

# Oracle Application Server 10g (9.0.4): Manually Managed Cluster

*An Oracle White Paper*  
*June 2004*

# Oracle Application Server 10g (9.0.4): Manually Managed Cluster

Introduction to Oracle Application Server Clustering.....	3
Audience.....	3
Oracle Application Server Clustering Framework.....	3
The Terminology.....	3
The Clustering Framework.....	4
Types of Oracle Application Server Clusters.....	5
Configuring Manaully Managed Oracle Application Server Cluster.....	6
Do I Really Need a Manually Managed Cluster?.....	7
1. Requirement of a Database.....	7
2. Tiered Deployment.....	7
Steps to Configure Manually Managed Cluster.....	9
1. Associate OracleAS Instances Together.....	10
2. Configure OC4J Instances.....	11
3. Configure the J2EE Application for Cluster.....	13
4. Configure OHS for Failover and Load Balancing.....	14
Summary.....	16

## INTRODUCTION TO ORACLE APPLICATION SERVER CLUSTERING

Oracle Application Server 10g consists of many components that can be deployed in distributed topologies. The underlying paradigm used to enable high availability for Oracle Application Server is clustering, which unites various Oracle Application Server components in certain permutations to offer scalable and unified functionality, and redundancy should any of the individual components fail.

Clustering components of a system together allows the components to be viewed functionally as a single entity from the perspective of a client. Several types of clusters can exist with Oracle Application Server components. The procedure to create and configure a manually managed OracleAS Cluster is documented in this paper.

### Audience

This paper targets the engineer or support analyst actually modifying configuration files for different implementation scenarios. For a higher-level view appropriate for managers and architects please see: *Oracle Application Server 10g: High Availability Guide*.

