndi-name>jdbc/OracleDS1</jndi-name>
    <global-transactions-protocol>TwoPhaseCommit</global-transactions-protocol>
  </jdbc-data-source-params>
  <jdbc-xa-params>
    <keep-xa-conn-till-tx-complete>true</keep-xa-conn-till-tx-complete>
    <need-tx-ctx-on-close>false</need-tx-ctx-on-close>
    <xa-end-only-once>false</xa-end-only-once>
    <keep-logical-conn-open-on-release>false</keep-logical-conn-open-on-release>
    <resource-health-monitoring>true</resource-health-monitoring>
    <recover-only-once>false</recover-only-once>
    <xa-set-transaction-timeout>false</xa-set-transaction-timeout>
    <xa-transaction-timeout>0</xa-transaction-timeout>
    <rollback-local-tx-upon-conn-close>false</rollback-local-tx-upon-conn-close>
    <xa-retry-duration-seconds>300</xa-retry-duration-seconds>
    <xa-retry-interval-seconds>60</xa-retry-interval-seconds>
  </jdbc-xa-params>
</jdbc-data-source>
```

The only difference between JDBC Data Source-1 for XA and non-XA is they point to a different instance of Oracle RAC (host:port).

C Whole Server Migration for Windows

This appendix describes the procedure for enabling a virtual IP address for Windows XP systems.

C.1 Using Windows Control Panel

This procedure assumes the system is windows XP, configured for static IP addresses.

1. Go to **Control Panel**.
2. Select **Network and Internet Connections**.
3. Select **Network Connections**.
4. Select **Local Area Connection**.
5. Choose **Internet Protocol (TCP / IP)** and click the **Properties** button.
6. In the next window, click the **Advanced** button at the bottom of the dialog.
7. Click the **Add** button next to the displayed IP addresses to add an IP address with a corresponding sub-net mask. (this works only if the system is configured for static IPs)

C.2 Using the netsh Command Line

The command line method is to use `netsh`. For example:

```
netsh interface ip show address <interface>
netsh interface ip add address <interface> <address> <netmask>
netsh interface ip delete address <interface> addr=<address>
```

The following commands are available:

Commands in this context:

Table C–1 Application URLs

Application	URL
?	Displays a list of commands.
add	Adds a configuration entry to a table
delete	Deletes a configuration entry from a table.
dump	Displays a configuration script.
help	Displays a list of commands.

Table C–1 (Cont.) Application URLs

Application	**URL**
ip	Changes to the `netsh interface ip' context.
ipv6	Changes to the `netsh interface ipv6' context.
portproxy	Changes to the `netsh interface portproxy' context.
reset	Resets information.
set	Sets configuration information
show	Displays information.

The following sub-contexts are available:

