



CHAPTER 1

Oracle Application Server 10g Architecture and Administration

2 Oracle Application Server 10g Administration Handbook



This text is intended to serve as a definitive handbook for the installation, administration, and maintenance of Oracle Application Server 10g. It is important to note from the outset that this is not a how-to book on using the program, and there are many other great books on how to apply this powerful software suite.

This book is tightly focused on the administrative responsibilities and maintenance techniques for database administrators using Oracle Application Server 10g.

Because Oracle has consolidated many software products under the umbrella of Application Server 10g, there has been widespread confusion about its scope and functionality. To a web developer, Application Server 10g is Oracle Portal and Oracle Web Cache, while to a developer, it is J2EE and OC4J. However, most users agree that the core functionality of the program is the support for Java development.

In order to properly administer Application Server 10g, you must first understand all of its components and how they fit together. Like any enterprise-wide solution, the components of the program are the result of an evolutionary process, with new subproducts being added as the software evolves. Because Application Server 10g is a broad offering of many tools, your particular functionality may be vastly different depending upon the way you have installed and configured the software. This chapter covers the following topics:

- Overview of the architecture
- Functional components
- Introduction to administration

Let's begin with a review of the Application Server 10g architecture and a look at each functional component.

Architectural Overview

Beginning with their WebServer product in the 1990s, Oracle has continuously improved and streamlined its products into a comprehensive solution for web-based applications.

Application Server 10g is the latest incarnation in a long evolution of application products. Starting in the mid-1990s with Oracle WebServer and Oracle Application Server, Oracle Application Server has evolved into an extremely sophisticated system of interrelated modules, all of which can be configured according to your specifications. There are two ways to view the architecture of Application Server 10g—from a design level and from a functional level. Both are based on a multitiered model.

The Multitiered Model

As Oracle products evolved into a multitiered architecture, we started to see Oracle products reside at several *tiers*, or layers, that represent hardware layers, with each tier made up of one or more servers (Figure 1-1). Because of the flexibility of Application Server 10g, Oracle shops can adopt a two-tiered, three-tiered, or four-tiered model. As a general rule, the larger the system, the more levels and more servers there will be at each level. Application Server 10g components reside at each of these layers in a four-tiered architecture.

Chapter 1: Oracle Application Server 10g Architecture and Administration 3

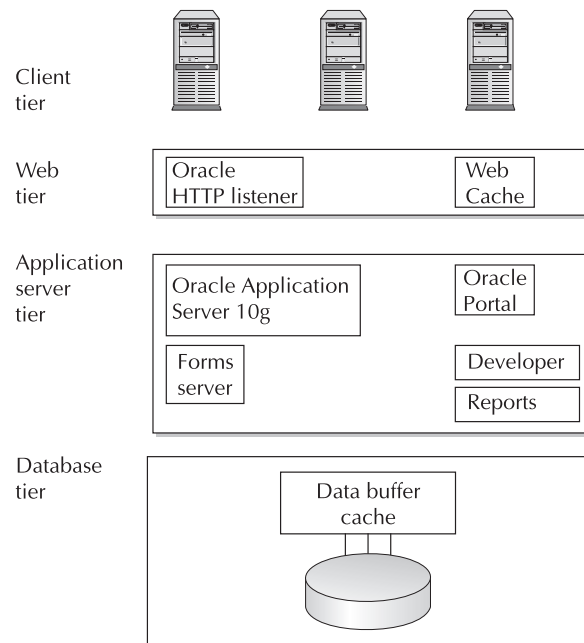


FIGURE 1-1. Oracle application tiers and component products

Application Server 10g components reside at each of these layers:

- **Client tier** Contains the web browsers for end users
- **Web tier** Contains the Oracle HTTP Server and the Web Cache
- **Application server (app server) tier** Contains the core Application Server 10g, plus ancillary products such as Oracle Application Server Portal 10g, Oracle Developer, Oracle Reports, and Oracle Forms Server
- **Database tier** Contains the core Oracle Database, which may be a single instance or many instances defined to a Real Application Cluster (RAC)

Not all shops will use all four tiers. Smaller shops commonly combine tiers into the same level. For example, in a three-tiered architecture, the web tier and app server tiers can be combined. Remember, most large four-tiered systems will have many servers at the web tier, dozens of application servers, and many Oracle instances (using Real Application Clusters) at each node. Also, one or many components may run on any number of servers, and small Oracle shops (or those with huge 16 CPU servers) may combine all three tiers onto a single server. The choice of the number of tiers is directly related to the size of the Oracle 10g implementation and the number of servers that are dedicated to the system.

For small shops, it is common to see a two-tiered data model. Figure 1-2 shows an example of the client tier consisting of all the external PC clients and a combination of the web server tier, the app server tier, and the database tier, all running on a large single server, usually with lots of

4 Oracle Application Server 10g Administration Handbook

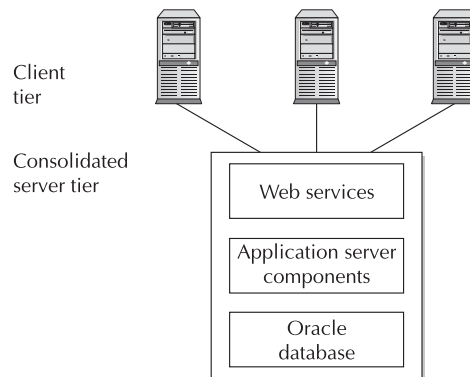


FIGURE 1-2. *Application Server 10g architecture for a two-tiered system*

RAM and multiple CPUs. The benefit of this approach is the shared server resources. The single server can supply additional CPU and RAM processing according to the specific demands of each of the Application Server 10g components. The downside of the two-tiered architecture is the limited flexibility. It is not easy to add hardware resources when you need them.

In medium-sized shops, the three-tiered data model predominates. In this model, shown in Figure 1-3, the client tier is followed by the web server tier and app server on separate servers.

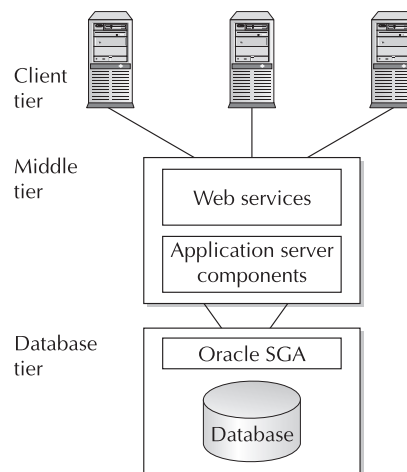


FIGURE 1-3. *Application Server 10g architecture for a three-tiered system*

Chapter 1: Oracle Application Server 10g Architecture and Administration 5

The database tier is also separated onto a different server, thereby providing isolated data resources for the Oracle Database. The three-tiered data model has a few benefits over the two-tiered model. First, increases in processing demands either at the database or the app server level will not affect the performance of the other components within the Application Server 10g architecture. Another benefit is that additional Application Server 10g instances can be created, and additional Oracle System Global Area (SGA) regions can be easily added when processing demands warrant an increase.