## Oracle Application Server Clustering Framework

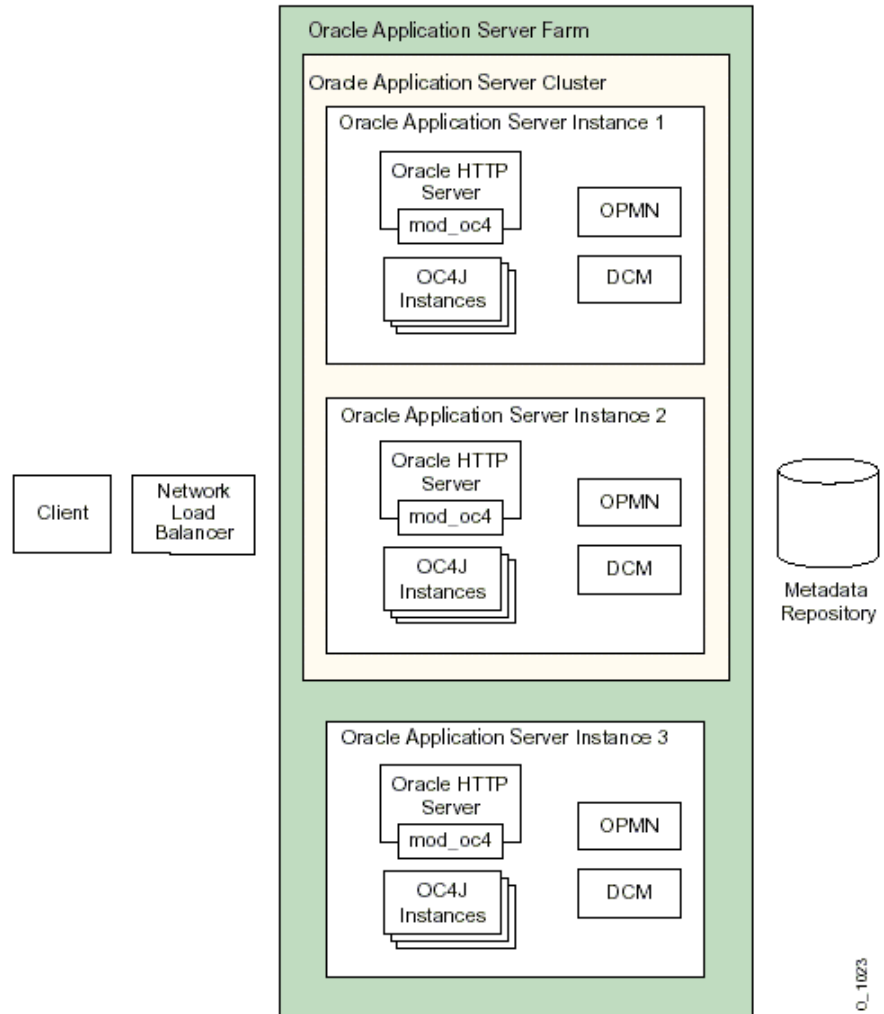
### The Terminology

The following terms are useful to review before discussing Oracle Application Server Clustering Framework:

- **Oracle Application Server Farm:** An Oracle Application Server Farm is a collection of application server instances that share the same Oracle Application Server Infrastructure or that use the same application server instance for their File-Based Repository (FBR) host.
- **Oracle Application Server Instance:** An Oracle Application Server Instance is the set of processes required to run the configured components within an application server installation (within an Oracle home).
- **Component Instance:** Component instances include a single Oracle HTTP Server process or multiple Oracle Application Server Containers for J2EE (OC4J) instances.
- **Oracle Application Server Cluster:** An Oracle Application Server Cluster (OracleAS Cluster) is a collection of application server instances with identical configurations and application deployments.
- **OC4J Island:** An Oracle Application Server Containers for J2EE (OC4J) Island is a group of OC4J processes that replicate session state, for stateful Web applications, among each OC4J process that is part of the island. OC4J processes in an island can be on multiple nodes.

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

Figure 1: OracleAS Cluster Architecture



0\_1023

### The Clustering Framework

The different components that constitute the clustering framework are:

- **Oracle Process Manager and Notification Server (OPMN)** provides process death detection and process restarting in the event that problems are detected for monitored processes, and channels notifications between the different processes.
- **Distributed Configuration Management (DCM)** distributes the configuration information across the components in the cluster, and also saves it to the repository.
- **DCM Metadata Repository** is a vital part of the whole OracleAS Metadata Repository that provides the Management Services. It stores the ultimate golden image (configuration information) for the OracleAS Instance,

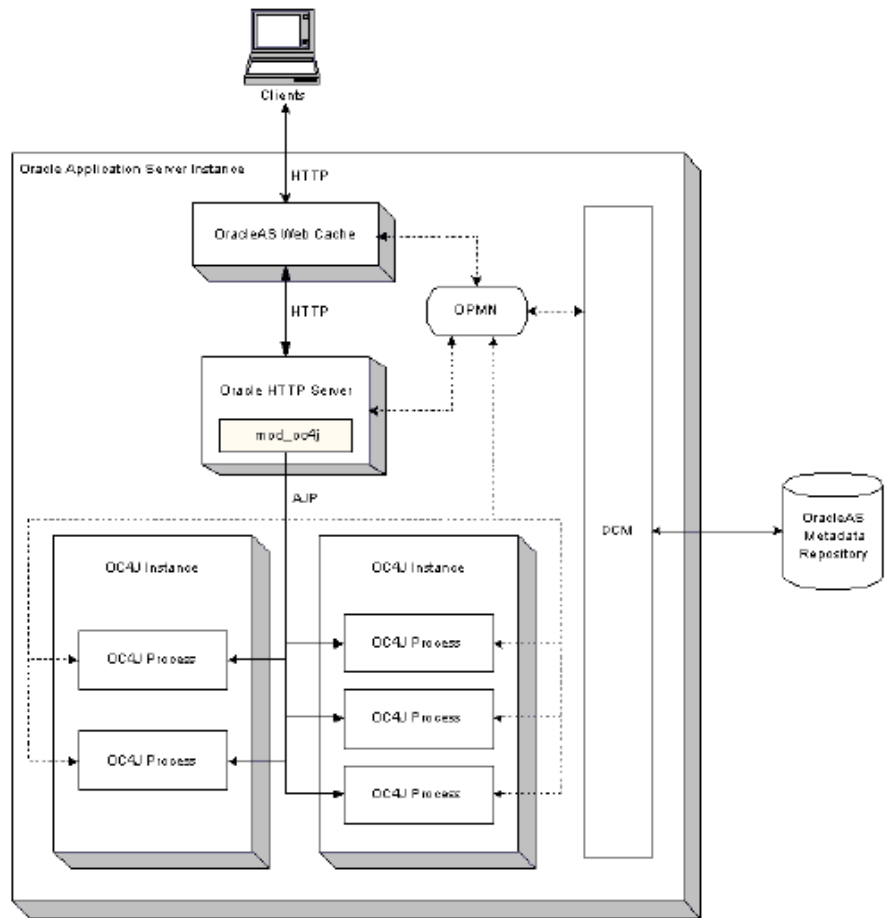
OPMN provides process monitoring i.e. *fault tolerance*, DCM and infrastructure repository provide configuration cloning i.e. *distributed deployment*, and mod\_oc4j provides smart routing i.e. *load balancing*.