```
 ip ipv6 portproxy
```

To view Help for a command, type the command, followed by a space, and then **?**.

D Oracle SOA Suite Workbook

This appendix is a workbook for Section 5.11, "Configuring High Availability for Oracle SOA Service Infrastructure and Component Service Engines.".

Using the information in Chapter 5.11, "Configuring High Availability for Oracle SOA Service Infrastructure and Component Service Engines" as an example, you can fill out the equivalent names that you plan to use in your environment when configuring high availability for the Oracle SOA Suite.

D.1 Workbook Tables for Oracle SOA Suite

Use Table D–1 for application URLs.

Table D–1 Application URLs

Application	URL
SOA-INFRA Application	
SOA Worklist Application	
WSM-PM Validation Application	
UMS Preferences Application	
B2B UI Application	
BAM Application	
Admin Console	
Web Logic Console	
FMW Control	

Use Table D–2 forVitrual IP Addresses.

Table D–2 Virtual IP Addresses

Purpose	IP Address	DNS Name
Administration Server		
WLS_WSM1		
WLS_WSM2		
WLS_SOA1		
WLS_SOA2		

Table D–2 (Cont.) Virtual IP Addresses

Purpose	IP Address	DNS Name
WLS_BAM1		
WLS_BAM2		

Use Table D–3 for file locations - Application Tier.

Table D–3 File Locations - Application Tier

Environment Variable or Artifact	Location	Shared
ORACLE_BASE		No
MW_HOME		Yes
ORACLE_HOME		Yes
WLS_HOME		Yes
DOMAIN_HOME		No
TLOG Directory		Yes
JMS Persistence Store		Yes

Use Table D–4 for file locations - Web Tier.

Table D–4 File Locations - Web Tier

Environment Variable or Artifact	Location	Shared
ORACLE_BASE		No
MW_HOME		Yes
ORACLE_HOME		Yes
ORACLE_INSTANCE		No

Use Table D–5 for Identity Management Details.

Table D–5 Identity Management Details

Identity Management Artifact	Value
LDAP Host Name	
LDAP Host	
LDAP SSL Port	
LDAP user DN	
LDAP Password	
Access Manager Host	
Access Manager Port	
Application Profile Password	

Use Table D–6 for Database Information

Table D–6 Database Information - Metadata Repository

SOA Database Artifact	Value
Database Hosts (VIPS if using RAC)	
Listener Port	
Database Service Name	
RCU Prefix	
Main Schema User/Password	
Auxiliary Schema User/Password	
MDS User/Password	
User Messaging User/Password	
BAM User/Password	
SOA Infrastructure User/Password	
SYS Password	

Use Table D–7 for Load Balancing Configuration

Table D–7 Load Balancer Configuration

Purpose	Virtual Name	Port	Externally Visible	SSL	Destination Hosts	Destination Ports	SSL
External Site							
Internal Site							
Administration Site							

Use Table D–8 for Port Information

Table D–8 Port Information

Component	Host(s)	Port
SSL Port on the Load Balancer	Load Balancer	
HTTP Port on Load Balancer	Load Balancer	
OHS HTTP Listening		
OHS Admin		
OPMN		
Weblogic Administration Server		
Single Sign-on Listening		
WLS_WSM1 listen address		
WLS_WSM2 listen address		
WLS_SOA1 listen address		

Table D–8 (Cont.) Port Information

Component	Host(s)	Port
WLS_SOA2 listen address		
WLS_BAM1 listen address		
WLS_BAM2 listen address		
Node Manager SOAHOST1		
Node Manager SOAHOST2		
Node Manager BAMHOST1		
Node Manager BAMHOST2		

E

Identity Management Workbook

This appendix is a workbook for Chapter 7, "Configuring High Availability for Identity Management Components."

Using the information in Chapter 7, "Configuring High Availability for Identity Management Components" as an example, you can fill out the equivalent names that you plan to use in your environment when configuring high availability for the Oracle Identity Management components.

E.1 Workbook Tables for Oracle Identity Management

Use the following tables to record the names you plan to use in your Oracle Identity Management high availability configuration.

Enter the application URLS for your configuration in Table E–1.

Table E–1 Application URLs

Application	URL
Oracle WebLogic Administration Console	
Oracle Enterprise Manager Fusion Middleware Control	

Enter the virtual IP address for your configuration in Table E–2.

Table E–2 Virtual IP Addresses

Purpose	IP Address	DNS Name
Oracle WebLogic Administration Server		

Enter the following generic file locations for your configuration in Table E–3.

Table E–3 Generic File Locations

Type	Location	Shared
ORACLE_BASE		No
MW_HOME		No

Enter the file locations for IDMHOST*n* for your configuration in Table E–4.

Table E–4 File Locations for IDMHOSTn

Environment Artifact	Directory Location	Shared
ORACLE_HOME		No
Administration Server ORACLE_INSTANCE		No
WLS_ODS1 ORACLE_ INSTANCE		No
WLS_ODS2 ORACLE_ INSTANCE		No
WLS_HOME		No
DOMAIN_HOME		No

Enter the file locations for OIDHOST*n* for your configuration in Table E–5.

Table E–5 File Locations for OIDHOSTn

Environment Artifact	Directory Location	Shared
ORACLE_HOME		No
ORACLE_INSTANCE		No

Enter the file locations for OVDHOST*n* for your configuration in Table E–6.

Table E–6 File Locations for OVDHOSTn

Environment Artifact	Directory Location	Shared
ORACLE_HOME		No
ORACLE_INSTANCE		No

Enter the file locations for OIFHOST*n* for your configuration in Table E–7.

Table E–7 File Locations for OIFHOSTn

Environment Artifact	Directory Location	Shared
ORACLE_HOME		No
Administration Server ORACLE_INSTANCE		No
WLS_OIF1 ORACLE_ INSTANCE		No
WLS_OIF2 ORACLE_ INSTANCE		No
WLS_HOME		No
DOMAIN_HOME		No

Enter the file locations for the web tier for your configuration in Table E–8.

Table E–8 File Locations for the Web Tier

Environment Artifact	Directory Location	Shared
ORACLE_HOME		No

Table E–8 (Cont.) File Locations for the Web Tier

Environment Artifact	Directory Location	Shared
ORACLE_INSTANCE		No

Enter Oracle Identity Management details for your configuration in Table E–9.

Table E–9 Identity Management Artifacts

Identity Management Artifact	Value
Single Sign-On URL	
Oracle Internet Directory Host Name	
Oracle Internet Directory Non-SSL Port	
Oracle Internet Directory SSL Enabled	
Oracle Internet Directory SSL Port	
Oracle Internet Directory Security Realm	
Oracle Virtual Directory Host Name	
Oracle Virtual Directory Port	
Oracle Virtual Directory SSL Enabled	
Oracle Virtual Directory SSL Port	

Enter Authentication LDAP Artifacts details for the Oracle Identity Federation configuration in Table E–10.

Table E–10 Authentication LDAP Artifacts for Oracle Identity Federation

Authentication LDAP Artifact	Value
LDAP Type	
LDAP URL	
LDAP Bind DN	
LDAP Bind DN Password	
User Credential ID Attribute	
User Unique ID Attribute	
Person Object Class	
Base DN	

Enter the Oracle Identity Federation Artifacts when using an LDAP User Data Store in your configuration in Table E–11.

Table E–11 LDAP User Data Store Artifacts for Oracle identity Federation

LDAP User Data Store Artifact	Value
LDAP Type	
LDAP URL	
LDAP Bind DN	
LDAP Bind DN Password	
User Description Attribute	
User ID Attribute	
Person Object Class	
Base DN	

Enter the Oracle Identity Federation Artifacts when using an LDAP Federation Data Store in your configuration in Table E–12.

Table E–12 LDAP Federation Data Store Artifacts for Oracle Identity Federation

Federation Data Store Artifact	Value
LDAP Type	
LDAP URL	
LDAP Bind DN	
LDAP Bind DN Password	
User Federation Record Context	
LDAP Container Object Class	

Enter the Oracle Identity Federation Artifacts when using an RDBMS User Data Store in your configuration in Table E–13.

