4

Конфигурирование и управление кластерами на основе сервера приложений Oracle Application Server 10g

ORACLE

Цели курса

После завершения этого курса Вы должны получить навыки:

- **Описать архитектуру кластеров на основе серверов приложений OAS 10g**
- **Управлять фермами серверов приложений OAS 10g (aka OracleAS Farms)**
- **Создавать кластера на основе серверов приложений OAS 10g**
- **Управлять кластерами серверов приложений OAS 10g (aka OracleAS Clusters)**
- **Разворачивать приложения (to deploy applications) на кластерах серверов OAS 10g**

ORACLE

Lesson Aim

This lesson introdudes the concepts and architecture of Oracle Application Server Farms and Oracle Application Server Clusters. You create an Oracle Application Server Farm by using a file-based repository as well as by using the OracleAS Metadata Repository. To provide a higher level of availability, you learn to create and manage Oracle Application Server Clusters.

Управление кластерами на основе OAS 10g: Новые особенности (vs. Oracle 9iAS R.2)

Следующие компоненты и модули были расширены или изменены:

- **Модуль управления распределенной конфигурацией (Distributed Configuration Management (DCM))**
- **Сервер уведомлений и мониторирования процессов (Oracle Process Monitoring and Notification Server (OPMN))**
- **Динамический модуль OHS для реализации интеграции между OHS и OC4J (mod_oc4j)**

ORACLE

Managing Oracle Application Server Clusters

Distributed Configuration Management (DCM) manages configurations among application server instances, and the `dcmctl` command is to be used for configuration management and application deployment tasks.

Oracle Process Manager and Notification Server (OPMN) has been expanded to provide process management and monitoring for most Oracle Application Server components, and `opmnctl` is the primary command-line tool for starting and stopping the components.

Note: The `dcmctl` command-line utility is the primary tool for configuration management. In Oracle9*i*AS, Release 2, `dcmctl` was also used to start or stop processes. For process management, you must use only `opmnctl`.

For `mod_oc4j` supporting additional load balancing, algorithms are the main enhancements.

Управление распределенной конфигурацией - DCM (Distributed Configuration Management)

- **Distributed Configuration Manager (DCM) дает Вам возможность управлять и синхронизировать конфигурации в кластере серверов приложений.**
- **Каждый инстанс сервера приложений OAS 10g имеет свой собственный DCM.**
- **DCM используется для:**
  - **Создания и удаления кластеров серверов приложений**
  - **Добавления в кластера и удаления инстансов серверов приложений из кластеров**
  - **Обслуживания конфигурационной версионности и синхронизации изменений сквозным образом через весь кластер серверов приложений**
  - **Возможности автоматических реконфигураций в кластере при системных сбоях**
  - **Развертывание J2EE-приложений на инстансах сервера приложений, или на кластерах серверов**

ORACLE

Distributed Configuration Management

Distributed Configuration Management (DCM) enables you to manage configuration by propagating the clusterwide configuration for the application server instances and its components. When you add application server instances to the OracleAS Cluster or modify the configuration, it is the DCM component that automatically replicates the base configuration to all instances in the OracleAS Cluster. DCM is a management feature in each application server instance. DCM is invoked either by Oracle Enterprise Manager or manually by a user through `dcmctl` to do the following:

- Create or remove an OracleAS Cluster
- Add or remove application server instances to or from an OracleAS Cluster
- Synchronize configuration changes across application server instances
- Enable automatic reconfiguration on system failure
- Deploy J2EE applications to an OracleAS instance or cluster

Управление распределенной конфигурацией – DCM: Новые особенности

- **Конфигурация архивирования при использовании `dcmctl createArchive`**
- **Расширенная поддержка управления репозиторием**
- **Управления кластерами**
  - **Управление OracleAS кластерами через file-based repository**
  - **Управление OracleAS кластерами через database repository**
  - **Управление OracleAS кластерами вручную**
- **Error handling and logging**

ORACLE

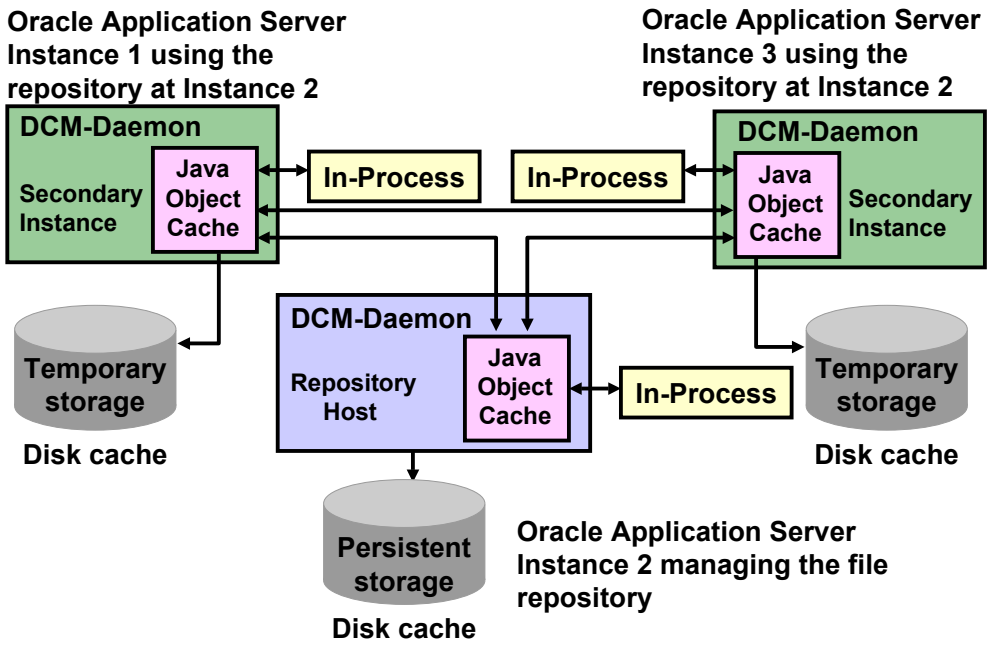
New Features in Distributed Configuration Management

Distributed Configuration Management (DCM) manages configurations among application server instances that are associated with common Infrastructure services (members of an Oracle Application Server Farm). It enables cluster-wide deployment so that you can deploy an application to an entire Oracle Application Server Cluster, or make a single host or instance configuration change that is applicable across all instances in a cluster. The Oracle Enterprise Manager Application Server Control (Application Server Control) uses DCM to make configuration changes, to propagate configuration changes and deployed applications across the members of a cluster, and keep the cluster configuration consistent.

In Oracle Application Server, DCM and the `dcmctl` command provide many new features, including the following:

- **Configuration archiving:** DCM provides a new archiving feature. Using the command `dcmctl createArchive`, you can create an archive of the configuration of an Oracle Application Server instance or cluster, and then apply the archived configuration to the same instance or cluster, or to a different instance or cluster. DCM archiving contains all of the functionality of the deprecated `saveInstance` and `restoreInstance` commands, and much more.

DCM File-Based Repository



DCM File-Based Repository

File-based repository is based on the Java Object Cache. The implementation of this functionality requires a point-to-point messaging system to propagate repository objects between the repository and remote installation. The updates in the repository must be synchronized to maintain consistency. Because file synchronization is very weak in Java, this is best handled by having a single process manage the files in the repository. To allow an installation to run when the repository is unavailable and to enable the repository to migrate in the case of major failures on the repository node, the configuration must be cached at each installation. A distributed invalidation mechanism is required to keep the caches in sync.

The scope of the distributed cache is the OracleAS Farm. As new instances are added to the OracleAS Farm, the scope of the cache is automatically expanded. The existing DCM persistence manager has been extended to retrieve repository objects from the cache. The cache either services the request locally, if it is currently available, or retrieves it from the cache, with direct access to the repository.