OracleAS Cluster and the whole OracleAS Farm. The DCM repository is either file-based or database-based, depending on the OracleAS middle tier installation option.

- **mod\_oc4j**, plugs into Oracle HTTP Server (OHS), provides configurable and intelligent routing for incoming requests, to all OC4J instances using AJP. Requests are routed only to processes and components that mod\_oc4j determines to be alive, through communication with the OPMN.

Figure 2 shows the architecture of an OracleAS Instance, including the features listed above. In this case, DCM metadata repository is stored in the Oracle database as part of OracleAS Metadata Repository.

*Figure 2: OracleAS Instance Architecture*



### Types of Oracle Application Server Clusters

There are two types of Oracle Application Server Clusters:

- **OracleAS Managed Cluster:** OracleAS managed cluster propagates configuration information automatically across all application server instances in the cluster, which simplifies configuration and deployment. All

the instances of an OracleAS managed cluster are part of the same OracleAS Farm, whose repository could either be file-based or database-based.

- **Manually Managed Cluster:** Manually managed cluster rely on the administrators to manually configure each instance within the cluster. With manually configured OracleAS Clusters, it is the administrator's job to make a group of application server instances function as a cluster. It can be cumbersome task to manually maintain the configuration and application deployment information on these OracleAS Clusters. Manually managed clusters provide the following load balancing and high-availability services, without any configuration management:
  - Replicate session state across all clustered instances of the application server.
  - Load-balance requests across all instances of the application server in the cluster.
  - In the event of a node or container failure, transparently fail-over requests to a surviving node or container in the cluster.

With this scenario, each OracleAS Instance must be managed independently. Configuration changes must be manually made to each instance and be identical for each instance. Applications deployed to one instance must be individually deployed to all other instances. There is no Distributed Configuration Management (DCM) functionality available with this configuration.

In essence, manually configured OracleAS Cluster provides scalability and availability, but not manageability. The administrator has the responsibility to synchronize the configuration of the OracleAS instances across the cluster.

---

---

**Note:** This document only covers the steps to create and configure a manually managed cluster. For steps to configure an OracleAS managed cluster, refer *Oracle Application Server 10g: High Availability Guide*.

---

---

## **CONFIGURING MANAULLY MANAGED ORACLE APPLICATION SERVER CLUSTER**

Starting with Oracle Application Server 10g (9.0.4), users have option to use a file-based DCM metadata repository (depending on the OracleAS middle tier installation option). Availability of File-Based Repository (FBR) greatly reduces, if not eliminates, the need to configure a manually managed OracleAS cluster.

## Do I Really Need a Manually Managed Cluster?

In releases prior to Oracle Application Server 10g (9.0.4), the primary reasons why somebody would have wanted to configure a manually managed (or sometimes referred to as “non-managed”) OracleAS Cluster, were the following:

1. Requirement of a Database
2. Tiered Deployment

### 1. Requirement of a Database

One of the principle reasons why people wanted to configure a manually managed cluster was to avoid the need of having a database (for storing metadata repository) to configure an OracleAS managed cluster. But now with the option of FBR, you can configure an OracleAS managed cluster without having to store the metadata repository in a database. Starting with OracleAS 10g (9.0.4), DCM metadata repository can either be file-based (FBR) or database-based (DBR).