Now that you've seen the components of each tier, let's examine how these tiers look when used in a large e-commerce system.

Hardware Architecture of Application Server 10g

Figure 1-4 shows that you can have multiple instances of the components at each tier. In this example, you see two sets of Oracle HTTP Servers (OHS), each listening on a different port for incoming database requests. As requests enter the system, OHS passes them to the least-loaded Application Server 10g instance on the app server tier.

At the app server tier, there may be multiple instances of Application Server 10g and multiple instances of the Oracle Forms Server, Oracle Developer, and Oracle Reports. These multiple instances are normally on separate servers, and this provides administrators with the ability to create an infinitely scalable architecture. Whenever any components at any tier become overwhelmed, administrators can create a new instance on a new server, add the instance into the Application Server 10g architecture using Oracle Universal Installer, and maintain it using the Enterprise Manager.

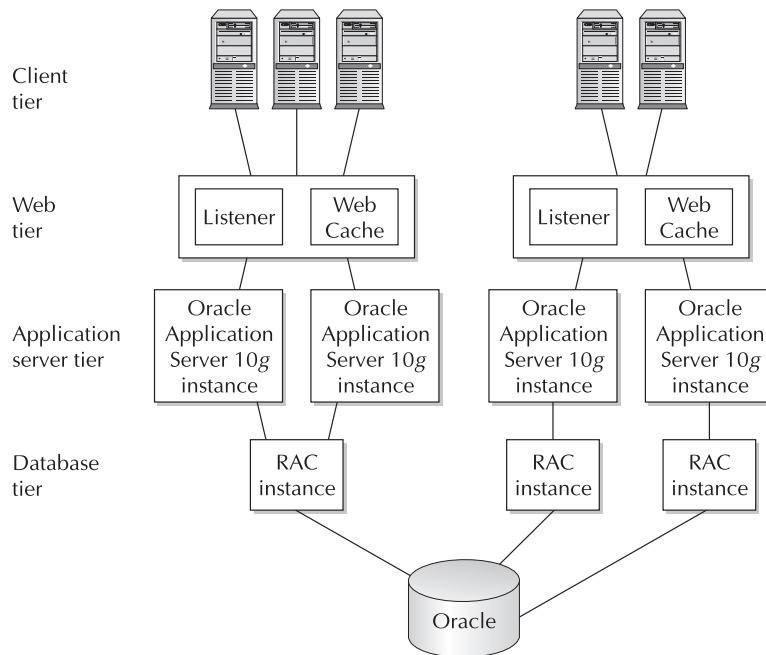


FIGURE 1-4. Application Server 10g tiers and instances

6 Oracle Application Server 10g Administration Handbook

The Application Server 10g instances will connect to the database tier. For very large systems, Real Application Clusters (RAC) provides the ability to have multiple instances of the database, all mapping to a single database. Using the same technique as the other tiers, whenever the existing instances become overloaded, another Application Server 10g instance can be created on a new server, and the server can be added to the architecture.

This ability to scale by adding new instances and servers is a critical aspect of Application Server 10g administration because it is the single most important tool for ensuring that the system always has adequate hardware resources.

Functional Architecture of Application Server 10g

Now let's look at the same architecture from a functional perspective. Figure 1-5 shows the functions of the instances at each level, and this should give you an idea about how the multitiered architecture is used to isolate the logical components of the application.

At the web tier, the main functions are the listener, which listens on a specific port for incoming requests; Web Cache components, which store web page components; and the load-balancing mechanism for ensuring optimal allocation of computing resources to the app server tier. The web tier is managed by the Oracle HTTP Server, which is based on the Apache web server.

The app server tier controls all of the business logic and content assembly. Components such as Oracle Portal are used to define web page components, Oracle Reports defines content specifications, and Oracle Single Sign-On (SSO) controls security for the app server layer. At the database tier are the standard Oracle data management functions for the storage and retrieval of application data. All the components running on the application tier can connect to and retrieve data from the database using any of the available J2EE database connection methods. These are discussed in detail in later chapters. However, Application Server 10g may have its own database if you install the Application Server 10g Infrastructure. With Infrastructure, an Oracle Database

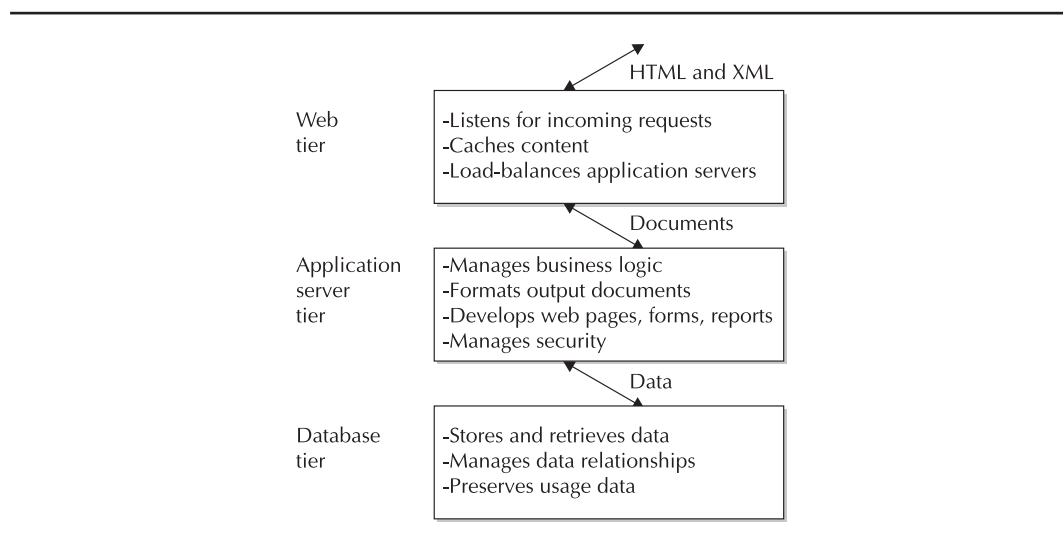


FIGURE 1-5. Application Server 10g functional tiers

Chapter 1: Oracle Application Server 10g Architecture and Administration 7

instance called `iasdb` manages Application Server 10g components and security, and preserves its usage data.

Now that you understand the Application Server 10g architecture from a high level, let's take a closer look at each of these tiers and see how they interact with each other.