Использование архивирования в DCM

The key benefits of DCM archiving are that it:

- **Provides rapid recovery from human error**
- **Allows to revert to a previous archive while the system is online**
- **Does not require a restore from complete backup**

There are two ways to create archives:

- **Using automatic archiving**
  - **Auto Archive Count => 1**
- **Using explicit archiving**
  - **`dcmctl createArchive`**

ORACLE

Using DCM Archiving

The DCM archive feature provides a convenient and easy means of managing “snapshots” of the DCM-managed portions of Oracle Application Server system configuration. You can use archives to restore the state of an Oracle Application Server instance or cluster to a prior state. Archives are useful for staging changes, recovering from errors, and to provision DCM-managed configuration information associated with one Oracle Application Server instance to another.

DCM-managed system configuration includes configuration for an OracleAS Farm, OracleAS Clusters, Oracle HTTP Server, OPMN, OC4J, and Oracle Application Server Java Authentication and Authorization Service. For OC4J, in addition to configuration information related to the container itself, DCM manages all deployed J2EE applications.

Automatic archiving creates an archive automatically before performing a DCM administrative operation. Use the `dcmctl set` command to verify whether auto-archiving is enabled. If the `Auto Archive Count` is nonzero, then auto-archiving is enabled and an archive will be automatically created before issuing a DCM administrative command. The number of automatic archives that are saved is configurable.

Administrators also have the option to explicitly create archives to satisfy the change management or staging policy of the site by using the `dcmctl createArchive` command.

Использование архивирования в DCM

Archives created by using `dcmctl createArchive` have the following properties:

- **Archives contain configuration and application deployment information**
- **Archives can be moved from:**
  - **Database repository to database repository**
  - **File-based repository to file-based repository**
  - **Database repository to file-based repository**
  - **File-based repository to database repository**
- **Can be applied to any compatible instance**
- **Can be exported to a file**

ORACLE

Using DCM Archiving (continued)

Archives created using the command `dcmctl createArchive` have the following properties:

- Archive contains the configuration and J2EE application deployment information associated with an instance or cluster stored in the repository. This prevents a redeployment when you restore from the archive.
- Archives can be moved from database repository to database repository, file-based repository to file-based repository, database repository to file-based repository, or file-based repository to database repository.
- Archives can be applied to any compatible instance:
  - The source of the archive must have the same installed components configured as the destination.
  - Clustering is not required.
  - Archive stores only cluster-type information. Instance-specific information, such as host names, are not saved. This enables the archive to be applied to other instances
- Archived objects can be applied to any compatible instance or cluster in the repository or exported to a file which can be applied to an instance or cluster in another repository, for ease of transitioning from development to production.

For more information, see the *Distributed Configuration Management Reference Guide 10g.*

Команда dcmctl createArchive

- **Use the following syntax to create an archive of a named cluster or instance:**

```
dcmctl createArchive -arch archiveName [cl
myCluster | -i myInstance] [-comment
"myComments"]
```

- **For example, to create an archive of the myInstance instance, enter:**

```
dcmctl createArchive -arch myArchive
-i myInstance -comment "my favourite
configuration"
```

ORACLE

The dcmctl createArchive Command

The syntax to create an archive of a named cluster or instance is as follows:

```
dcmctl createArchive -arch archiveName [-cl myCluster |
-i myInstance] [-comment "myComments"]
```

An archive is created of a named instance or cluster. If you do not specify a cluster or instance, the current instance is archived.

The difference between an archive of a named cluster and an instance archive is as follows:

- Cluster-wide archives that are created with `dcmctl createArchive -cl` contain only cluster-specific information and do not contain any information specific to the instance that the archive is created on.
- Instance-specific archives that are created with `createArchive -i` or with no options contain cluster-specific information as well as instance-specific information. If the archive is applied back to the same instance it was created from, then the instance-specific information will be applied. If the archive is applied to another instance, then the instance-specific information will not be applied.

Экспорт и импорт архивов

The import and export functionality enables you to:

- **Save an archive to the file system**
- **Move an archive from one repository to another, which can be a file-based or database repository**

Use the following `dcmctl` commands to achieve this:

- **`exportArchive`**
- **`importArchive`,**
- **`applyArchiveTo`**

ORACLE

Exporting and Importing Archives

Archives are stored in the repository as either file-based or database-based. If you do not export an archive to the file system, and the repository is destroyed, then the archives saved there are lost. Exporting the archive to a file system provides an extra measure of safety.

You can use `dcmctl exportArchive` and `dcmctl importArchive` on instance or cluster level. You can export an archive from the repository to a file, and then import the file back to the same repository or to a different repository. You can change the name of the archive and associated comments during the import. The original archive name and comments are the defaults.

You can also export from a repository to a file, and import from a file to a repository. The import and export functionality enables an archive to be moved from one repository to another. Use the following `dcmctl` commands to move an archive:

- `exportArchive` exports the named archive from the repository to a JAR file.
- `importArchive` imports an archive file to the current repository.
- `applyArchiveTo` applies an archived configuration to an instance or a cluster.
- `repositoryRelocated` notifies an instance that it is no longer hosting a repository.

Перемещение архивов

On the source instance/cluster, apply:

1. `dcmctl createArchive –arch <archivename> [-cl myCluster | -i myInstance] [-comment "myComments"]`
2. `dcmctl exportArchive –arch <archivename> -f myFile [-comment myComment]`

On the destination instance/cluster, enter:

1. `dcmctl importArchive [-arch <archivename>] -f myFile [-comment "myComments"]`
2. `dcmctl applyArchiveTo –src archiveName [-cl clusterName | -i instanceName]`

ORACLE

Moving Archives

When an archive is created, the configuration and application deployment information associated with the archived object (the Oracle Application Server instance or cluster) is stored in the repository. This archived image can then be applied to any compatible instance or cluster in the repository, or exported to a file that is to be applied to an instance or cluster in another repository. The compatibility of an archive with an instance or cluster is similar to the compatibility of instances that are to be clustered.

Applying archives to instances and clusters is subject to these compatibility rules:

- The source of the archive must have the same installed components configured as the destination. If the destination is a cluster, the archive cannot contain any nonclusterable objects.
- Information specific to an instance, such as host name, can only be applied back to the instance from which it came.
- The configuration from one instance or cluster may be applied directly to another instance or cluster. That is, the apply functionality can take, as a source, an archive of an instance or a cluster.

Перемещение File-Based репозиториев

To move a file-based repository from one OracleAS Farm to another perform the following steps:

1. **On instance A in Farm A:**
   1. **`dcmctl exportRepository -f myFile [-force]`**
   2. **`dcmctl shutdown`**
2. **On instance B in Farm B:**
   1. **`dcmctl shutdown`**
   2. **`dcmctl importRepository -f myFile [-force]`**
3. **On instance A in Farm A:**
   1. **`dcmctl repositoryRelocated`**

ORACLE

Moving File-Based Repositories

It is important to understand that `exportArchive` and `importArchive` are applied at per instance or cluster level, whereas the `exportRepository` and `importRepository` command is used on OracleAS Farm level. The commands can only be used to move a file-based repository from one OracleAS Farm to another OracleAS Farm. There is no direct way to move a database repository to another database repository; you will have to manually dissociate each instances out of one repository (using `dcmctl leaveFarm`) and associate those with another repository (using `dcmctl joinFarm`). In doing so, cluster information would not get exported/imported; you will have to manually set up those in the new repository.

To relocate the file-based repository host from instance A to instance B, perform the following steps:

On instance A, the original file-based repository host,

```
dcmctl shell
dcmctl> exportrepository -f /export/repository_save_file
dcmctl> shutdown
```

If you have more than two instances, then execute the `shutdown` command on all the other instances.

Новые особенности в OPMN

- **Process management**
- **Process monitoring**
- **Improved starting and stopping of components**
- **`opmnctl` is the primary command-line tool for starting and stopping the components**
- **Increased power and flexibility for configuring the Oracle Application Server**
- **Event hooks**
- **Operating system-level statistics**

ORACLE

New Features in OPMN

OPMN has been expanded to provide process management and monitoring for most Oracle Application Server components, and `opmnctl` is the primary command-line tool for starting and stopping the components. In Oracle Application Server 10*g*, you use a single command to start (`opmnctl startall`) or stop (`opmnctl stopall`) all the components in an application server instance in the proper order. The `opmnctl` command even allows to specify start order dependencies such as do not start component B until A has started.