### 2. Tiered Deployment

Some people have mistakenly believed that an OracleAS managed cluster can't be configured if they want a “tiered deployment”, meaning that your WebCache, OHS, OC4J and database are running on different tiers. Depending on how you configure each of your tiers, you can still use an OracleAS managed cluster.

Since you cannot selectively install the OracleAS components you want on a particular tier, you would have to either stop or disable the non-required component. For example, a J2EE & WebCache type install will at least install OHS and OC4J, but if you do not want to run OC4J on that tier, you will either have to stop or disable the OC4J component in that OracleAS Instance.

If you decide to stop the non-required components on a tier (as compared to disable them), you can still use an OracleAS managed cluster. Lets look at the following two configurations in more detail:

- a. Non-Required Components can be Stopped
- b. Non-Required Components Need to be Disabled

#### a. Non-Required Components can be Stopped

If you don't have the requirement to disable the non-required components on a particular tier, then you can simply stop them (not running) and put all your tiers in one OracleAS Farm. By doing this, you can create an OracleAS managed cluster and avoid a manually managed cluster. Figure 3 shows this configuration.

#### b. Non-Required Components Need to be Disabled

In cases where it is a business requirement to disable the non-required components on a particular tier, you will have to configure a manually managed cluster. This is because in an OracleAS managed cluster you cannot selectively disable the

components in an OracleAS Instance. Disabling a component in one OracleAS Instance would cause the same component to be disabled in all the OracleAS instances in the OracleAS managed cluster. For example, if you disable OHS in one OracleAS Instance, it will disable OHS in all OracleAS instances that are in the cluster.

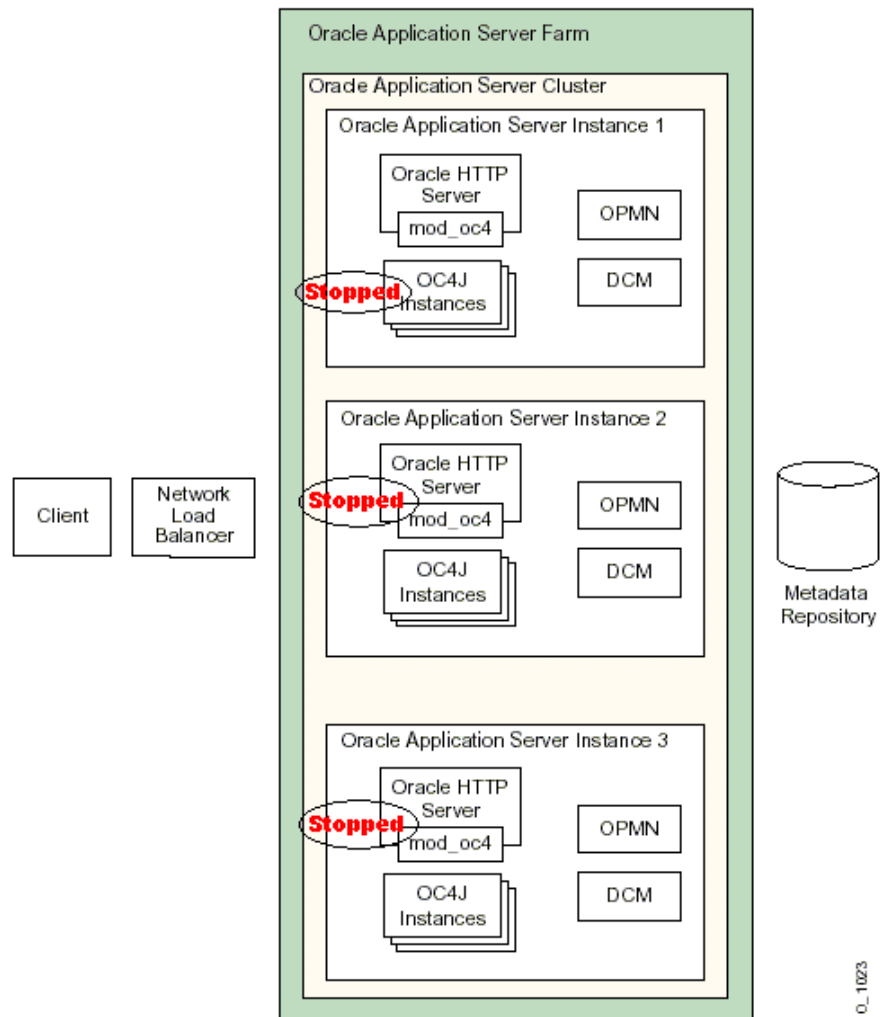
If it is required to disable the non-required components, you cannot put all your tiers in an OracleAS managed cluster. But you can still put all your tiers in an OracleAS Farm and would not have to worry about associating them with each other, when it comes to configuring your manually managed cluster. Figure 4 shows this configuration.