Client Tier

The client level consists of either a Java client application or a web browser. Using a web browser as the client allows the entire application to be located on the server. The client always gets the latest version when the application starts. Also, the client can use any web browser from any location, provided that the client can connect to the application server. The client tier can also consist of an application running on the client's desktop (usually presenting a rich user interface) and connecting directly or through HTTP to the application server. This requires that the client have the application installed on the desktop.

Web Tier

The web server layer contains two important components, the Oracle HTTP Server (OHS) and Web Cache component (Figure 1-6). This tier is responsible for managing incoming HTTP requests, caching web messages, and sending XML and HTML back to the client.

Let's take a closer look at the components inside the web tier.

Oracle HTTP Server (OHS)

All Oracle web systems must have enough listener processes so that a single port is not overwhelmed with incoming requests. The Oracle HTTP Server is a component of Application Server 10g that listens on a specific port and forwards J2EE incoming requests through `mod_oc4j` to the least-loaded OC4J container. It is imperative that the web servers have load-balancing intelligence so that a single OC4J container is not overloaded with work. Oracle has addressed

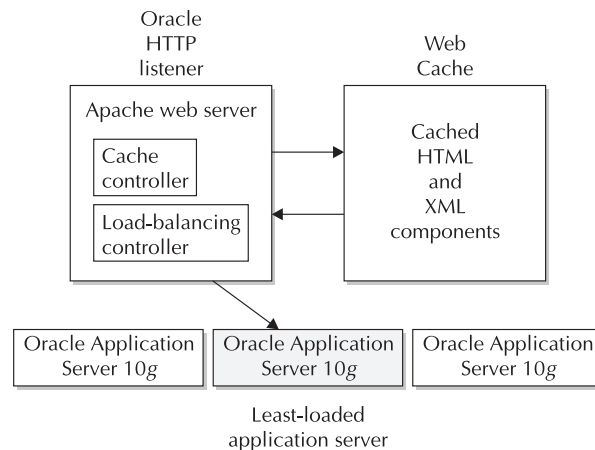


FIGURE 1-6. The Application Server 10g web tier

8 Oracle Application Server 10g Administration Handbook

this issue by incorporating the open source Apache product into the Oracle HTTP Server and providing the load-balancing capability to the mod_oc4j module. This makes customization quite easy.

Another huge benefit of the web listener load balancing is that you can customize the Web Cache to load-balance multiple Oracle HTTP Servers, thereby improving scalability. When the existing app servers become overwhelmed, more app servers can be easily added to the architecture.

It is the job of the web servers to manage the flow of the HTML and XML. On the incoming end, the web server validates and parses incoming XML strings. For outbound transactions, the web server takes data from the application server and creates the outbound HTML pages or XML strings. When an incoming transaction requests services, OHS either serves the HTML page or forwards the transaction to an OC4J container where the appropriate component (JSP engine, servlet, or Enterprise JavaBean) services the request.

Web Cache

The Application Server 10g Web Cache significantly enhances performance by reducing the need to regenerate dynamic or static information. The Web Cache is positioned in front of the HTTP server and stores both static and dynamic web content. It has a number of unique features, including partial-page caching, content-aware web server load balancing, the ability to cluster Web Caches so that multiple caches operate as a single logical cache, and the ability to cache content from third-party servers such as Sun, IBM, BEA, and others. Internal Oracle tests reveal that adding the Application Server 10g Web Cache to a three-tiered application (client, application server, and database tiers) can reduce the load on the database back end by 95 percent. The Web Cache feature has a dramatic impact on the ability of the application to scale to meet growing e-commerce demands.

Now, let's drill-down and examine the central tier, the app server tier.

App Server Tier

The core of Application Server 10g, along with a host of other tools and products, is in the app server tier. The central components are the Application Server 10g instances, and these instances support the Oracle Containers for Java (OC4J). The OC4J container hosts the application's Enterprise JavaBeans, providing security, naming, and connectivity support. In addition to the Application Server 10g instances, the app server tier contains separate components for the following functional areas:

- **Oracle Portal** This component allows for the fast definition and deployment of a dynamically created content-based web site.
- **Oracle Discoverer** This component allows for the easy end-user query implementation.
- **Oracle Forms Server** This component is used to format, deploy, and render end-user presentation pages, based on data in an Oracle Database.
- **Oracle Personalization** This component provides personalized URL referrer tracking and a facility for creating customized web pages, depending upon the user and his or her web page viewing history. The web page history is kept in Oracle Databases.
- **Oracle Wireless** This component allows for communications between Application Server 10g and wireless devices such as PDAs and cell phones. Wireless dynamically reformats information to display correctly on the limited screens of most wireless devices.

Chapter 1: Oracle Application Server 10g Architecture and Administration 9

- **Oracle Reports Server** This component allows for the fast deployment of reports, documents, and spreadsheets, all using data from the Oracle Database.
- **Single Sign-On (SSO)** This is a complete authentication system for identifying users, managing roles and web services, as well as functionality for Java and portal security.
- **Oracle Internet Directory (OID)** This LDAP-compliant directory service provides centralized storage of information about users, applications, and resources in your enterprise.
- **Metadata repository (Infrastructure)** This critical component is sometimes referred to as the Infrastructure. It stores Application Server 10g metadata and allows for a common management interface between multiple instances of Application Server 10g and its other components.
- **Oracle Management Server (OMS)** This component of the Enterprise Manager console allows for managing the Application Server 10g instances, databases, and other applications.
- **Oracle Application Server TopLink** This component provides object persistence for Java information. TopLink contains the mapping interfaces to translate the Java structures into relational tables, thereby making Java persistent across independent executions.

These components are partitioned within the Application Server 10g app server layer, allowing administrators flexibility in the creation of multiple Application Server 10g instances.

Partitioning with Farms and Clusters

Application Server 10g provides several levels of collections within the App Server layer:

- **Instances** An *instance* is defined as a collection of processes required to run a component within an application server instance. An instance is made up of one or more Java containers and the structure needed to support them. The Application Server 10g Infrastructure is an instance with a supporting database to store metadata.
- **Clusters** A *cluster* is an arbitrary collection of instances that are part of the same farm and also share a common configuration and J2EE applications.
- **Farms** A *farm* is a collection of instances and clusters that make up your Application Server 10g system and share a common repository infrastructure.

In sum, a farm is any related group of Application Server 10g instances sharing a repository, while a cluster must share a common definition and J2EE applications (Figure 1-7). Any Application Server 10g architecture may have many farms and many clusters defined within the system.

Application Server 10g Clusters As just defined, a cluster is a collection of Application Server 10g instances that share identical configuration parameters, application deployment schemes, and J2EE applications. Clusters are used to enforce heterogeneity within the Application Server 10g instances. Hence, additions are commonly made to clusters when processing demands require additional Application Server 10g instances in order to manage an increased demand at the application server level. Instances in a cluster are managed by the Application Server 10g Infrastructure, which provides an easy method for creating and maintaining clusters.