The scope of the `opmnctl` command has expanded—you can start a specified instance in the OracleAS Farm, all instances in the OracleAS Farm, and Oracle Application Server Clusters. You can configure OPMN to execute your own custom event scripts whenever a particular component starts, stops, or crashes. You can select from one or more of the following event types:

- **Prestart:** OPMN runs the pre-start script after any configured dependency checks have been performed and passed, and before the Oracle Application Server component starts. For example, the pre-start script can be used for site-specific initialization of external components.

Quick Reference для opmnctl

Scope	Command	Options
	start	
	startall	
	stopall	
	shutdown	
[<scope>]	startproc	[<attr>=<val>..]
[<scope>]	restartproc	[<attr>=<val>..]
[<scope>]	stopproc	[<attr>=<val>..]
[<scope>]	reload	
[<scope>]	status	[<options>]
	ping	[<max_retry>]
	validate	[<filename>]
	help	
	usage	[<command>]

ORACLE

Quick Reference for opmnctl

The preceding slide lists `opmnctl` commands for quick reference. You can obtain the same output information by executing the `opmnctl help` command. Use:

- `opmnctl start` to start OPMN
- `opmnctl startall` to start OPMN and all managed processes
- `opmnctl stopall` to stop OPMN and all managed processes
- `opmnctl shutdown` to shut down OPMN and all managed processes
- `opmnctl reload` to trigger OPMN to reread the configuration file `opmn.xml`
- `opmnctl startproc` to start OPMN managed processes in the requested scope, which can be one of following: `@instance`, `@cluster`, or `@farm`
- `opmnctl restartproc` to start OPMN managed processes in the requested scope
- `opmnctl stopproc` to stop OPMN managed processes in the requested scope
- `opmnctl status` to get managed process status
- `opmnctl ping` to ping local OPMN
- `opmnctl validate` to validate the given `xml` file
- `opmnctl help` to print brief usage description
- `opmnctl usage` to print detailed usage description

For more information, see the *Oracle Process Manager and Notification Server Administrator's Guide.*

Конфигурационный файл opmn.xml в Oracle9*i*AS, Release 2

```
<ias-instance>
  <process-manager>
    <ohs gid="ohs">
    </ohs>
    <oc4j instanceName="home">
      <island id="default_island" numProcs="1"/>
    </oc4j>
  </process-manager>
</ias-instance>
```

ORACLE

The opmn.xml Configuration File Used in Oracle9*i*AS, Release 2

In Oracle9*i*AS, Release 2 (9.0.2 and 9.0.3), OPMN managed only two application server components—Oracle HTTP Server and OC4J.

For Oracle9*i*AS Release 2 (9.0.2 and 9.0.3), the `opmn.xml` file contained XML element names specific to each Oracle Application Server component; an Oracle HTTP Server element encloses the Oracle HTTP Server configuration, and an OC4J element encloses the OC4J configuration. Because of this requirement, Oracle9*i*AS Release 2 (9.0.2 and 9.0.3) `opmn.xml` lacked flexibility.

Adding management tags of new Oracle9*i*AS components required changes to the XML schema and changes to the configuration processing code to look for the new elements.

In Oracle Application Server 10*g*, all component-specific element names have been removed. In addition, all component-specific management code has been moved into Process Management (PM) modules, which get loaded by OPMN at startup according to what has been specified in the modules section of `opmn.xml`.

Новый конфигурационный файл opmn.xml

```
<opmn>
  <process-manager>
    <ias-instance id="10gTrain1">
      <ias-component id="HTTP_Server">
        <process-type id="HTTP_Server" module-id="OHS">
          <process-set id="HTTP_Server" numprocs="1"/>
        </process-type>
      </ias-component>
      <ias-component id="OC4J">
        <process-type id="home" module-id="OC4J">
          <process-set id="default_island"
    numprocs="1"/>
        </process-type>
      </ias-component>
    </ias-instance>
  </process-manager>
</opmn>
```

ORACLE

The New opmn.xml Configuration File

The opmn.xml file has changed to provide more power and flexibility for configuring the Oracle Application Server. You can edit opmn.xml by using Application Server Control. Click the Process Management link at the bottom of the Oracle Application Server instance Home page. Do not stop the OPMN server after you edit the opmn.xml file. Application Server Control automatically reloads the updated opmn.xml file after you edit the file.

If you are manually editing the opmn.xml file, then run the dcmctl updateConfig command on the command line. dcmctl updateConfig reloads the updated file and updates the configuration repository with the manual changes.

Модуль mod_oc4j

- **The mod_oc4j module is an OHS module.**
- **mod_oc4j routes requests from client for J2EE application to OC4J instances for processing**
- **It contains information on the active/inactive OC4J processes within the OracleAS Instance.**
- **This information is used in routing requests to OC4J processes.**

ORACLE

The mod_oc4j Module

- The mod_oc4j module is an OHS module.
- The module mod_oc4j acts as dispatcher for clients requesting J2EE applications. mod_oc4j routes these requests to OC4J instances for processing.
- It maintains the run-time configuration information on the active and inactive OC4J processes that are members of the OracleAS instance.
- This information is used in routing requests to OC4J processes:
  - This module is responsible for load balancing incoming OC4J requests among all OC4J processes in the OracleAS Cluster.
  - The OHS automatically forwards stateful requests to the original OC4J process, or if that process is dead, to another OC4J process.
  - If the request is stateless, then the request is routed to the OC4J process in the OracleAS Cluster based on the load-balancing algorithm (for example, round robin, random, or metric based) selected in mod_oc4j.conf.

Новые особенности в mod_oc4j

- **Load-balancing algorithm available with Oracle9*i*AS, Release 2:**
  - **Round robin**
- **New load-balancing algorithms:**
  - **Random**
  - **Random with local affinity**
  - **Random with routing weight**
  - **Round robin with local affinity**
  - **Round robin with routing weight**
  - **Metric based**
  - **Metric based with local affinity**

ORACLE

New Features in mod_oc4j

There was only one load-balancing algorithm available with Oracle9*i*AS, Release 2:

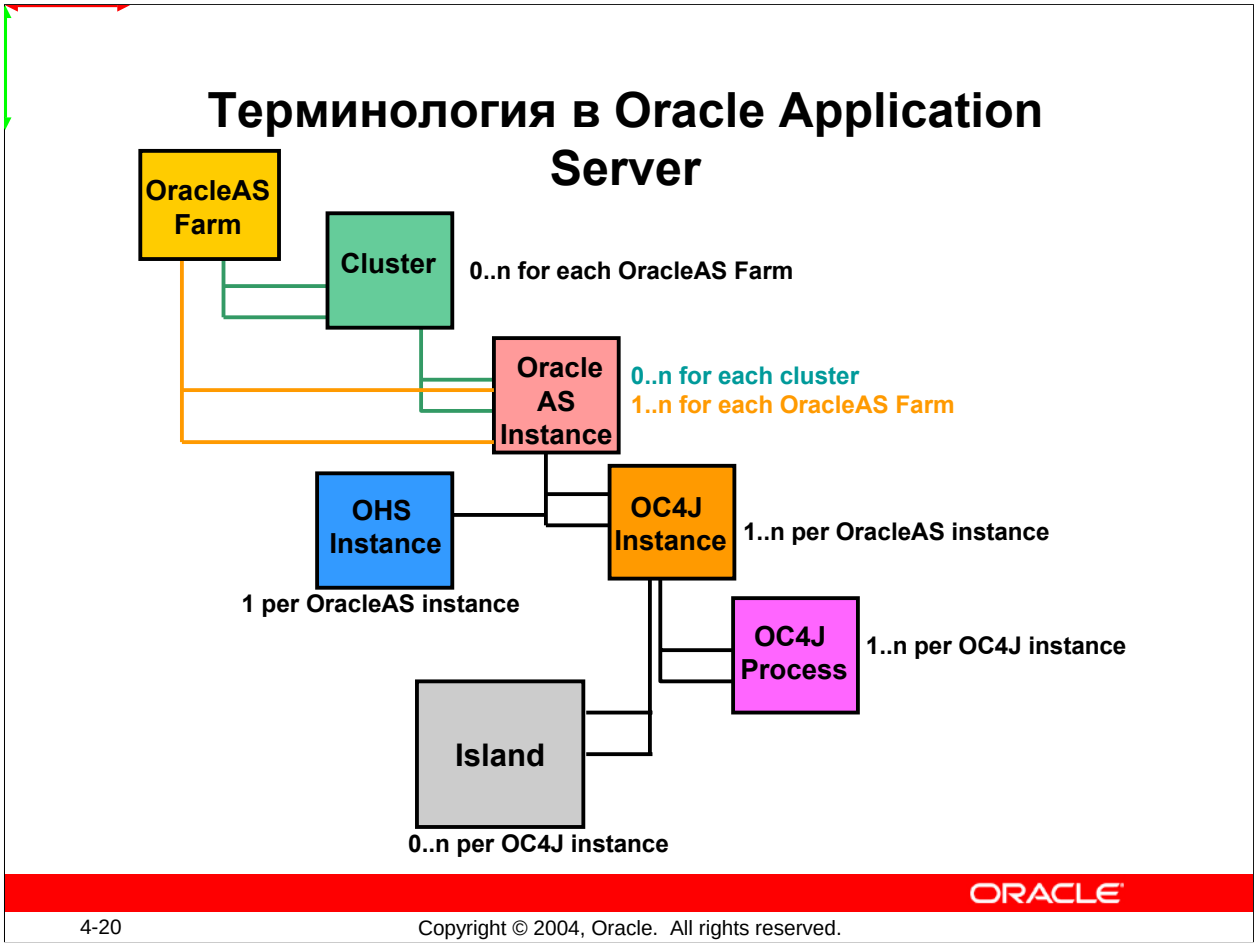
- **Round robin:**
  - On the first request, each `httpd` process picks an OC4J instance at random from an ordered list.
  - OC4Js for subsequent requests to the same `httpd` process are picked from this ordered list in a round robin manner.
  - All OC4J processes (remote and local) are treated equally.

The new load-balancing algorithms are as follows:

- **Random:**
  - On each request, the `httpd` process picks an OC4J instance randomly from an OC4J list.
  - All OC4J processes (remote and local) are treated equally.
- **Random with local affinity:**
  - `mod_oc4j` maintains two OC4J lists—one contains local OC4J processes and the other contains remote processes.
  - On each request, the `httpd` process tries to pick an OC4J instance randomly from the local list and only when there is no local OC4J process available, the `httpd` process picks one from the remote list.

New Features in mod_oc4j (continued)

- **Random with routing weight:**
  - Each machine in a cluster is associated with one integer (>= 1) representing its ability to service requests.
  - On each request, the `httpd` process first uses the routing weight to decide which machine will be selected and then randomly selects an OC4J process from the list on this machine.
- **Round robin with local affinity:**
  - `mod_oc4j` maintains two OC4J lists—one contains local OC4J processes and the other contains remote processes.
  - The `httpd` process always tries to select an OC4J from the local list in a round robin manner.
  - Only when there is no local OC4J process available, the `httpd` process selects an OC4J process from the remote list.
- **Round robin with routing weight:**
  - Each machine in a cluster is associated with one integer (>= 1) representing its ability to service requests.
  - On each request, the `httpd` process first uses routing weight to decide which machine will be selected, and then it selects an OC4J instance in a round robin manner from the OC4J list on this machine.
- **Metric based:**
  - OC4J uses run-time metrics to evaluate its ability to service requests, represents it by an integer, and embeds the number in the registration event.
  - The `httpd` process sends an OC4J process to service requests by using this number.
  - It does not differentiate local OC4Js from remote ones.
- **Metric based with local affinity:**
  - Each OC4J uses run-time metrics to evaluate its ability to service requests, represents it by an integer, and embeds the number in the registration event.
  - On each request, the `httpd` process tries to send an OC4J process from the local OC4Js, according to the routing weight associated with each OC4J process and only when there is no available local OC4Js, it selects one from the remote OC4Js.

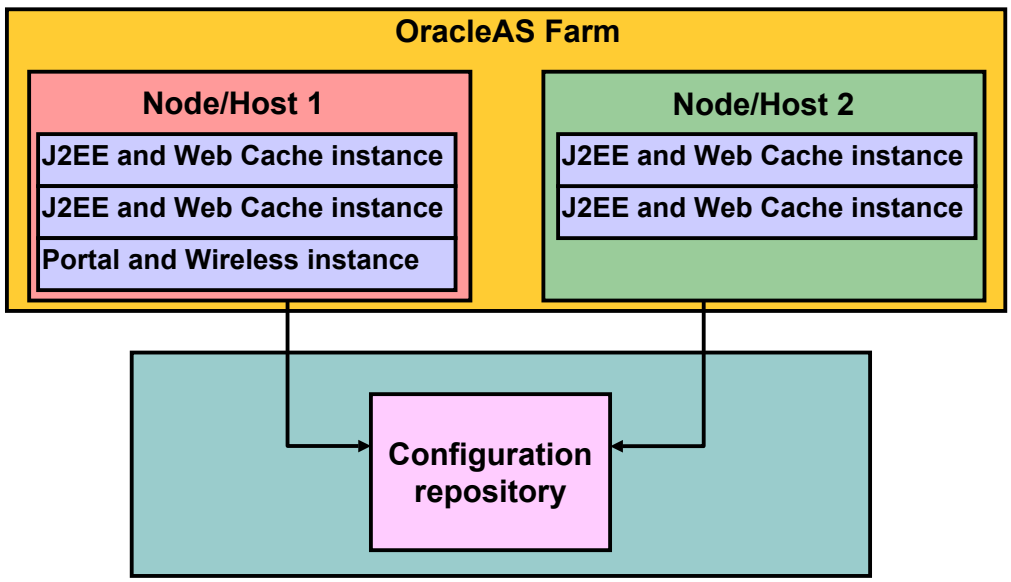


OracleAS Terminology

In the OracleAS clustering paradigm, the following terms are used:

- **Node or host:** A physical machine hosting zero or more OracleAS instances
- **OracleAS instance:** The set of processes required to run the configured components within a single application server installation. There can be only one application server instance per application server installation. The terms “installation” and “instance” must not be used interchangeably. It is important to remember that an installation is the set of files installed into an Oracle Home and an instance is a set of processes associated with those files.
- **OracleAS Farm:** Consists of a collection of OracleAS instances or OracleAS Clusters sharing a single configuration repository and allows for distributed configuration management, distributed monitoring, and remote process control
- **OracleAS cluster:** A collection of OracleAS instances all associated with the same OracleAS Farm, and have an identical configuration
- **OracleAS components:** A configurable piece of the application server such as the Oracle HTTP server, OC4J, or OracleAS Portal

Ферма в OracleAS



OracleAS Farm

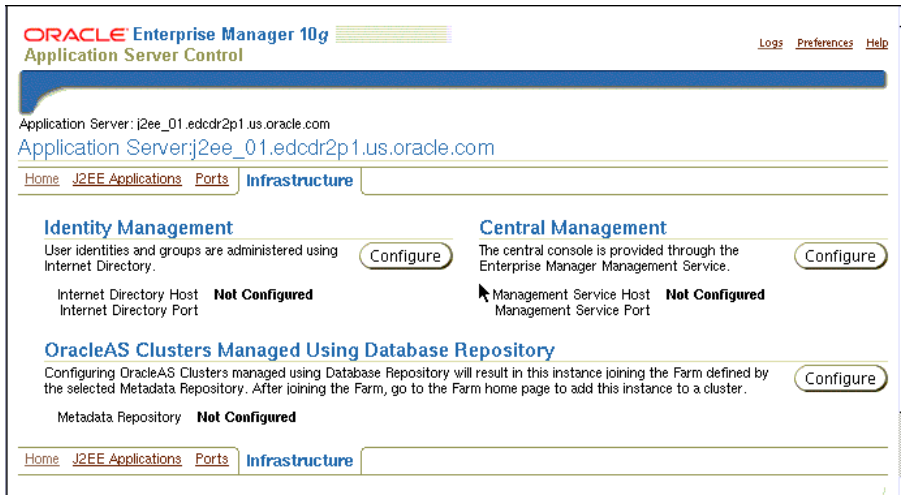
An OracleAS Farm consists of a collection of OracleAS instances being installed by using the same or different operating systems or OracleAS Clusters sharing a single configuration repository. The configuration repository can be file-based or stored in the OracleAS Metadata Repository. The instances that share the configuration repository can be on different physical machines or nodes.