---

**Note:** Several other permutations of the deployment setups are possible, but these two can be used as building blocks for other kinds of setup.

---

*Figure 3: Configuration with Stopped Components*

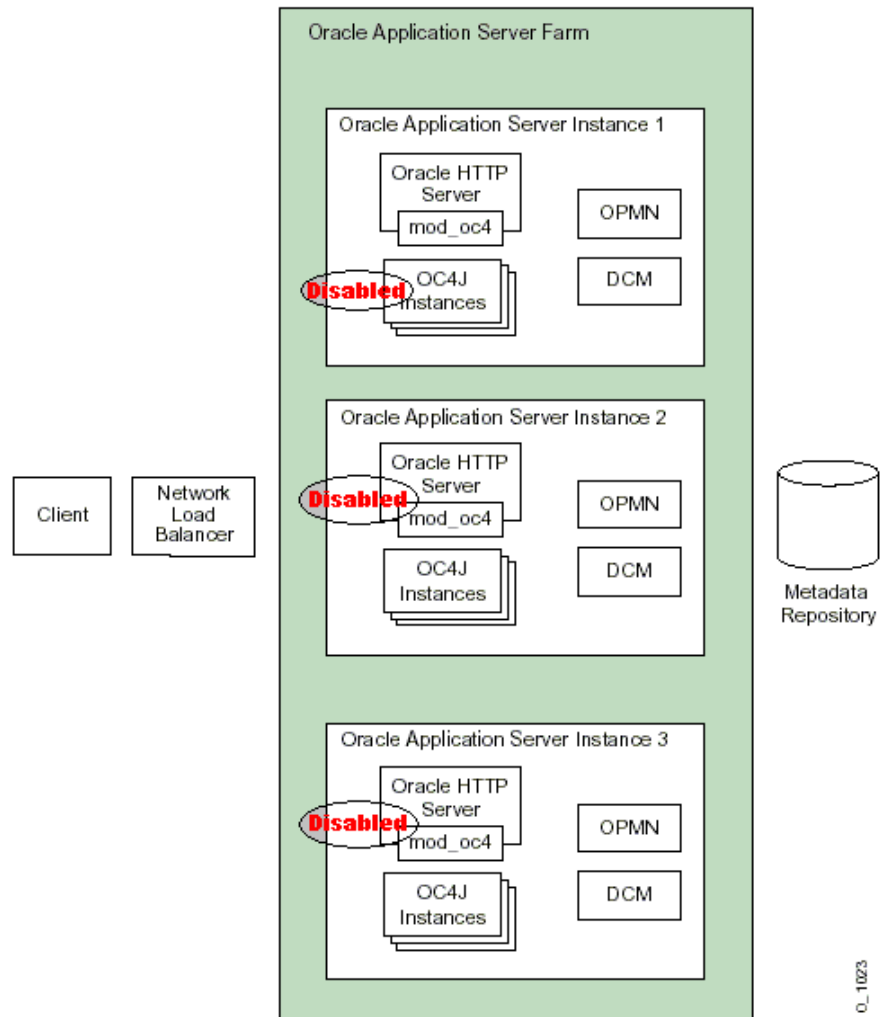


O\_1023

We recommend configuring an OracleAS managed cluster wherever possible. It provides manageability along with scalability and availability; it takes care of synchronization of application server instances configuration across the cluster.