10 Oracle Application Server 10g Administration Handbook

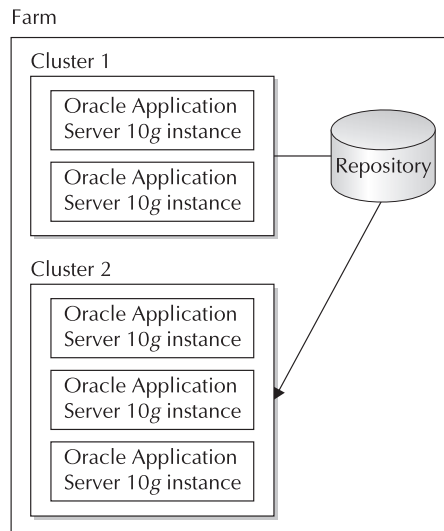


FIGURE 1-7. Application Server 10g farms and clusters

Clusters are used in conjunction with the Web Cache load-balancing algorithms, such that the load balancing at the Web Cache layer monitors all of the Application Server 10g instances in the clusters, and feeds work to the least-loaded Application Server 10g instance. Note that only J2EE and Web Cache components can be clustered, and that clusters must host a common set of J2EE applications.

Application Server 10g Farms There is an important one-to-many relationship between an Application Server 10g instance and a metadata repository. Each Application Server 10g instance may have one, and only one, metadata repository, while each metadata repository may service many Application Server 10g instances.

From the Application Server 10g architecture point of view, a farm is a collection of Application Server 10g instances that all map to the same metadata repository. Because each Application Server 10g instance within a farm must contain the same metadata repository, all instances within the farm must share the same configuration and application membership information.

Database Tier

The standard Application Server 10g relational database (or any other database) resides in the database tier. The function of the database tier is to provide the application with persistent storage. The Application Server 10g also contains a special instance called the Infrastructure that uses a 10g database to store metadata. This database is more correctly in the application server tier since it does not provide persistent storage for the application. The application server provides a method to place

Chapter 1: Oracle Application Server 10g Architecture and Administration 11

the Infrastructure database schema into a database in the database tier; however, best practices will still recommend that the Infrastructure database support only the infrastructure and be separate from the customer database for performance reasons.

The following components have a tight integration with the Oracle Infrastructure database:

- **Oracle Application Server Portal** Web screen component definitions are stored inside the Oracle Infrastructure database.
- **Oracle Reports** Report specifications are stored inside the Infrastructure database.
- **Oracle Application Server Discoverer** Discoverer metadata is stored inside the Infrastructure.
- **Oracle Application Server Personalization** The Infrastructure database is used to store consumer group information and historical page viewing (referrer statistics) information.

Component Overview

Now that you have an overview of the architecture of Application Server 10g, let's continue our tour with a review of the components. Not all shops will have all of these components installed, but Application Server 10g allows for any or all of them to be created inside the architecture.

Application Server Portal

Like the non-Oracle tools such as Dreamweaver and Microsoft FrontPage, Portal allows developers to create and deploy web content. The important difference is that developers can include dynamically created, personalized web pages from multiple data sources using Portlets. The Portal product provides the following features:

- Portal page creation, management, and maintenance
- Assembly of web content from multiple sources using Portlets
- Web page content that contains data retrieved from a database
- Publishing facilities using easy wizards
- Advanced features such as text searching (via Oracle Text) and wireless support via XML and HTML interfaces

These components fit together into an architecture that allows developers to quickly create and deploy web page content. Figure 1-8 depicts a Portal administrator defining the Portlet content and the content for the basic web pages. At run time, Portal users access these definitions to create dynamic publishing content, using the Portlet definitions, the web page definitions, and data from the Oracle Database.

It is beyond the scope of this book to examine all of the content delivery features of Oracle Application Server Portal. For complete information on using Portal, see *Oracle9i Application Server Portal Handbook* by Vandiver and Cox (McGraw-Hill/Osborne, 2001).

12 Oracle Application Server 10g Administration Handbook

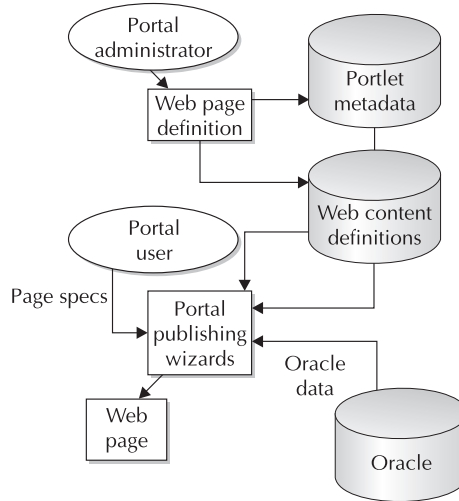


FIGURE 1-8. *The Application Server Portal 10g architecture*

Application Server Discoverer

This component allows for the easy end-user query implementation. In essence, Discoverer is an ad hoc query, reporting, analysis, and web publishing tool. Like Crystal Reports and Business Objects (commercial products that generate SQL queries from a graphical display, making database querying possible for those who do not understand SQL syntax), Discoverer provides a GUI metaphor for the specification of Oracle Database content and display format.

In addition, Discoverer is a business analysis intelligence tool, with interfaces with Oracle Clickstream and the Oracle Database. When using Discoverer, the end user develops workbooks. At a high level, a *workbook* is a bundle of metadata that includes the following components:

- Tables that participate in the query
- Report formatting for the result set
- Calculations to perform on the data

Once defined, these workbooks allow inexperienced end users to easily create ad hoc reports against the Oracle Database using the Discoverer End-User Layer (EUL) graphical user interface. In addition, Discoverer allows end users to view data at several levels, drilling down to more detail or rolling up to summary level.

As you see in Figure 1-9, there are two main phases in Discoverer usage. First, the Discoverer administrator creates the workbooks by specifying the tables, formatting, and computation rules for any given report. Second (the run-time phase), the end user accesses the EUL and creates customized reports using the Discoverer wizards.

Chapter 1: Oracle Application Server 10g Architecture and Administration **13**

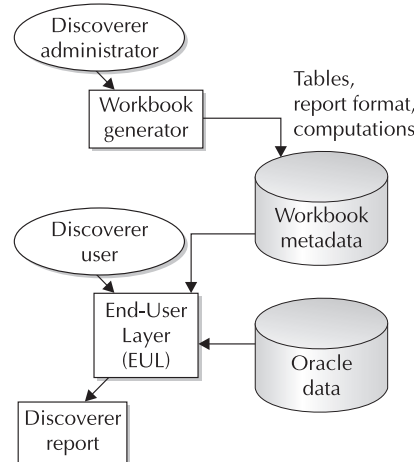


FIGURE 1-9. Application Server Discoverer 10g architecture

The core of administration for Oracle Discoverer is the development and maintenance of the workbooks and metadata objects. For example, each time an end user runs a report, Discoverer refers to the eul_qpp_statistics metadata table in the infrastructure to produce a time estimate for the report. For more details on the administration and use of Discoverer, see *Oracle Discoverer Handbook*, by Armstrong-Smith and Armstrong-Smith (McGraw-Hill/Osborne, 2000).