Table E–13 RDBMS User Data Store Artifacts for Oracle Identity Federation

RDBMS User Data Store Artifact	Value
Hostname	
Username	
Password	
Login Table	
User ID Attribute	
User Description Attribute	

Enter the Oracle Identity Federation Artifacts when using an RDBMS Federation Data Store in your configuration in Table E–14.

Table E–14 RDBMS Federation Data Store Artifacts for Oracle Identity Federation

RDBMS Federation Data Store Artifacts	Value
Hostname	
Username	
Password	

Enter RDBMS Transient Data Store Artifacts for the Oracle Identity Federation configuration in Table E–15.

Table E–15 RDBMS Transient Data Store Artifacts for Oracle Identity Federation

Transient Data Store Artifact	Value
Hostname	
Username	
Password	

Enter database information for the metadata repository for your configuration in Table E–16.

Table E–16 Database Information for the Metadata Repository

Database Details	Value
Database Hosts (VIPs if using Oracle RAC)	
Listener Port	
Database Service Name	
RCU Prefix	
Main Schema Name/Password	
Auxiliary Schema Name/Password	
SYS Password	

Enter load balancer configuration information for your configuration in Table E–17.

Table E–17 Load Balancer Configuration

Purpose	Virtual Name	Port	Externally Visible	SSL	Destination Hosts	Destination Ports	SSL
Oracle Internet Directory							
Oracle Virtual Directory							
Administration Server							
Single Sign-On							

Enter port information for your configuration in Table E–18.

Table E–18 Port Information

Component	Host(s)	Port
Oracle Internet Directory		
Oracle Internet Directory (SSL)		
Oracle Virtual Directory		
Oracle Virtual Directory (SSL)		
WebLogic Server Console		
Oracle Enterprise Manager Fusion Middleware Control		
Oracle Directory Services Manager		
Oracle HTTP Server		
Oracle HTTP Server (SSL)		
Oracle HTTP Server Admin		
OPMN		
Node Manager		

F WebCenter Workbook

This appendix is a workbook for Section 6.4, "Configuring High Availability for Oracle WebCenter."

Using the information in Chapter 6.4, "Configuring High Availability for Oracle WebCenter" as an example, you can fill out the equivalent names that you plan to use in your environment when configuring high availability for the Oracle WebCenter components.

F.1 Workbook Tables for Oracle WebCenter

Use the following tables to record the names you plan to use in your Oracle WebCenter high availability configuration.

Use Table F–1 for application URLs.

Table F–1 Application URLs

Application	URL
WebCenter Spaces Application	
Portal Tools Application	
WSM Policy Manager Application	
WebCenter Discussions Forum	
WebCenter Wiki	
Admin Console	
WebLogic Console	
FMW Control	

Use Table F–2 for Virtual IP Addresses.

Table F–2 Virtual IP Addresses

Purpose	IP Address	DNS Name
Administration Server		

Use Table F–3 for File Locations - App Tier.

Table F–3 File Locations - App Tier

Type	Location	Shared
ORACLE_BASE		No
MW_HOME		Yes
ORACLE_HOME		Yes
WLS_HOME		Yes
DOMAIN_HOME		No
ADMIN_DOMAIN_HOME		Yes

Use Table F–4 for File Locations - Web Tier.

Table F–4 File Locations - Web Tier

Application	URL	Shared
ORACLE_BASE		No
MW_HOME		Yes
ORACLE_HOME		Yes
WLS_HOME		Yes
DOMAIN_HOME		No
ADMIN_DOMAIN_HOME		Yes

Use Table F–5 for Identity Management Details.

Table F–5 Identity Management Details

LDAP Host Name
LDAP Port
LDAP SSL Enabled
LDAP user DN
LDAP Password
Access Manager Host
Access Manager Port
Application Profile Password

Use Table F–6 for Database Information - Metadata Repository.

Table F–6 Database Information - Metadata Repository

Database Hosts (VIPS if using RAC)
Listener Port
Database Service Name
RCU Prefix
Main Schema User/Password

Table F–6 (Cont.) Database Information - Metadata Repository

Auxiliary Schema User/Password	
MDS User/Password	
WebCenter Spaces User/Password	
Portlets User/Password	
Discussion Forum User/Password	
Wiki User/Password	
Content Server User/Password	
SYS Password	

Use Table F–7 for Load Balancer Configuration

Table F–7 Load Balancer Configuration

Purpose	Virtual Name	Port	Externally Visible	SSL	Destination Hosts	Destination Ports
WC Application						
Admin						
WC Internal Application						

Use Table F–8 for Port Information

Table F–8 Port Information

Comment	Host(s)	Port
SSL Port on the Load Balancer	Load Balancer	
HTTP Port on Load Balancer	Load Balancer	
OHS HTTP Listening		
OHS Admin		
WebLogic Administration Server		
OPMN		
Single Sign on Listening		
WLS_WSM1		
WLS_WSM2		
WLS_SPACES1		
WLS_SPACES2		
WLS_PORTLETS1		
WLS_PORTLETS2		
WLS_SERVICES1		
WLS_SERVICES2		

Table F–8 (Cont.) Port Information

Comment	Host(s)	Port
Node Manager		

G Oracle Portal, Forms, Reports, and Discoverer Workbook

This appendix is a workbook for Section 12.6, "Configuring Oracle Portal, Forms, Reports, and Discoverer for High Availability.".

Using the information in Chapter 12.6, "Configuring Oracle Portal, Forms, Reports, and Discoverer for High Availability" as an example, you can fill out the equivalent names that you plan to use in your environment when configuring high availability for the Oracle Portal, Forms, Reports, and Discoverer.

G.1 Workbook Tables for Oracle Portal, Forms, Reports, and Discoverer

Use Table G–1 for application URLs.

Table G–1 Application URLs

Application	URL
Portal	
Forms	
Reports	
Discoverer	
Web Logic Console	
FMW Control	

Use Table G–2 for Virtual IP Addresses.

Table G–2 Virtual IP Addresses

Purpose	IP Address	DNS Name
Administration Server		

Use Table G–3 for file locations.

Table G–3 File Locations

Type	Location	Shared
ORACLE_BASE		No
MW_HOME		No
ORACLE_HOME		No

Table G–3 (Cont.) File Locations

Type	Location	Shared
ORACLE_INSTANCE		No
WLS_HOME		No
DOMAIN_HOME		No
Reports Cache		Yes

Use Table G–4 for Identity Management Details.

Table G–4 Database Information - Metadata Repository

Single Sign-On URL	
OID Host Name	
OID Port	
OID SSL Enabled	

Use Table G–5 for Database Information.

Table G–5 Database Information - Metadata Repository

Database Hosts (VIPS if using RAC)	
Listener Port	
Database Service Name	
RCU Prefix	
Portal User Name/Password	
Portlet User Name/Password	
Discoverer User Name/Password	
SYS Password	

Use Table G–6 for Load Balancing Configuration.

Table G–6 Load Balancer Configuration

Purpose	Virtual Name	Port	Externally Visible	SSL	Destination Hosts	Destination Ports
Main Site						
Internal Site						
Web Cache Invalidations						

Use Table G–7 for Port Information.

Table G–7 Port Information

Comment	Host(s)	Port
SSL Port on the Load Balancer	Load Balancer	
HTTP Port on Load Balancer	Load Balancer	
Web Cache HTTP		
Web Cache Invalidation		
Web Cache Administration		
Web Cache Statistics		
OHS HTTP Listening		
OHS HTTPS Listening		
OHS Admin		
WebLogic Administration Server		
OPMN		
Single Sign on Listening		
WebLogic Managed Server		
WebLogic Managed Server		
OID HTTP		
WLS_PORTAL		
WLS_PORTAL1		
WLS_FORMS		
WLS_FORMS1		
WLS_REPORTS		
WLS_REPORTS1		
WLS_DISCO		
WLS_DISCO1		
Node Manager		

H ascrsctl Online Help

The detailed usage of ascrsctl can be obtained by following the instructions generated from the command `ascrsctl help`. This appendix provides the full content of the help pages to serve as a complete offline reference.

H.1 create/as

TOPIC