But for the cases where OracleAS managed cluster is either not desirable or not feasible, users still have the option to configure a manually managed cluster. Here we are going to describe the steps in detail to create and configure a manually managed cluster.

**Figure 4: Configuration with Disabled Components**



### Steps to Configure Manually Managed Cluster

Follow these steps to configure a manually managed cluster of OracleAS instances with both Web and EJB state replication enabled. We will also talk about how to configure a J2EE application, so it can be load balanced and failed over among available OracleAS instances.

For the steps below, we are going to assume the following use case:

- A manually managed OracleAS Cluster is required for the purpose of a custom J2EE application, myapp.
- The manually managed cluster will have two standalone (not part of any OracleAS Farm) OracleAS instances, instance names inst1 and inst2.
- The two OracleAS instances (inst1 and inst2) are installed on either two different machines or on one machine but in two separate ORACLE\_HOME.
- J2EE application, myapp, is deployed on an OC4J instance named home, that is not used by any of the OracleAS component internally (e.g. OC4J\_BI\_Forms, OC4J\_Portal etc.).

This section covers the following steps:

1. Associate OracleAS Instances Together
2. Configure OC4J Instances
3. Configure the J2EE Application for Cluster
4. Configure OHS for Failover and Load Balancing

### 1. Associate OracleAS Instances Together

For this use case we have assumed that the two OracleAS instances are standalone, meaning they are not associated with an OracleAS Farm. But if in your case the OracleAS instances (which you want to cluster together) are already associated with an OracleAS Farm (using either file-based or database-based repository) then skip this step and go to step 2.

On both inst1 and inst2, run following command

```
%cd $ORACLE_HOME/dcm/bin
%dcmtl getOPMNPport
```

This command returns the hostname and the ONS remote port for the local application server instance. It retrieves this information from the *ons.conf* file. The output will be of the format – *IPaddress:PortNumber*, e.g.

```
127.2.141.148:6200
```

We will refer the output from inst1 as <IP of inst1:Port of inst1> and output from inst2 as <IP of inst2:Port of inst2>.

On inst1, run following command

```
%cd $ORACLE_HOME/dcm/bin
%dcmtl addOPMNLINK <IP of inst2:Port of inst2>
```

On inst2, run following command

```
%cd $ORACLE_HOME/dcm/bin
```

```
%dcmctl addOPMNLInk <IP of inst1:Port of inst1>
```

The above commands will cause the two OPMN to communicate with each other, as if the two OracleAS instances are in the same OracleAS Farm.

---

---

**Notes:**

- To run addOPMNLInk, all instances must be J2EE and Web Cache instances and the instances must not be part of an OracleAS Farm (associated with a repository); otherwise, the command will fail.
  - If you would like to change the ONS remote port for an instance in a cluster, you must remove the instance from the cluster using removeOPMNLInk, change the remote port, and add it to the cluster again using addOPMNLInk. You must repeat the command in every Oracle home.
  - If you create a cluster and then want to add another instance to the cluster, you must run the command again in all Oracle homes (essentially creating a new cluster with the added instance).
- 
- 

## 2. Configure OC4J Instances

To assure that Oracle Application Server maintains, across OracleAS Cluster, the state of stateful applications you need to configure state replication for the Web and/or EJB applications. Replication properties are at an OC4J instance level and not at the J2EE application level, meaning once enabled, these will be enabled for all the J2EE applications deployed on that OC4J Instance.

If you do not require session state replication or your J2EE application is stateless, you can skip this step and go to step 3.

This section covers the following steps:

- a. Configure State Replication for Web Applications
- b. Configure State Replication for EJB Applications

### a. Configure State Replication for Web Applications

To configure state replication for stateful Web applications, do the following on both instances inst1 and inst2:

- Invoke OracleAS Control and navigate to the Home Page of the "home" OC4J instance.

Session state for Web applications is replicated in OC4J islands with the same name across application boundaries and across the cluster. To assure high availability, with stateful applications, the OC4J island names must be the same in each OC4J instance across the cluster.