Oracle Forms Server

An evolution of the Oracle SQL*Forms application development tool, the Oracle Forms Server was originally used to render screen display from Oracle content. Enhanced to provide support for HTML, Oracle Forms Server is now used within Application Server 10g to render web pages that include Oracle Database content.

Because the Forms Server is the main engine for rendering web pages, tuning and administration of this component are critical aspects of overall Application Server 10g administration. We will discuss Oracle Forms Server administration and tuning in more detail in Chapter 10.

Application Server Personalization

Analyzing page viewing behavior and creating custom web page content on a busy e-commerce site constitute a formidable computing challenge. To address these issues, Oracle has developed the Oracle Application Server Personalization 10g and the Oracle Data Mining suite. Personalization is extremely sophisticated and relies on internal data about end-users' web page visits, web page clicks, and referrer statistics. Even more powerful, Personalization allows for the incorporation of external metadata such as customer demographics. It is worthwhile to note that Oracle has several competitors in the web personalization market, notably Blue Martini, Vignette, and Personify.

14 Oracle Application Server 10g Administration Handbook

The goal of Personalization is to accurately identify classes of end users and correlate their behavior with the behavior of other known groups of end users. Using sophisticated multivariate correlation techniques, web page content can be customized according to predictions about each end user's preference for web page content. The nature of this analysis is very resource intensive, and almost all large Application Server 10g shops devote large servers exclusively to developing these predictive recommendations.

IT marketing professionals know that it is critical to get the right products onto a custom web page. To be successful, Application Server 10g must be able to accurately predict a user's propensity to buy a product, based on prior buying and browsing patterns, and buying patterns of like-minded customers (customer profiling). The challenge in developing these predictive models is accurately placing visitors into consumer groups. A *consumer group* is a group of customers with similar demographics and buying patterns.

Figure 1-10 shows the process of analyzing demographic information to place visitors into consumer groups. A visitor can be placed into a consumer group in two ways:

- Demographic category (collected from personal information)
- Pattern of page views (collected from referrer URLs)

Once consumer groups have been defined in Personalization, you next start a data mining procedure to correlate the patterns of each consumer group with specific products. The customized HTML personalization is based on data from three sources:

- **Known consumer group data** These groups consist of predetermined summaries of consumer group characteristics.

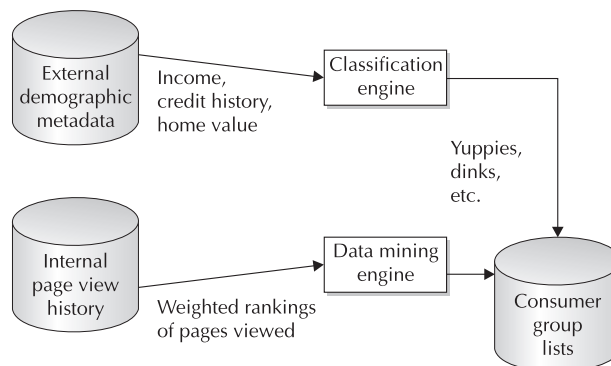


FIGURE 1-10. Architecture of Personalization

Chapter 1: Oracle Application Server 10g Architecture and Administration 15

- **Weighted rankings of pages viewed** This is a measure of the popularity of product pages according to each consumer group.
- **Historical data** This is historical sales data, correlated by consumer group.

Personalization uses these sophisticated consumer group and data mining component mechanisms to create the web content (Figure 1-11). The administration of Personalization is simplified by using the Personalization GUI, and the Oracle documentation has an excellent discussion of Personalization administration.

Oracle Application Server Wireless

This component allows for wireless communications between remote wireless servers and the Application Server 10g architecture. The core of Oracle Application Server Wireless 10g is the use of XML communications. Wireless transforms XML data into whatever markup language is used by the wireless system, including standard HTML, Wireless Markup Language (WML), and other special wireless markups such as VoiceXML and HDML. This allows the application to generate one set of XML data that is reformatted for the presentation device, be it a cell phone, personal digital assistant (PDA), or pager.

Wireless communications with Oracle is becoming commonplace because of the ubiquitous nature of Internet service providers creating wireless infrastructures (mostly in major cities). Within these areas, Wireless can be used to establish direct communications with Application Server 10g using a standard J2EE and XML communications model. Wireless has the benefit of isolating the database communications from the complexity of the wireless protocol by encapsulating the communications into a separate, intermediate layer.

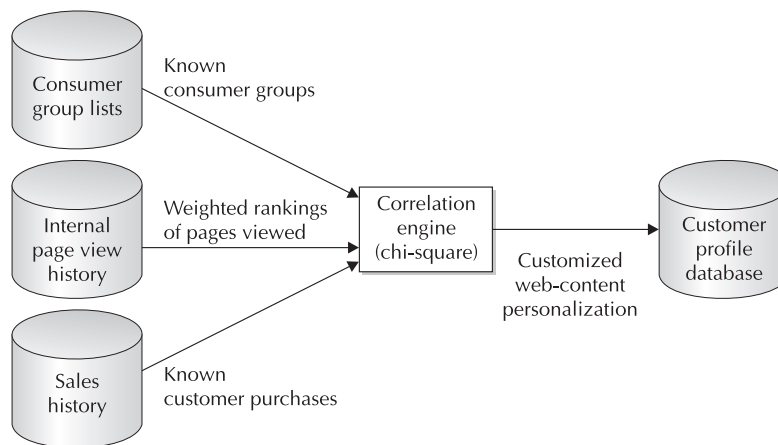


FIGURE 1-11. The Oracle Application Server Personalization 10g engine at run time

16 Oracle Application Server 10g Administration Handbook

This is one of the most exciting components of Application Server 10g because it holds the promise for wireless voice communications with Oracle Databases. This technology could bring millions of end users into far closer contact with their valuable data.

Oracle Reports Server

This component allows for fast deployment of reports, documents, and spreadsheets, all using data from the Oracle Database. To achieve this function, the Oracle Reports Server must interface with an Application Server 10g instance (and Portal) to manage the incoming report requests and send the completed reports back to the requesting user. To understand the Oracle Reports Server, let's take a simple example and follow the report steps (Figure 1-12).