```
create/as - create as ASCRS resource
```

COMMAND

```
ascrsctl create -name <string> -type as -componentHome <string>
            -vip <string> -disk <string> [<string> ...]
            [options]
```

DESCRIPTION

This ASCRS command is used to create (or register) a WebLogic Administration Server resource in CRS.

These are mandatory arguments:

-name, -n

This argument specifies the resource name to be created.

-type, -t

This argument specifies the resource type. Its value must be **as**.

-componentHome, -ch

This argument specifies the location of WebLogic domain that contains the targeted administration server.

-vip

This argument specifies the virtual IP resource the WebLogic Administration Server depends upon.

-disk

This argument specifies the shared disks hosting the WebLogic software, the administration server domain directory and other shared disks this resource directly depends upon. If a shared disk is used for multiple purposes, it needs to be specified only once. The value for this parameter is a space or comma-separated list of ASCRS disk resource names.

The other non-mandatory options:

-clusterNodes, -nodes, -cn

This argument specifies the valid nodes of the cluster across which the resource can failover. The value is a space or comma-separated subset of (or all) nodes of a cluster. If it is not specified, all the nodes are included.

-resourceParams, -params, -p

This argument specifies a string of comma separated name value pairs to set CRS configuration for a specific resource, for example, **as**=1, **rt**=400.

The resource values are listed in Table H–1. Except for **as**, **st** and **ra**, all the other numbers are in seconds.

Table H–1 Resource Values

Parameter	Min	Max	Default	Purpose
as	0	1	1	Auto Start
ci	5	6000	600	Check Interval
fd	5	600	50	Failure Delay
fi	5	6000	50	Failure Interval
ft	0	20	5	Failure Threshold
ra	0	20	3	Restart Attempts
st	20	900	30	Script Timeouts
rt	20	900	30	Start Timeout
pt	20	900	30	Stop Timeout

-policy

This argument specifies what resource parameter values are used. These policies and values are available in the ascrs.properties file.

The valid values are **normal** or **fast**. If it not specified **normal** is assumed. However, the **-resourceParams** values always take precedence over the policy.

EXAMPLE(S)