For the OracleAS Farm that uses the file-based repository, the repository is co-located with one of the participating instances. An OracleAS Farm provides for distributed configuration, multiple instance process management, distributed monitoring of instances, and the ability to cluster OHS and OC4J.

An instance that is not associated with an OracleAS Infrastructure is usually referred to as a stand-alone instance.

The configuration repository maintains information about the available instances in the OracleAS Farm. This simplifies the process of creating managed OracleAS Clusters and synchronizing applications and state information across an OracleAS Cluster, because all OracleAS instances share the same configuration repository.

OracleAS Ферма, которая использует OracleAS Metadata Repository



ORACLE

OracleAS Farm That Uses OracleAS Metadata Repository

Before you begin to associate a J2EE and Web Cache instance with an OracleAS Metadata Repository, you should obtain the password for the DCM database user in the OracleAS Metadata Repository.You can use the `oidadmin` utility in OracleAS Metadata Repository and search the DCM user by using the following path:

```
Entry Management > cn=OracleContext > cn=Products > cn=IAS >
cn=IAS Infrastructure Databases > OrclResourceName=DCM.
```

When you click the `OrclResourceName=DCM` entry, you see the password in the `orclpasswordattribute` field.

To associate an instance with an OracleAS Farm that uses an OracleAS Metadata Repository, perform the following steps:

1. Navigate to the Oracle Application Server Instance Home page of the Application Server Control of your middle-tier (J2EE and Web Cache type) instance.
2. Select the Infrastructure link to invoke the Infrastructure page.

OracleAS Farm That Uses OracleAS Metadata Repository (continued)

3. Click Configure under Managed OracleAS Clusters by using Database Repository region to invoke the wizard. In the wizard Location step, provide the following information:
   - User Name: `DCM`
   - Password: DCM password that you obtained from `oidadmin`
   - Host Name and Port: Name of the host of OracleAS Metadata Repository and the Port where the database listener is operational (for example, `edcdr6p1:1521`)
   - Service Name: Fully qualified service name of the database (for example, `asdb.us.oracle.com`)

Initiating an OracleAS Farm That Uses a File-Based Repository

- Test if the instance is a stand-alone instance.
- Join the first OracleAS instance to the OracleAS Farm and initialize the file-based repository.
- Join other OracleAS instances to the OracleAS Farm.

Note: An instance cannot be a member of more than one OracleAS Farm.

ORACLE

Initiating an OracleAS Farm That Uses a File-Based Repository

To create a file-based repository, you must start with a stand-alone application server instance. To verify whether an Oracle Application Server instance is a stand-alone instance, execute the following command:

```
dcmctl whichFarm
```

This command returns the following when an instance is not associated with any OracleAS Farm:

```
Standalone instance
```

If you see that the instance is already associated with an OracleAS Farm, then you can make the instance a stand-alone instance by executing the following command:

```
dcmctl leaveFarm
```

You can use these steps to create an OracleAS Farm and initialize the file-based repository on the repository host instance:

1. To get the repository ID, execute this command on the OracleAS instance that is to be the repository host instance for the file-based repository:

   ```
   dcmctl getRepositoryid
   ```

   The command returns, for example, the following result:

   ```
   edcdr2p1.us.oracle.com:7101
   ```

Initiating an OracleAS Farm That Uses a File-Based Repository (continued)

2. Using the repository ID from step 1, execute the command:
   ```
   dcmctl joinFarm -r <repositoryID>
   ```
   For example, enter:
   ```
   dcmctl joinfarm –r edcdr2p1.us.oracle.com:7101
   ```
   If you use the command `dcmctl joinFarm`, then your instance will use the database repository instead of the file-based repository.
3. To test whether the instance successfully joined the OracleAS Farm, enter the following command:
   ```
   dcmctl whichFarm
   ```
   The result that you get for the example is:
   ```
   Farm Name: .home.oracle.j2ee1.dcm.repository
   Host Instance: j2ee_01.edcdr2p1.us.oracle.com
   Host Name:edcdr2p1.us.oracle.com
   Repository Type: Distributed File Based (host)
   SSL In Use: false
   ```

The `dcmctl joinFarm` command sets up the repository host instance and initializes the Farm using a file-based repository to store the configuration information of the OracleAS Farm.

When you want to join other stand-alone instances to the Farm that you have initialized on the preceding page, then you must execute the following `dcmctl` command in such instances. Get the repository ID from the existing Farm and execute the command:

```
dcmctl joinFarm -r <repositoryID>
```

Обзор OracleAS Кластера

- **Oracle Application Server Cluster (OracleAS Cluster) is a collection of application server instances configured to provide greater scalability and availability.**
- **An OracleAS Cluster can span multiple hosts, distributing application execution over a greater number of CPUs.**
- **OracleAS Clusters enforce homogeneity among member instances.**

ORACLE

OracleAS Clusters Overview

An Oracle Application Server Cluster (OracleAS Cluster) is a set of application server instances configured to perform together to deliver greater scalability and availability than a single instance. You can join instances to OracleAS Clusters when they belong to the same OracleAS Farm. This simplifies the process of creating managed OracleAS Clusters and synchronizing applications and state information, because all OracleAS instances share the same OracleAS Metadata Repository. Changing the configuration or deploying an application to one instance in an OracleAS Cluster triggers the OracleAS Metadata Repository to update the other instances in the OracleAS Cluster with the new information.

An OracleAS Cluster can span multiple hosts, distributing application execution over a greater number of CPUs. A cluster of application server instances can appear and function as a single virtual instance; therefore, any instance in an OracleAS Cluster can serve client requests.

Because instances may run on different hardware configurations and experience varying load-over time, OracleAS instances in an OracleAS Cluster may not always maintain equivalent performance. It is not supported to cluster OracleAS instances that are installed on different operating systems.

Преимущества использования OracleAS кластера

- Scalability
  - Performance enhancements
  - Load sharing
- Manageability
  - Configuring Distributed Environment
  - Automated management
  - Simplifies addition or removal of hosts
- Availability
  - Automatic recovery
  - High Availability or Failover

ORACLE

Benefits of OracleAS Clusters

When applications are deployed in a clustered environment, you have the following benefits:

- **Load sharing:** Applications running in different instances can share the load of processing user requests among themselves. The distribution of traffic is transparent to users.
- **Performance:** When the load can be shared among multiple instances, you can overcome the bottlenecks in performance such as memory and CPU power by running the applications on different systems. Effective load balancing increases system scalability and availability because system resources are used more efficiently.
- **Management:** The OracleAS Clusters can propagate configuration changes made to a single host or instance to other instances across the OracleAS Cluster and maintain uniformity. This eliminates the task of manually updating each instance configuration throughout the OracleAS Cluster. Clustering also reduces human intervention by enabling automatic restart of the failed components.
- **Availability:** Clustering enhances recovery from system failures by simplifying the management of planned and unplanned down times. During system or application maintenance, you can update one host at a time and those changes are propagated clusterwide. Unplanned down times, such as system failure or human error, are minimized through the High Availability infrastructure.