- Select the Administration link.
- Select **Replication Properties** in the Instance Properties column.
- Scroll down to the Web Applications section. Figure 5 shows this section.
- Select the **Replicate session state** checkbox, do not enter anything in Multicast Host and Multicast Port.


Optionally, you can provide the multicast host IP address and port number. If you do not provide the host and port for the multicast address, it defaults to host IP address 230.0.0.1 and port number 9127. The host IP address must be between 224.0.0.2 through 239.255.255.255. Do not use the same multicast address for both HTTP and EJB multicast addresses.

*Figure 5: Web State Replication Configuration*

## Replication Properties

Page Refreshed **May 19, 2004 12:41:06 PM** 

### Web Applications

 **TIP** Setting session state replication here will enable session state replication for all web applications. The load-on-startup property will be automatically set to true for all web modules.

**Replicate session state**

Multicast Host (IP)

Multicast Port

### b. Configure State Replication for EJB Applications

To configure state replication for EJB applications, do the following on both instances inst1 and inst2:

- Invoke OracleAS Control and navigate to Home Page of "home" OC4J instance.
- Select the Administration link.
- Select **Replication Properties** in the Instance Properties column.
- In the EJB Applications section, select the **Replicate State** checkbox. Figure 6 shows this section.
- Do not enter anything in Multicast Host and Multicast Port.

Optionally, you can provide the multicast host IP address and port number. If you do not provide the host and port for the multicast address, it defaults to host IP address 230.0.0.1 and port number 23791. The host IP address must be between 224.0.0.2 through 239.255.255.255. Do not use the same multicast address for both HTTP and EJB multicast addresses.

- Provide the username and password, which is used to authenticate itself to

other hosts in the cluster. If the username and password are different for other hosts in the cluster, they will fail to communicate. You can have multiple username and password combinations within a multicast address. Those with the same username/password combinations will be considered a unique cluster.

- Provide RMI Server Host name, this is usually the name of the machine where the OC4J instance is running.
- Click Apply and then OK to confirm.

**Figure 6: EJB State Replication Configuration**

**EJB Applications**

**TIP** EJB applications replicate state between all OC4J processes in the OC4J instance.

Replicate State

Multicast Host (IP)

Multicast Port

Username

Password

RMI Server Host

This is usually the name of the machine where the OC4J instance is running.

**3. Configure the J2EE Application for Cluster**

For the J2EE applications to be cluster aware, you need to make following changes in each of your applications on both instances inst1 and inst2:

- Add <distributed> Tag in web.xml
- Add <cluster-config> Tag in orion-web.xml
- Add replication attribute in orion-ejb-jar.xml

**a. Add <distributed> Tag in web.xml**

Add the <distributed/> tag to all web.xml files. If the Web application is serializable, you must add this tag to the web.xml file.

The following shows an example of this tag added to web.xml:

```
<web-app>
    <distributed/>
    <servlet>
        ...
    </servlet>
```

```
</web-app>
```

#### **b. Add <cluster-config> Tag in orion-web.xml**

Add the <cluster-config/> tag to all *orion-web.xml* files.

The following shows an example of this tag added to *orion-web.xml*:

```
<orion-web-app ...>
    ...
    <cluster-config/>
</orion-web-app>
```

#### **c. Add replication Attribute in orion-ejb-jar.xml**

Modify the *orion-ejb-jar.xml* file to add the state replication configuration for stateful session beans. Since you configure the replication type for the stateful session bean within the bean deployment descriptor, each bean can use a different type of replication (either VMTermination or EndOfCall).

The following shows an example of this attributed added to *orion-ejb-jar.xml*:

```
<session-deployment ... replication="EndOfCall" />
```

### **4. Configure OHS for Failover and Load Balancing**

mod\_oc4j, a module of Oracle HTTP Server (OHS), identifies the requests it needs to act on, determines which OC4J to route those requests to and communicate with a particular process. Every J2EE (web) application, when deployed, needs to be associated with a *root context*. This *root context* (i.e. URL prefix) acts as the identifier of requests that need to be handled by mod\_oc4j. Requests are routed only to OC4J instances and processes that mod\_oc4j determines to be alive, through communication with the OPMN.