- **Invocation** The Reports Server is invoked via the end user entering a URL (or clicking a link on a web page).
- **Routing** The Application Server 10g instance intercepts the HTML or XML request and directs the request to the Reports CGI (or Reports servlets).
- **Request validation** Oracle Reports then parses the HTML or XML request and determines the report and the security rules for the report. If secure, Oracle Reports sends an HTML page back to the end user to accept a username and password.
- **Execution** The verified request is then queued for execution in the Reports Server. Note that you can configure multiple run-time engines for each Reports Server.

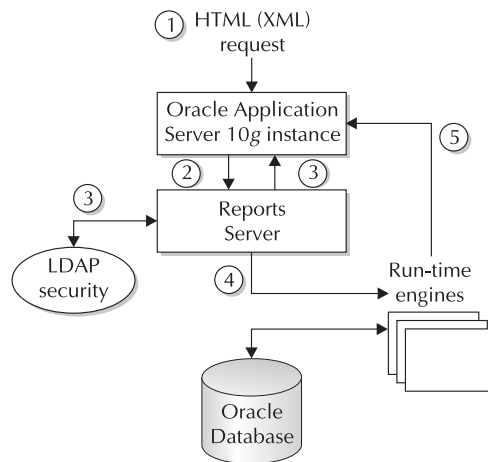


FIGURE 1-12. The Application Server Reports Server 10g at run time

Chapter 1: Oracle Application Server 10g Architecture and Administration 17

- **Formatting** Upon completion of the execution, the Reports Server formats the output as HTML and forwards the completed report to the Application Server 10g instance.
- **Delivery** The Application Server 10g instance then completes the request by sending the completed report to the end user.

Single Sign-On (SSO)

With Single Sign-On, a client can sign onto the application once and be automatically authenticated for other components within the application server, as well as to external applications if properly set up. SSO provides a central authentication repository rather than having a separate authentication for each application on the server. SSO uses the Infrastructure instance to validate users as they move from application to application without forcing them to reauthenticate.

The SSO component interacts with the Oracle HTTP Server (OHS) and allows the formatting of Single Sign-On information as an open source Apache header. Note that SSO only functions within the domain of your Oracle system. Many distributed e-commerce systems communicate with third-party portals, and SSO cannot be extended to service these external clients. For example, an Oracle e-commerce site might need to process a payment request with Cybercash, and Cybercash would require its own independent SSO mechanism. Hence, many Application Server 10g administrators must develop XML Data Type Definition (DTD) protocols for communication with external third-party systems.

We will discuss SSO and other components of Application Server 10g security in great detail in Chapter 12.

Oracle Internet Directory

The Oracle Internet Directory (OID) is a Lightweight Directory Access Protocol (LDAP) directory service that provides centralized storage of information about users, applications, and resources in your enterprise. Coupled with SSO, OID allows end users to sign on one time and use their predefined OID credential (set up by the DBA). This credential defines those components of Application Server 10g with which the end user is allowed to interface.

Because it is LDAP-compliant, OID can be viewed as a simple lookup mechanism for web services. For example, LDAP entries can be used instead of entries in the traditional `tnsnames.ora` file, thereby allowing connectivity for clients anywhere on your network. This technique has replaced the obsolete Oracle*Names tool as a method for defining services for Oracle.

In sum, OID is an easy-to-configure tool for defining end-user access with Application Server 10g. Because it is tightly coupled with SSO and advanced security, OID is a critical component of Oracle security management. OID is managed with a GUI called Oracle Directory Manager (ODM). We will discuss this tool for managing data access rules in great detail in Chapter 12, along with other security topics.

Metadata Repository (Infrastructure)

The metadata repository is a critical component of Application Server 10g because it allows for a common management interface between multiple instances of Application Server 10g and the other components. The metadata repository is commonly referred to as the Infrastructure, which

18 Oracle Application Server 10g Administration Handbook

is common to all Application Server 10g farms and components that share a common definition. We will discuss the Infrastructure in great detail in Chapter 3.

Oracle Management Server (OMS)

With the Oracle Management Server, administrators can include the Application Server in a centrally managed configuration using Oracle Enterprise Manager (OEM--a separate product). OMS is a component of the Oracle Enterprise Manager console, used to manage Application Server 10g instances, databases, and other components. The foremost feature of OMS is its ability to store OEM data inside the metadata repository. This storage ability of OMS allows administrators to share server configuration information, scheduled events and jobs, and notifications of failures. To start OMS, you use the emctl command and issue the emctl start oms command to start the web servers and OMS processes.

Because OMS is the "glue" that binds all of the Application Server 10g components together, we will be visiting OMS functionality throughout this book. OMS provides the important functions of user administration, and manages the flow of information between the OEM console and all managed nodes. OEM allows for any server to become a managed node by installing an Oracle intelligent agent (OIA), thereby making it accessible with the central administrative GUI. An OIA is a daemon process that interfaces with the database and operating system on each server within each Application Server 10g farm. The intelligent agent performs localized execution of tasks as directed by the OMS, and for Oracle servers, the OIA performs time-based database monitoring. The concept of managed nodes adds power to OEM, allowing the Application Server 10g DBA to quickly apply configuration changes to many server components.

TopLink

For Java developers, TopLink provides a mechanism for making Java objects persistent across sessions. In object-oriented (OO) languages such as Java, C#, or C++, objects can be instantiated and destroyed according to the needs of the program.

The problem is that OO languages like Java create objects in the RAM heap, and upon termination of the program, all of the program's objects are destroyed. Oracle Application Server TopLink 10g is a persistence framework that enables object persistence by supplying routines that can be invoked to store Java objects in relational database tables (in any relational database that supports JDBC). In addition, TopLink provides a GUI tool, the Mapping Workbench, that greatly simplifies the task of mapping Java objects and their attributes to database tables. TopLink also provides powerful features like a query framework, object-level transaction support, relationship mappings, object caching, and much more. Prior to TopLink, the programmer would have to write custom JDBC code to store and retrieve the Java object's attributes to/from a relational table. This is not only extremely time consuming and error prone but also difficult to change. TopLink is built on top of JDBC but does not require developers to use JDBC (or even SQL!). TopLink supports all J2EE-compliant application servers, and can be used to store object data from standard Java objects, as well as entity beans. Please refer to the Oracle Application Server TopLink 10g documentation for more information.

Oracle Application Server 10g Administration

Unlike an Oracle Database, which has only a few administrative interfaces (OEM, SQL*Plus), Application Server 10g has many administrative utilities. To make matters even more challenging,

Chapter 1: Oracle Application Server 10g Architecture and Administration 19

these administrative tools are often tightly coupled, as in the case of the Application Server 10g Web Cache administration pages and the Oracle HTTP Server administration pages. Both of these administrative interfaces are separate, yet they are closely intertwined in the architecture.

Application Server 10g provides two methods for administration, the command-line interface and Oracle Enterprise Manager. This book will show both methods, and the choice of Application Server 10g administration methods is largely up to the individual.