```
ascrsctl create -n idm.weblogic -t as
          -ch /sharedisk/fmw/user_projects/domains/IDMDomain
          -vip idmvip -disk idmdisk

ascrsctl create -n idm.weblogic -t as
          -ch /sharedisk/fmw/user_projects/domains/IDMDomain
          -vip idmvip -disk idmdisk -p st=800,rt=800,pt=800

ascrsctl create -n idm.opmn -t as
          -ch /sharedisk/fmw/asinst_1
          -vip idmvip -disk idmdisk -p st=800,rt=800,pt=800
```

H.2 create/db

TOPIC

create/db - create **db** ASCRS resource

COMMAND

```
ascrsctl create -name <string> -type db -oraHomestring>
            -oraSID <string> -disk <string> [<string> ...]
            -lsnr <string> [-pfile <string>] [options]
```

DESCRIPTION

This ASCRS command is used to create (or register) an Oracle database resource in CRS.

These are mandatory arguments:

-name, -n

This argument specifies the resource name to be created.

-type, -t

This argument specifies the resource type. Its value must be **db**.

-oraHome, -oh

This argument specifies the database Oracle Home location.

-oraSID, -sid

This argument specifies the Oracle SID name.

-disk

This argument specifies the shared disks hosting the Oracle Home and data files. The value for this parameter is a space or comma-separated list of ASCRS disk resource names.

-lsnr

This argument specifies the listener resource used by this database.

The other non-mandatory options:

-**clusterNodes, -nodes, -cn**

This argument specifies the valid nodes of the cluster across which the resource can failover.

The value is a space or comma-separated subset of (or all) nodes of a cluster. If it is not specified, all the nodes are included.

-resourceParams, -params, -p

This argument specifies a string of comma separated name value pairs to set CRS configuration for a specific resource, for example, **as**=1,**rt**=400.

The resource values are listed in Table H–1. Except for **as**, **st** and **ra**, all the other numbers are in seconds.

-policy

This argument specifies what resource parameter values are used. These policies and values are available in the ascrs.properties file.

The valid values are **normal** or **fast**. If it not specified **normal** is assumed. However, the **-resourceParams** values always take precedence over the policy.

-pfile, -pf

This argument specifies the database pfile for starting the database.

EXAMPLE(S)

```
ascrsctl create -n mydb -t db -oh /cfcdb1 -sid orcl
          -disk ohdisk datafiledisk -lsnr mydblsnr
```

H.3 create/dnlsnr

TOPIC

```
create/dnlsnr - create dblsnr ASCRS resource
```

COMMAND

```
ascrsctl create -name <string> -type dblsnr -listenerName <string>
          -listenerOracleHome <string>
          -vip <string> -disk <string>
          [-tnsAdmin <string>] [options]
```

DESCRIPTION

This ASCRS command is used to create (or register) an Oracle database listener resource in CRS.

These are mandatory arguments:

-name, -n

This argument specifies the resource name to be created.

-type, -t

This argument specifies the resource type. Its value must be **dblsnr**.

-listnerOracleHome, -lsnroh, -loh

This argument specifies the Oracle Home of the database that owns this listener.

-vip

This argument specifies the vip resource this listener runs on.

-disk

This argument specifies the disk resource the Oracle Home resides on.

-lsnr

This argument specifies the listener resource used by this database.

The other non-mandatory options:

-**clusterNodes, -nodes, -cn**

This argument specifies the valid nodes of the cluster across which the resource can failover.

The value is a space or comma-separated subset of (or all) nodes of a cluster. If it is not specified, all the nodes are included.

-resourceParams, -params, -p

This argument specifies a string of comma separated name value pairs to set CRS configuration for a specific resource, for example, **as**=1,**rt**=400.

The resource values are listed in Table H–1. Except for **as**, **st** and **ra**, all the other numbers are in seconds.

-policy

This argument specifies what resource parameter values are used. These policies and values are available in the ascrs.properties file.

The valid values are **normal** or **fast**. If it not specified **normal** is assumed. However, the **-resourceParams** values always take precedence over the policy.

-tnsAdmin, -ta

This argument specifies the location of the listener configuration if it is not in the default location within the Oracle Home.

EXAMPLE(S)

```
ascrsctl create -name mydblsnr -type dblsnr -listenerName orcl
          -listenerOracleHome /cfcdb1
          -vip 192.168.1.10 -disk ohdisk
```

H.4 create/disk

TOPIC

```
create/disk - create disk ASCRS resource
```

COMMAND

```
ascrsctl create -name <string> -type disk -path <string>
         -mountCommand <string> -umountCommand <string>
         [options]
```

DESCRIPTION

This ASCRS command is used to create (or register) a shared disk resource in CRS. To successfully create a disk resource, a signature file needs to be created on the root of the shared disk.

These are mandatory arguments:

-name, -n

This argument specifies the resource name to be created.

-type, -t

This argument specifies the resource type. Its value must be **disk**.

path

This argument specifies the mount point of the shared disk.

-mountCommand, -mc

This argument specifies a platform specific command to be invoked when mounting the shared disk. Command "nop" takes no action.

-umountCommand, -umc

This argument specifies a platform specific command to be invoked when unmounting the shared disk. Command "nop" takes no action.

The other non-mandatory options:

-**clusterNodes, -nodes, -cn**

This argument specifies the valid nodes of the cluster across which the resource can failover.

The value is a space or comma-separated subset of (or all) nodes of a cluster. If it is not specified, all the nodes are included.

-resourceParams, -params, -p

This argument specifies a string of comma separated name value pairs to set CRS configuration for a specific resource, for example, **as**=1,**rt**=400.

-policy

This argument specifies what resource parameter values are used. These policies and values are available in the ascrs.properties file.

The valid values are **normal** or **fast**. If it not specified **normal** is assumed. However, the **-resourceParams** values always take precedence over the policy.

EXAMPLE(S)