#### **mod\_oc4j Configuration**

All mod\_oc4j related configuration information is kept in \$ORACLE\_HOME/Apache/Apache/conf/mod\_oc4j.conf file. OHS uses an Oc4jMount directive to map the root context to the OC4J instance where application was deployed.

It keeps a table of available OC4J instances and load balancing information. We will need to update *mod\_oc4j.conf* routing to send the requests to the OC4J instance in other OracleAS instances (incase the local OC4J instance goes down).

#### **Understanding Request Routing**

Oracle HTTP Server (OHS) uses the Oc4jMount directives, defined in mod\_oc4j.conf file, to route requests containing a particular path to a destination. A destination can be a single OC4J process, or a set of OC4J instances.

The syntax for the Oc4jMount directive is as follows:

```
Oc4jMount path [destination]
```

Where path is the context root, it must be the same as the application context root specified during application deployment and where destination is one of these types:

- ajp13\_dest
- cluster\_dest (this is the default destination type)
- instance\_dest

**For Details Refer:** *Oracle HTTP Server Administrator's Guide 10g (9.0.4)*

A portion of the Oc4jMount section of a default mod\_oc4j.conf file is shown below:

```
Oc4jMount /j2ee/*  
Oc4jMount /webapp home  
Oc4jMount /webapp/* home
```

#### Identifying the Instance Names

Identify the fully qualified names of each OracleAS Instance that will be participating in this configuration.

On both inst1 and inst2, run following command

```
%cd $ORACLE_HOME/dcm/bin  
%dcmctl whichInstance
```

This command returns the name of the local application server instance. We will refer the output from inst1 as <inst1\_name> and output from inst2 as <inst2\_name>.

#### Configure Request Routing

The default mounts points only indicate the local OC4J instance. For this configuration it is necessary to edit the Oc4jMount directives to point to each OracleAS Instance/OC4J instance in the configuration.

The following edits should be done after an application is deployed, not before, since part of the deployment process an Oc4jMount directive is written to the mod\_oc4j.conf file. Do the following on both OracleAS instances inst1 and inst2 (or do it only on one instance, e.g. inst1, if you wish to use OHS of inst1 and would never use OHS of inst2):

- Invoke OracleAS Control and navigate to Home Page of "HTTP\_Server".
- Select the Administration link and then select **Advance Server Properties**.

- Select **mod\_oc4j.conf** in the File Name column.
- In the editor, scroll down and find the Oc4jMount directives for the application you deployed and change these to be as follows:

```
Oc4jMount /myapp instance://<inst1_name>:home,  
<inst2_name>:home
```

```
Oc4jMount /myapp/* instance://<inst1_name>:home,  
<inst2_name>:home
```

This configuration automatically makes these instances (inst1 and inst2) fail-over candidates for each other.

- Click Apply and then OK to confirm. This will require bouncing Oracle HTTP Server (OHS).

## **SUMMARY**

With the introduction of File-based Repository (FBR) in Oracle Application Server 10g (9.0.4), the need to configure a manually managed cluster has greatly reduced. For cases when OracleAS managed cluster is either not desirable or not feasible, this paper described the detailed steps necessary to configure a manually managed cluster.

Here we have described a generic deployment setup that can be used as the building block for other kinds of setup. Several other permutations are possible.



Oracle Application Server 10g (9.0.4): Manually Managed Cluster

June 2004

Author: Shail Goel

Contributing Authors: John Lang, Rick Ehram

Oracle Corporation

World Headquarters

500 Oracle Parkway

Redwood Shores, CA 94065

U.S.A.

Worldwide Inquiries:

Phone: +1.650.506.7000

Fax: +1.650.506.7200

[www.oracle.com](http://www.oracle.com)

Oracle Corporation provides the software  
that powers the internet.

Oracle is a registered trademark of Oracle Corporation. Various  
product and service names referenced herein may be trademarks  
of Oracle Corporation. All other product and service names  
mentioned may be trademarks of their respective owners.

Copyright © 2002 Oracle Corporation

All rights reserved.