We will start with a review of each administrative component and then look at using OEM and the command-line interfaces within each component.

Administrative Component Overview

As an Application Server 10g administrator, it is your job to become intimate with all of the management components. Of course, your shop may not have some of the optional components, such as Single Sign-On, but it is imperative that you understand the administrative components and how they fit together. This section will review the general administration tools, Web Cache administration tools, and application layer administration tools.

General Administration Components

Here are the main administrative interfaces for the Application Server 10g Infrastructure:

- **LDAP Server (OID)** This is the Oracle Internet Directory (OID) component of Application Server 10g. The LDAP server is the foundation of the automated provisioning methodology, and administrators must manage the LDAP repository (the directory) to maintain user-access privileges.
- **Single Sign-On (SSO)** The SSO component provides for centralized management among all of the Application Server 10g components. Large shops may have dozens of components, and SSO allows for easy password management and access control.
- **Metadata repository (isadb)** The isadb is an Oracle database that stores configuration information and metadata. This includes data used by LDAP, OMS, and SSO.
- **Mod_osso module** This provides communication between the SSO-enabled login server and the Oracle HTTP Server (OHS) listener. The mod_osso module is controlled by editing the mod_osso.conf file.

Web Administration Components

From the top down, the web server component (Web Cache and OHS) is one of the most important components of Application Server 10g, and one where tuning is vital. For details on Application Server 10g Web Cache and Oracle HTTP Servers, see Chapter 10.

- **Oracle HTTP Server (OHS)** This is the HTTP listener software that intercepts incoming requests and routes them to the appropriate Application Server 10g component. Upon completion of the transaction, the OHS sends the completed HTML or XML back to the originating IP address.
- **Web Cache** This component is associated with an OMS instance and server to provide RAM caching for images (GIFs and JPEGs), as well as page content. The Web Cache and the OHS are closely coupled, and tuning the Web Cache is addressed in Chapter 10.

20 Oracle Application Server 10g Administration Handbook

Application Management Components

Moving down the Application Server 10g hierarchy, you next see the administrative tools for application development, primarily for Java applications. Administrators must use these interfaces to ensure optimal configuration of their systems.

- **J2EE server (OC4J)** This component allows you to deploy and manage Java-based applications. Administrators must configure the J2EE server to ensure proper communications between OC4J and other Application Server 10g components.
- **Oracle Process Manager and Notification (OPMN)** OC4J is started and managed with OPMN, which is also responsible for monitoring all Application Server 10g processes and propagating configuration changes across clusters.
- **Distributed Configuration Manager (DCM)** DCM is a handy command-line utility that can be used instead of the GUI for starting and stopping Application Server 10g services.

Command-Line Interfaces or OEM?

As we have already noted, administrators have two choices for managing Application Server 10g—the OEM console GUI or the command-line interfaces. Using the OEM console, the GUI will issue the appropriate commands without your having to memorize the syntax. On the other hand, many experienced Application Server 10g administrators find that the command-line interface offers a full range of administration commands.

Of course, some tasks must be done from the command-line interfaces. For example, you cannot use OEM until the OMS is started, so you must issue the `emctl start oms` command before you can use OEM. Internally, it makes no difference whether you use OEM or a command-line utility to manage Application Server 10g. This is because the OEM console uses DCM (the `dcmctl` utility) to make configuration changes, and to propagate configuration changes and deployed applications across the cluster.



CAUTION

If you use the Infrastructure and you manually edit the configuration files, you may introduce corruption into the Infrastructure. This is true for both v9.0.2 and v9.0.3. Be sure to shut down the Enterprise Manager web site (`emctl stop`) before using `dcmctl` to change configuration. If/when both are used "at the same time," there is a strong possibility that the Infrastructure data may become corrupted, and you may have to reinstall Application Server 10g. The `dcmctl-updateConfig` command can be used to notify the environment that config files were updated so that the changes are properly picked up. This requirement will be referenced throughout the book.

Let's start with a quick tour of OEM for Application Server 10g and then review the command-line interfaces.

Managing Application Server 10g with Enterprise Manager

The Enterprise Manager console is the central management component for Application Server 10g. From a page of the EM central console, you can manage most of the areas of Application Server 10g on multiple servers.

If you have installed the Infrastructure component of Application Server 10g (iasdb database repository), then the default EM console page will be the EM Farm page. The Farm page is the highest level of the EM pages and is used to administer all instances within your Application Server 10g configuration. Let's quickly review the component hierarchy from the bottom up:

- **Instances** Each J2EE app server or infrastructure is called an instance (not to be confused with an Oracle Database instance, which is quite different).
- **Clusters** A cluster is an arbitrary collection of instances.
- **Farms** A farm is a collection of instances and clusters that make up your Application Server 10g system and share a common repository database (iasdb).

Each farm may have many clusters, each cluster may have many instances, and each instance may have many Application Server 10g components. It is your job as the administrator to configure your components, instances, clusters, and farms according to the processing requirements of your application.

The purpose of the EM Farm page is to serve as the master console and display summary information about each instance and cluster within the farm (Figure 1-13).

Remember, each instance within the Farm page is an independent J2EE app server or an infrastructure, and the Farm page allows you to drill-down and see the details for each instance using the EM Instance Manager page. Using the EM Farm page, you can also define new clusters and assign instances to clusters. In Application Server 10g parlance, a "standalone" instance is a J2EE app server, belonging to a farm, which has not been assigned to a cluster. A cluster is two or more identically configured app server instances. To assign an instance to a cluster, you simply choose it and click the Join Cluster button.

Next, let's step down one level and look at what you see when you drill-down into an instance and see the EM Instance Manager page.

Instance Manager Home Page

Instance Manager is somewhat of a misnomer. To Oracle DBAs, an instance is a running Oracle Database, while to Application Server 10g administrators, an instance refers to a J2EE app server or an infrastructure within their Application Server 10g farm.

For each instance, the Instance Manager page allows you to manage all of the Application Server 10g components. When you select a server from the OEM Farm page, you get the Instance Manager page with details on all components on that server (Figure 1-14). The top of the page displays the host name and status of the server. You also see CPU and RAM memory usage for the server. The bottom half of the page shows all of the Application Server 10g components on that server.

For each component, you see the current status (up or down), the start time for the component, and the relative amount of CPU and RAM usage for each component. By selecting a component and clicking the management buttons, you can start, stop, enable, disable, and configure each component on the instance. Let's take a look at the links on this page.