Типы OracleAS кластеров

- **OracleAS Clusters managed by using file-based or database repository:**
  - **Enable configuration to be managed automatically by DCM**
  - **Contain a collection of application server instances with identical application deployments**
  - **Propagate configuration information across all application server instances in the OracleAS Cluster**
  - **Enforce homogeneity between instances that are part of the OracleAS Cluster**
- **OracleAS Clusters managed manually by the administrator**

ORACLE

Types of OracleAS Clusters

There are two types of OracleAS Clusters: OracleAS Clusters managed using a repository (either file-based or database) and OracleAS Clusters managed manually.

- **OracleAS Clusters managed using file-based or database repository:**
  - Contain a collection of application server instances with identical application deployments
  - Propagate configuration information across all application server instances in the OracleAS Cluster, which simplifies configuration and deployment.
  - Enforce homogeneity between instances that are part of the OracleAS Cluster so that the OracleAS Cluster appears and functions as a single instance.
- **OracleAS Clusters managed manually**
  - Rely on the administrators to manually configure each instance within the OracleAS Cluster and synchronize the configuration of the application server instances across the OracleAS Cluster.
  - Provide scalability and availability, but not manageability. This will not be covered in this class, but if you want to set up and configure OracleAS Cluster manually, refer to the collateral material that is available on OTN. (OTN is the Web site for the Oracle Technology Network and is located at http://otn.oracle.com.)

OracleAS Кластера, управляемые с использованием репозитория

- All OracleAS instances participating in an OracleAS Cluster, managed using a repository:
  - Have the same operating system, Oracle Application Server installation version and identical application deployments
  - May differ in instance-specific configurations such as host name, port number, and processes
- OracleAS instances with OHS and OC4J can be part of an OracleAS Cluster.
- Based on the farm that the OracleAS Clusters belongs to, they can be managed using:
  - File-based repository
  - Database (OracleAS Metadata) Repository

ORACLE

OracleAS Clusters Managed Using a Repository

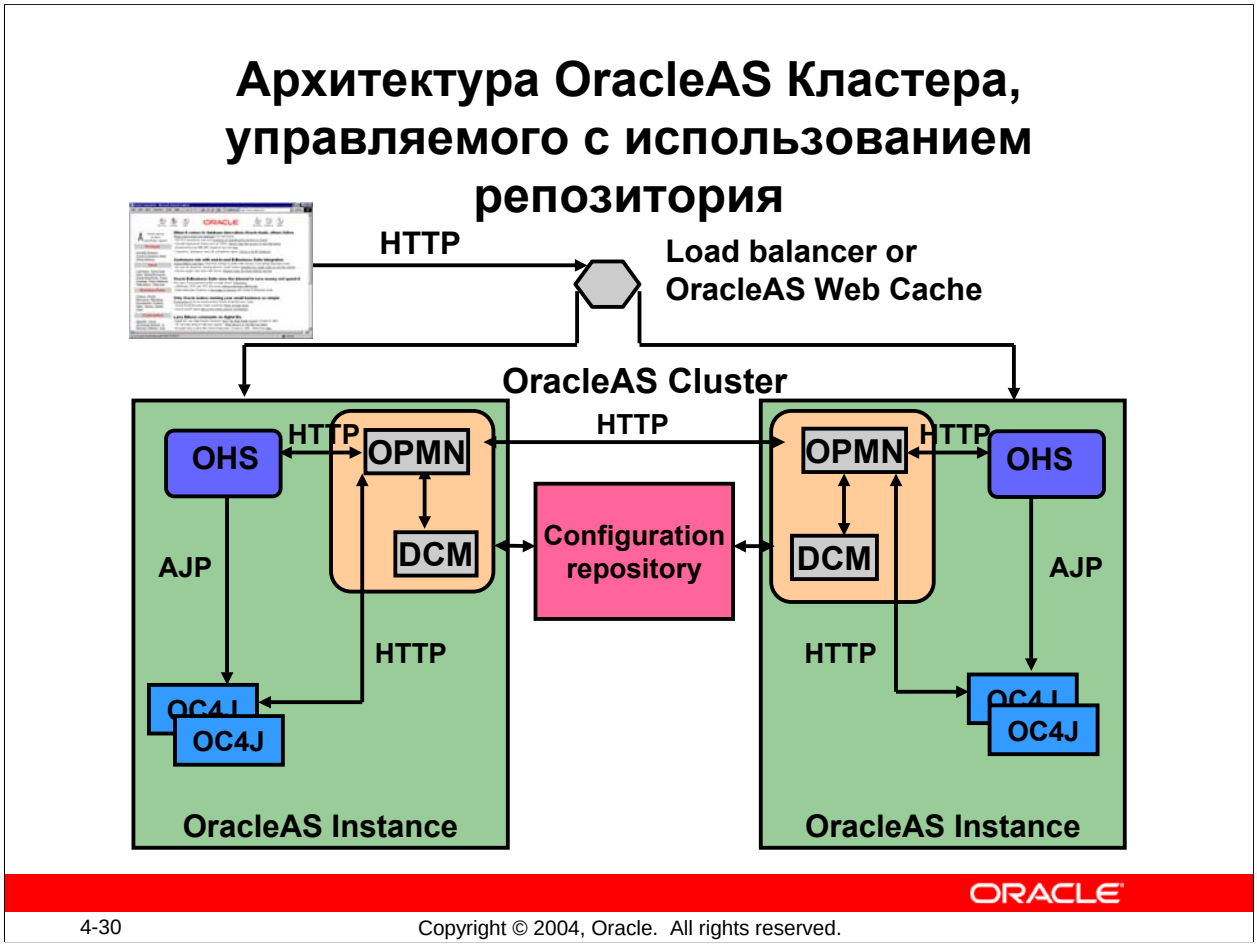
All OracleAS instances participating in an OracleAS Cluster, managed using a repository must have the same operating system, Oracle Application Server installation version and application deployments. They may differ in a few instance-specific configurations such as host name, port number, and number of processes.

An instance can be part of a managed OracleAS Cluster, only if all its configured components can be clustered. Currently, OracleAS instances consisting of Oracle HTTP Server and OC4J can be clustered.

An OracleAS Cluster managed using a repository can include OracleAS instances from multiple nodes and multiple installations. An OracleAS instance can be part of, at the most, one OracleAS Cluster. If an instance is part of an OracleAS Cluster, then all its configured components are implicitly part of that OracleAS Cluster.

Based on the farm that the OracleAS Clusters belong to, they can be managed using:

- **File-based repository:** The configuration in this type of OracleAS Cluster is stored in a file-based repository and one of the instances is designated to act as the repository instance.
- **Database repository:** The clusterwide configuration in this type is stored in the OracleAS Metadata Repository.



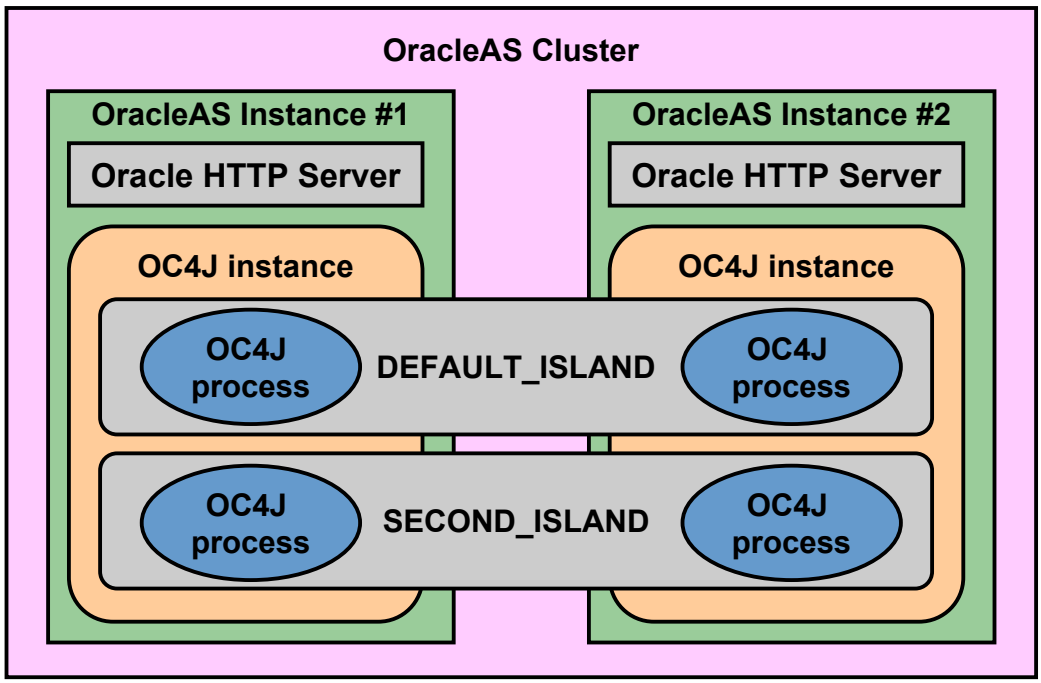
Architecture of an OracleAS Cluster Managed Using a Repository