```
ascrsctl create -n dbhome -t disk -path /cfcdb1
          -mc "/bin/mount /dev/sda1 /cfcdb1" -p fd=30
ascrsctl create -n dbhome -t disk -path /cfcdb1
          -mc "/bin/mount /dev/sda1 /cfcdb1"
          -umc "/bin/umount /dev/sda1"
```

H.5 create/vip

TOPIC

```
create/vip - create vip ASCRS resource
```

COMMAND

```
ascrsctl create -name <string> -type vip -ipAddr <ip> -netmask
          <string> -interface <string> [options]
```

DESCRIPTION

This ASCRS command is used to create (or register) a virtual IP resource in CRS

These are mandatory arguments:

-name, -n

This argument specifies the resource name to be created.

-type, -t

This argument specifies the resource type. Its value must be **vip**.

-ipAddr, -ip

This argument specifies the IP address of the virtual IP or its hostname.

-netmask, -nm

This argument specifies the network mask with the above virtual IP.

-interface, -if

This argument specifies the network interface(s) on which the IP should be enabled.

Value can be one or more interface names such as eth0 or "eth0|eth1".

The other non-mandatory options:

-clusterNodes, -nodes, -cn

This argument specifies the valid nodes of the cluster across which the resource can failover.

The value is a space or comma-separated subset of (or all) nodes of a cluster. If it is not specified, all the nodes are included.

-resourceParams, -params, -p

This argument specifies a string of comma separated name value pairs to set CRS configuration for a specific resource, for example, **as**=1,**rt**=400.

The resource values are listed in Table H–1. Except for **as**, **st** and **ra**, all the other numbers are in seconds.

-policy

This argument specifies what resource parameter values are used. These policies and values are available in the ascrs.properties file.

The valid values are **normal** or **fast**. If it not specified **normal** is assumed. However, the **-resourceParams** values always take precedence over the policy.

EXAMPLE(S)

```
ascrsctl create -n myvip -t vip -ip 192.168.1.10 -nm 255.255.255.0
           -if eth1 -p ci=5
```

H.6 delete

TOPIC

```
delete - delete an ASCRS resource
```

COMMAND

```
ascrsctl delete -name <string> [-type <string>] [-noprompt]
```

DESCRIPTION

This ASCRS command is used to delete a resource created with ASCRS. Once a resource is successfully deleted, the resource is no longer managed by the CRS.

An ASCRS resource can't be deleted if there are still other resource(s) depending it or it is not in offline state.

These are mandatory arguments:

-name, -n

This argument specifies the resource name.

Non-mandatory arguments:

-type, -t

This argument specifies the resource type if the resource name is in short form. It is not needed if the name is in canonical form.

-noprompt, -np

When specified, the user is not prompted for confirmation.

EXAMPLE(S)

```
ascrsctl delete -n mydisk -np
ascrsctl delete -n ora.myvip.cfcvip
```

H.7 start

TOPIC

```
start - start an ASCRS resource
```

COMMAND

```
ascrsctl start -name <string> [-type <string>] [-node <string>]
```

DESCRIPTION

This ASCRS command is used to start a resource already created with ASCRS.

These are mandatory arguments:

-name, -n

This argument specifies the resource name.

Optional arguments:

-type, -t

This argument specifies the resource type if the resource name is in short form. It is not needed if the name is in canonical form.

-node

This argument specifies the cluster node for starting the resource. If it is not specified, the node will be chosen by CRS based on the placement policy of this resource

EXAMPLE(S)

```
ascrsctl start -n mydisk -t disk
ascrsctl start -n ora.myvip.cfcvip -node hostA.mycompany.com
```

H.8 status

TOPIC

status - check the status of ASCRS resources

COMMAND

```
ascrsctl status [-name <string>] [-type <string>] [-long]
```

DESCRIPTION

This ASCRS command is used to check the status of one or all resources created with ASCRS. The status of a resource includes its current running state, its basic CRS profile information and its relationship to the other ASCRS resources.

-name, -n

This argument specifies the resource name. If not specified, check all resources.

-type, -t

This argument specifies the resource type if the resource name is in short form. It is not needed if the name is in canonical form.

-long, -l

if specified, the status information is displayed in a detailed format.

EXAMPLE(S)

```
ascrsctl status
ascrsctl status -name ora.mydisk.cfcdisk
ascrsctl status -l
```

H.9 stop

TOPIC

stop - stop an ASCRS resource

COMMAND

```
ascrsctl stop -name <string> [-type <string>] [-force] [-noprompt]
```

DESCRIPTION

This ASCRS command is used to stop a resource created with ASCRS.

Mandatory arguments:

-name, -n

This argument specifies the resource name.

Non-mandatory arguments:

-type, -t

This argument specifies the resource type if the resource name is in short form. It is not needed if the name is in canonical form.

-force, -f

Shutdown the named resource and take it offline from CRS management. This option guarantees to take offline the CRS monitoring of the resource, but won't guarantee to shutdown the resource if it is already in an unmanageable state.

-noprompt, -np

When specified, the user won't be prompted for confirmation.

EXAMPLE(S)

```
ascrsctl stop -n mydisk -t disk
ascrsctl stop -n ora.myvip.cfcvip -f -np
```

H.10 switch

TOPIC

```
switch - switchover an ASCRS resource to another cluster node
```

COMMAND

```
ascrsctl switch -name <string> [-type <string>]
         [-node <string>]  [-force] [-noprompt]
```

DESCRIPTION

This ASCRS command is used to switchover an online ASCRS resource to an alternative node of the cluster.

Mandatory arguments:

-name, -n

This argument specifies the resource name.

Non-mandatory arguments:

-type, -t

This argument specifies the resource type if the resource name is in short form. It is not needed if the name is in canonical form.

-node

This argument specifies the target cluster node of this resource will switched to. If it is not specified, the target node will be chosen by CRS based on the placement policy of this resource.

-force, -f

force the switchover to happen so that all the dependents and depended resources are also switched over to a new node.

-noprompt, -np

When specified, the user won't be prompted for confirmation.

EXAMPLE(S)

```
ascrsctl switch -n mydisk -t disk hostB.mycompany.com
ascrsctl switch -n ora.myvip.cfcvip -np
```

H.11 update/as

TOPIC

```
update/as - update as ASCRS resource
```

COMMAND

```
ascrsctl update -name <string> [-type as] [-componentHome <string>]
          [-vip <string>] [-disk <string> [<string> ...]]
          [options]
```

DESCRIPTION

This ASCRS command is used to update an as resource created with ASCRS.

Mandatory arguments:

-name, -n

This argument specifies the resource name to be updated.

Non-mandatory arguments:

-type, -t

This argument specifies the resource type and its value must be as. If the resource name is in canonical form, -type option can be omitted.

-componentHome, -ch

This argument specifies the location of the WebLogic domain that contains the targeted administration server.

-vip

This argument specifies the virtual IP resource this component depends upon.

-disk

This argument specifies the shared disk resources this as resource directly depends upon. The value is a space or comma-separated list of disk resource names.

-clusterNodes, -nodes, -cn

This argument specifies the valid nodes of the cluster across which the resource can failover.

The value is a comma or space-separated subset of at least two nodes of the cluster. The special value default will include all the nodes.

-resourceParams, -params, -p

This argument specifies a string of comma separated name value pairs to set CRS configuration for a specific resource, for example, **as**=1, **rt**=400.

The resource values are listed in Table H–1. Except for **as**, **st** and **ra**, all the other numbers are in seconds.

-policy

This argument specifies what resource parameter values are used. These policies and values are available in the ascrs.properties file.

The valid values are normal or fast. If it not specified, no policy will be referenced. The -resourceParams values always take precedence over those from the policy.

EXAMPLE(S)

```
ascrsctl update -n myas -t as -vip newvip -disk instdisk wldisk
ascrsctl update -n ora.myas.cfcas -p st=800,rt=800,pt=800
```

H.12 update/db

TOPIC

```
update/as - update as ASCRS resource
```

COMMAND

```
ascrsctl update -name <string> -type db [-oraHome <string>]
          [-oraSID <string>] [-disk <string> [<string> ...]]
          [-lsnr <string>] [-pfile <string>] [options]
```

DESCRIPTION

This ASCRS command is used to update an ASCRS db resource registered in CRS.

Mandatory arguments:

-name, -n

This argument specifies the resource name to be updated.

Non-mandatory arguments:

-type, -t

This argument specifies the resource type and its value must be **db**. If the resource name is in canonical form, **-type** option can be omitted.

-oraHome, -oh

This argument specifies the database Oracle Home location.

-oraSID, -sid

This argument specifies the Oracle SID name

-disk

This argument specifies the shared disk resources this as resource directly depends upon. The value is a space or comma-separated list of disk resource names.

lsnr

This argument specifies the listener resource.

The other non-mandatory options:

-clusterNodes, -nodes, -cn

This argument specifies the valid nodes of the cluster across which the resource can failover.

The value is a comma or space-separated subset of at least two nodes of the cluster. The special value **default** will include all the nodes.

-resourceParams, -params, -p

This argument specifies a string of comma separated name value pairs to set CRS configuration for a specific resource, for example, **as**=1, **rt**=400.

The resource values are listed in Table H–1. Except for **as**, **st** and **ra**, all the other numbers are in seconds.

-policy

This argument specifies what resource parameter values are used. These policies and values are available in the ascrs.properties file.

The valid values are **normal** or **fast**. If it not specified, no policy will be referenced. The **-resourceParams** values always take precedence over those from the policy.

-pfile, -pf

This argument specifies the database pfile for starting the database. The special value **default** will unset the its current value.

EXAMPLE(S)

```
ascrsctl update -n ora.mydb.cfcdb -disk ohdisk -lsnr newlsnr
          -p st=60,rt=60,pt=60
```

H.13 update/dblsnr

TOPIC