The different components that constitute the clustering framework include the following:

- Oracle Process Manager and Notification Server (OPMN) that starts, stops, and monitors the processes within an OracleAS instance, and channels events between the instances
- Distributed Configuration Management (DCM) that distributes the configuration information across the components in the OracleAS Cluster, and also saves it to the repository
- Configuration repository that contains all the relevant configuration information. All instances that have their configuration in the same repository are said to be members of an OracleAS Farm. The repository can be a database or just file based.
- The `mod_oc4j` plugs into Oracle HTTP Server (OHS) and routes to all OC4J instances. Its routing table is continuously kept updated by OPMN.

Thus OPMN provides fault tolerance, DCM and the configuration repository provide distributed deployment, and `mod_oc4j` provides load balancing.

OC4J Инстанс, Процесс, Остров



ORACLE

OC4J Instance, Process, and Island

OC4J Instance

- The OC4J instance is the entity to which J2EE applications are deployed and configured. Consider OC4J instance as container of one or more OC4J processes.
- Each OC4J instance can be configured with one or more OC4J processes.
- The configuration of the OC4J instance is valid for all the OC4J processes in that OC4J instance.
- An application deployed to an OC4J instance is deployed to all OC4J processes defined in the instance.
- In the application server instance, you can configure multiple OC4J instances.

OC4J Process

- The OC4J process is the JVM process that executes J2EE applications. You can define one or more OC4J processes within an OC4J instance to provide load balance and failover.
- The state of the application can be replicated across multiple OC4J processes.
- OHS (`mod_oc4j`) provides load balancing for all OC4J processes in the OC4J instance.
- The OPMN component notifies each OHS when a new OC4J process is initiated.

OC4J Instance, Process, and Island (continued)

OC4J Island

An island is a logical grouping of OC4J processes. You can use it to determine which OC4J processes will replicate state.

OC4J processes in an OC4J instance can be partitioned into islands.

The OC4J processes are grouped across application server instances by the name of the island.

Web applications replicate state within the island subgrouping (spanning Oracle AS instances).

Enterprise Java Bean (EJB) applications replicate state between all OC4J processes in the OC4J instance and do not use the island subgrouping.

Конфигурирование и управление OracleAS Кластерами

The following tasks are involved in configuring a managed OracleAS Cluster:

- **Associating the OracleAS instances with an OracleAS Farm**
- **Creating an OracleAS Cluster**
- **Joining the first instance to the OracleAS Cluster**
- **Deploying applications to the OracleAS Cluster**
- **Starting, stopping, joining an instance, and deploying applications are some activities that are similar whether you use file-based repository or database repository.**

ORACLE

Configuring Managed OracleAS Clusters

To participate in a managed OracleAS Cluster, an Oracle Application Server instance must belong to the same OracleAS Farm as the cluster. The method of associating an Oracle Application Server instance to an OracleAS Farm varies depending on the type of managed OracleAS Cluster that you intend to use.

- If you intend to use database repository for managed OracleAS Cluster, then the participating instances should have been associated with an OracleAS Metadata Repository.
- If you intend to use the file-based repository for managed OracleAS Cluster, then the participating instances should have been installed without association to an OracleAS Metadata Repository.

Добавление инстансов в кластер

- **The OracleAS Instance and the OracleAS Cluster must be in the same OracleAS Farm.**
- **The managed OracleAS Clusters can contain instances that have OHS and OC4J components.**
- **The deployed J2EE applications on a joining instance must be distributable.**
- **OracleAS Web Cache does not affect the creation of managed OracleAS Clusters.**
- **The configuration settings for OHS and OC4J of the joining instance will be replaced by that from the OracleAS Cluster.**

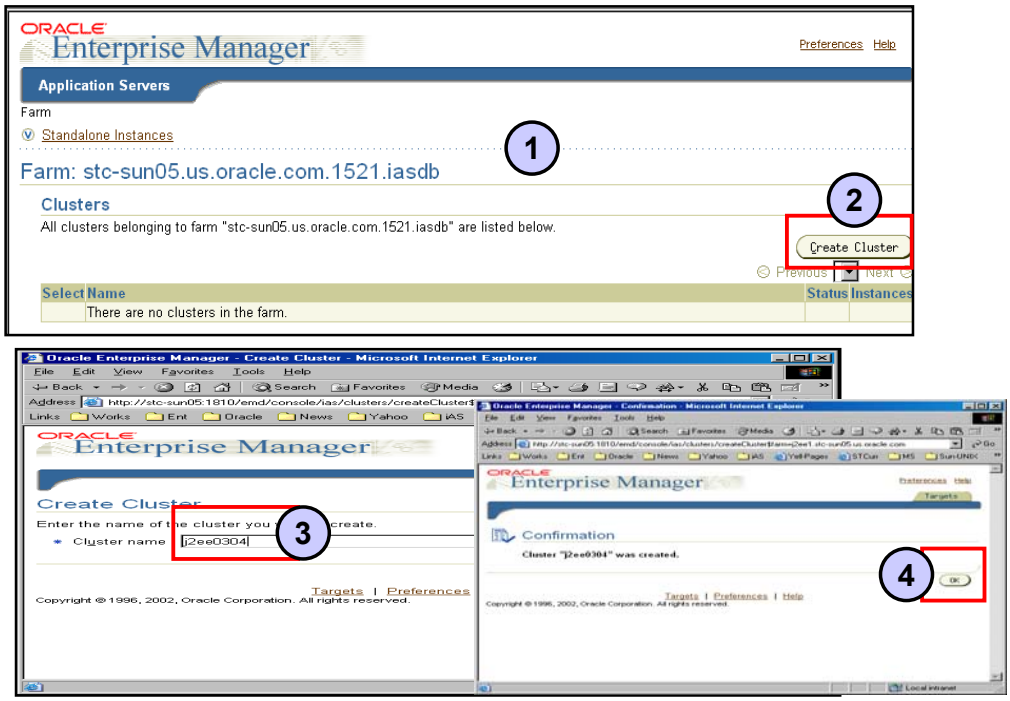
ORACLE

Considerations for Adding an Instance to a Cluster

You should consider the following before adding an instance to the cluster:

- The instance and the cluster must be on the same OracleAS Farm. So, before you add an instance to the cluster, you must associate the instance with the same OracleAS Farm on which the OracleAS Cluster has been created.
- The instance that participates in managed OracleAS Clusters can only be of J2EE and Web Cache installation type. Components, such as Oracle Forms Server or Oracle Reports Server, cannot be clustered.
- Joining an OracleAS instance to an OracleAS Cluster is only possible if all deployed J2EE applications on this instance are distributable.
- OracleAS Web Caches can be a component in the clustered instance. OracleAS Cluster does not enable Web Cache failover, load balance and so on. You must enable Cache Clustering to enable these features on OracleAS Web Cache.
- The first instance to join the cluster creates the configuration information for the OracleAS Cluster. Other instances joining the OracleAS Cluster later inherit the clusterwide properties.

Создание и управление кластерами



ORACLE

Creating and Managing a Cluster

OracleAS instances that are members of an OracleAS Farm can be clustered. You can use Application Server Control or `dcmctl` to create the OracleAS Cluster. To create a cluster by using Application Server Control, perform the following steps:

1. Navigate to the OracleAS Farm Home page.
2. Click Create Cluster, and the Create Cluster page appears.
3. Enter a name for the cluster and click Create. A confirmation message appears.
4. Click OK to return to the Clusters Home page.
   The new cluster is listed in the clusters table.

To add an instance to an OracleAS Cluster, perform the following steps:

1. Navigate to the OracleAS Farm Home page. The target OracleAS Cluster must be listed in the clusters table and the stand-alone instance that you want to add to the OracleAS Cluster must be listed in the Standalone Instances table.
2. Select the instance from the Standalone Instances table and click Join Cluster. The Join Cluster page appears.
3. Select the OracleAS Cluster that contains the OracleAS instance.
4. Click Join. OracleAS adds the instance to the selected OracleAS Cluster and then displays a confirmation page.
5. Click OK to return to the OracleAS Farm Home page.

Запуск и останов OracleAS кластеров

- **When you start an OracleAS Cluster, the OracleAS instances and their component processes are started.**
- **Click the Cluster option button and then click the Start, Restart, or Stop button in the OracleAS Farm Page Cluster region to take action on the cluster.**
- **Drill down to the OracleAS Cluster page, click the Instance option button, and click Start, Restart, or Stop to apply at instance level.**

ORACLE

Starting and Stopping an OracleAS Cluster

OracleAS Clusters make it easy to deploy applications and apply the same configuration changes across all instances of an OracleAS Cluster. In addition, you can start, stop, and restart all the OracleAS instances within an OracleAS Cluster in one simple action by clicking Restart All.

To start, stop, or restart all the OracleAS instances in an OracleAS Cluster, perform the following steps:

1. Navigate to the OracleAS Farm Home page.
2. Select the OracleAS Cluster in the Clusters table.
3. Click Start, Stop, or Restart.

To start, stop, or restart an individual OracleAS instance in an OracleAS Cluster, perform the following steps:

1. Navigate to the OracleAS Farm Home page.
2. Click the name of the OracleAS Cluster in the Clusters table.
3. OracleAS displays the Clusters Home page.
4. Select the instance and click Start, Stop, or Restart.

Конфигурирование Load Balancing для OracleAS Кластеров: Пример 1

Пример 1:

- **System and requirement:**
  - **Several Oracle Application Server instances in the same OracleAS Cluster on different nodes with roughly the same power**
  - **Provision to randomly select OC4J processes**
- **Configuration of `mod_oc4j`**
  - **`Oc4jSelectMethod random:local`**

ORACLE

Configuring Load Balancing for OracleAS Clusters: Example 1

To select a load-balancing algorithm, use the following syntax in the `mod_oc4j.conf` file:

```
Oc4jSelectMethod <method>[:<affinity>]
method -> random | roundrobin | metric
affinity -> local | weighted
```

`Oc4jSelectMethod` selects an OC4J instance for load balancing. This directive is only applicable to the base server for Oracle Application Server 10*g* and an error will be printed at startup, if specified within a `VirtualHost` container.

The default that is used is: `Oc4jSelectMethod roundrobin`

```
Oc4jSelectMethod roundrobin
Oc4jSelectMethod roundrobin:local
Oc4jSelectMethod roundrobin:weighted
Oc4jSelectMethod random
Oc4jSelectMethod random:local
Oc4jSelectMethod random:weighted
Oc4jSelectMethod metric
Oc4jSelectMethod metric:local
Oc4jSelectMethod metric:weighted
```

Configuring Load Balancing for OracleAS Clusters: Example 1 (continued)

`Oc4jRoutingWeight` associates a request routing weight for each machine during load balancing. Weighted routing is a load-balancing strategy that distributes requests according to a predefined value assigned to each machine based on the predicted ability to handle load. To select the routing weight, make these entries in the `mod_oc4j.conf` file:

```
Oc4jRoutingWeight <node_name> <weight>
node_name -> machine name
weight    -> integer >= 1
```

Note: This only takes effect when `Oc4jSelectMethod random:weighted` or `Oc4jSelectMethod roundrobin:weighted` is used.

Example:

There are two hosts in an OracleAS Cluster: `edcdr2p1` and `edcdr6p1`. Each has Oracle HTTP Server and OC4J processes running on them.

```
Oc4jSelectMethod random:local
Oc4jRoutingWeight edcdr2p1 4
Oc4jRoutingWeight edcdr6p1 3
```

`Oc4jRoutingWeight` directives are ignored. `mod_oc4j` on `edcdr2p1` randomly routes all requests to OC4J processes on `edcdr2p1`, and `mod_oc4j` on `edcdr6p1` randomly routes all requests to OC4J processes on `edcdr6p1`.

Конфигурирование Load Balancing для OracleAS Кластеров: Пример 2

Example 2:

- **System and requirement:**
  - **Three Oracle Application Server instances on different machines in the same OracleAS Cluster and their power is: edcdr2p1:edcdr6p1:edcdr1p1=5:4:1**
  - **Provision to select OC4J processes, using a round-robin technique**
- **Configuration**
  - `Oc4jSelectMethod roundrobin:weighted`
  - `Oc4jRoutingWeight edcdr2p1 5`
  - `Oc4jRoutingWeight edcdr6p1 4`
  - `Oc4jRoutingWeight edcdr1p1 1`

ORACLE

Configuring Load Balancing for OracleAS Clusters: Example 2

There are three hosts in an OracleAS Cluster: `edcdr2p1`, `edcdr6p1`, and `edcdr1p1`. Each has Oracle HTTP Server and OC4J processes running on them.

`mod_oc4j` on all the machines route five times the number of requests to OC4J processes running on `edcdr2p1`, four times the number of requests on `edcdr6p1`, and one time the number of requests on `edcdr1p1`in a round robin manner.

Конфигурирование Load Balancing для OracleAS Кластеров: Пример 3

Example3:

- **System and requirement:**
  - **Several Oracle Application Server instances on different nodes in the same OracleAS Cluster**
  - **Provision to select OC4J using run-time metrics**
  - **OC4J to be configured to send run-time metrics**
- **Configuration:**
  - **`Oc4jSelectMethod metric` or**
  - **`Oc4jSelectMethod metric:local`**

ORACLE

Configuring Load Balancing for OracleAS Clusters: Example 3

Metric-based load balancing can be used when `Oc4jSelectMethod metric` or `Oc4jSelectMethod metric:local` is specified in `mod_oc4j.conf` and the `<metric-collector>` element is configured in `server.xml`, which is an OC4J configuration file.

If `<metric-collector>` is not configured, then OC4J does not send any run-time metrics to `mod_oc4j`. `mod_oc4j` and then assumes that the missing value for OC4J is "50."

If `Oc4jSelectMethod random:weighted` is specified in `mod_oc4j.conf` and the `<metric-collector>` element is configured, then OC4J sends run-time metrics to `mod_oc4j`, but `mod_oc4j` ignores it. It uses the "Random using Routing Weight" policy for each machine to perform load balancing.

Заключение

В этом уроке Вы познакомились с тем, как:

- **Описать архитектуру кластера на основе сервера приложений Oracle Application Server 10g**
- **Создать ферму (Farm) для Oracle Application Server 10g**
- **Создать кластер на основе серверов приложений OAS 10g**
- **Простейшим процедурам управления кластером на основе сервера приложений Oracle AS 10g**
- **Развернуть (to deploy) приложения на кластерах OAS 10g**

ORACLE

Информация для контактов по техническим вопросам:

Отдел предпродажного консалтинга по базовым технологиям Oracle CIS: phone: (095)2584180

fax: (095)2584190

Глеб Ладыженский **Технический директор Oracle CIS**
gleb.ladyzhensky@oracle.com

Игорь Лукьянов **Ведущий консультант по OAS продуктам**
igor.lukjanov@oracle.com

ORACLE