```
update/dblsnr - update dblsnr ASCRS resource
```

COMMAND

```
ascrsctl update -name <string> [-type dblsnr]
         [-listenerName <string>]
         [-listenerOracleHome <string>]
         [-vip <string>] [-disk <string>]
         [-tnsAdmin <string>] [options]
```

DESCRIPTION

This ASCRS command is used to update a dblsnr resource created with ASCRS.

Mandatory arguments:

-name, -n

This argument specifies the resource name to be updated. If it is fully qualified name, **-type** option can be omitted.

Non-mandatory arguments:

-type, -t

If specified, must be **dblsnr**.

-lisnerName, -ln

This argument specifies the database listener name.

-listnerOracleHome, -lsnroh, -loh

This argument specifies the Oracle Home of the database that owns this listener.

-vip

This argument specifies the vip resource this listener runs on.

lsnr

This argument specifies the listener resource.

-clusterNodes, -nodes, -cn

This argument specifies the valid nodes of the cluster across which the resource can failover.

The value is a comma or space-separated subset of at least two nodes of the cluster. The special value **default** will include all the nodes.

-resourceParams, -params, -p

This argument specifies a string of comma separated name value pairs to set CRS configuration for a specific resource, for example, **as**=1, **rt**=400.

The resource values are listed in Table H–1. Except for **as**, **st** and **ra**, all the other numbers are in seconds.

-policy

This argument specifies what resource parameter values are used. These policies and values are available in the ascrs.properties file.

The valid values are **normal** or **fast**. If it not specified, no policy will be referenced. The **-resourceParams** values always take precedence over those from the policy.

-tnsAdmin, -ta

This argument specifies the new location of the listener configuration file. The special value default will unset its current value.

-disk

This argument specifies the disk resource the Oracle Home resides on.

EXAMPLE(S)

```
ascrsctl update -n mydblsnr -t dblsnr -vip newvip
ascrsctl update -n mydblsnr -t dblsnr -disk newdisk -p st=30,pt=40,rt=40
```

H.14 update/disk

TOPIC

```
update/disk - update disk ASCRS resource
```

COMMAND

```
ascrsctl update -name <string> [-type disk] [-path <string>]
          [-mountCommand <string>] [-umountCommand <string>]
          [options]
```

DESCRIPTION

This ASCRS command is used to update a **disk** resource created with ASCRS.

Mandatory arguments:

-name, -n

This argument specifies the resource name to be updated. If it is fully qualified name, **-type** option can be omitted.

Non-mandatory arguments:

-type, -t

If specified, must be **disk**.

-path

This argument specifies the mount point of the shared disk.

-mountCommand, -mc

This argument specifies a platform specific fully qualified command or script name be executed for mounting the shared disk.

-umountCommand, -umc

This argument specifies a platform specific fully qualified command or script name be executed for unmounting the shared disk.

-clusterNodes, -nodes, -cn

This argument specifies the valid nodes of the cluster across which the resource can failover.

The value is a comma or space-separated subset of at least two nodes of the cluster. The special value **default** will include all the nodes.

-resourceParams, -params, -p

This argument specifies a string of comma separated name value pairs to set CRS configuration for a specific resource, for example, **as**=1, **rt**=400.

The resource values are listed in Table H–1. Except for **as**, **st** and **ra**, all the other numbers are in seconds.

-policy

This argument specifies what resource parameter values are used. These policies and values are available in the ascrs.properties file.

The valid values are **normal** or **fast**. If it not specified, no policy will be referenced. The **-resourceParams** values always take precedence over those from the policy.

EXAMPLE(S)

```
ascrsctl update -n mydisk -t disk -umfc "/bin/umount -l /sharedisk"
```

H.15 update/vip

TOPIC

update/vip - update **vip** ASCRS resource

COMMAND

```
ascrsctl update -name <string> [-type vip] [-ipAddr <string>
         [-netmask <string>] [-interface <string>] [options]
```

DESCRIPTION

This ASCRS command is used to update a vip resource created with ASCRS.

Mandatory arguments:

-name, -n

This argument specifies the resource name to be updated. If it is fully qualified name, **-type** option can be omitted.

Non-mandatory arguments:

-type, -t

If specified, must be **vip**.

-ipAddr, -ip

This argument specifies the IP address of the virtual IP or its hostname.

-netmask, -nm

This argument specifies the network mask with the above virtual IP.

-umountCommand, -umc

This argument specifies a platform specific fully qualified command or script name be executed for unmounting the shared disk.

-interface, -if

This argument specifies the network interface on which the IP should be enabled.

Value can be one or more interface names such as eth0 or "eth0|eth1".

-clusterNodes, -nodes, -cn

This argument specifies the valid nodes of the cluster across which the resource can failover.

The value is a comma or space-separated subset of at least two nodes of the cluster. The special value default will include all the nodes.

-resourceParams, -params, -p

This argument specifies a string of comma separated name value pairs to set CRS configuration for a specific resource, for example, **as**=1, **rt**=400.

The resource values are listed in Table H–1. Except for **as**, **st** and **ra**, all the other numbers are in seconds.

-policy

This argument specifies what resource parameter values are used. These policies and values are available in the ascrs.properties file.

The valid values are **normal** or **fast**. If it not specified, no policy will be referenced. The **-resourceParams** values always take precedence over those from the policy.

EXAMPLE(S)

```
ascrsctl update -n ora.myvip.cfcvip -ip 192.168.1.10
ascrsctl update -n ora.myvip.cfcvip -if eth1 -p ci=3
```

Index