22 Oracle Application Server 10g Administration Handbook

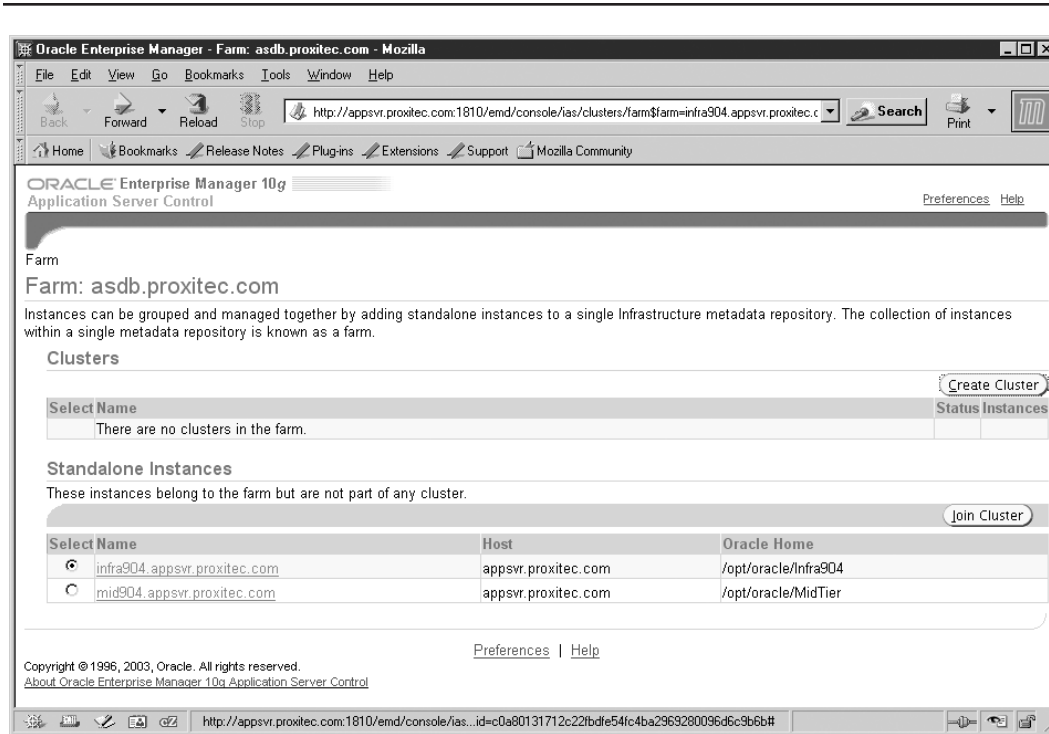


FIGURE 1-13. The main Enterprise Manager console screen

- **Infrastructure button** The Infrastructure link allows you to associate Application Server (“instances”) components with database schemas. This includes associating a component to a schema in a database that is not in the farm’s infrastructure. This allows you to share schemas across instances that do not belong to a particular farm.
- **Logs button** On the top right of this page, you can click the Logs link to see all of the log files for each component.
- **J2EE deployment button** In the Home tab, you can click J2EE Applications to see a list of all J2EE applications that are deployed on this server.
- **Ports button** This link displays the port numbers for each server component and allows you to change the port number for any component on the instance (server).

Managing Application Server 10g with Command-Line Interfaces

Many experienced Application Server 10g administrators prefer to use the command-line interfaces instead of EM. Remember, at the lowest level, EM generates the commands and

Chapter 1: Oracle Application Server 10g Architecture and Administration 23

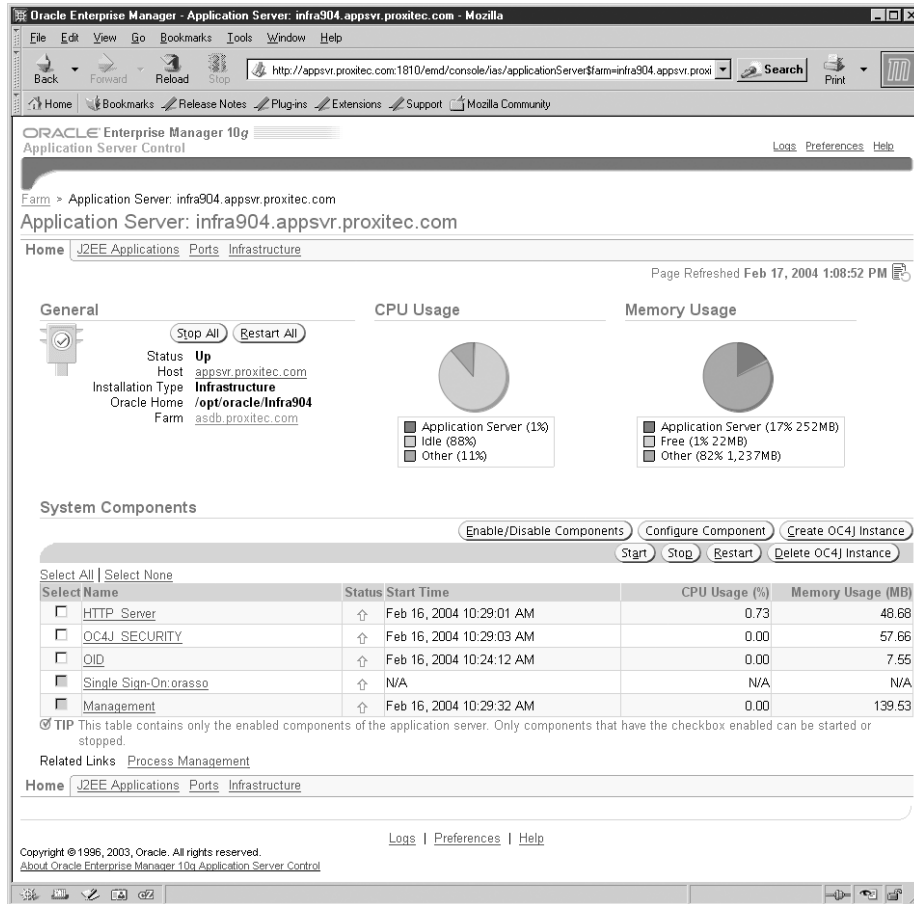


FIGURE 1-14. The EM console Instance screen

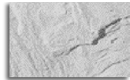
parameters for Application Server 10g control files, and knowledgeable administrators already know the commands and parameters.

The command-line interfaces are spread across many file locations, and you will find Application Server 10g command-line programs in the following directories on your operating system:

```
$ORACLE_HOME/bin/
$ORACLE_HOME/dcm/bin/
$ORACLE_HOME/j2ee/home/
$ORACLE_HOME/ldap/bin/
$ORACLE_HOME/ldap/odi/admin/
$ORACLE_HOME/oca/bin/
$ORACLE_HOME/opmn/bin/
```

24 Oracle Application Server 10g Administration Handbook

```
$ORACLE_HOME/portal/admin/plsql/sso/
$ORACLE_HOME/sso/lib/
$ORACLE_HOME/uddi/lib/
$ORACLE_HOME/upgrade/
$ORACLE_HOME/wireless/bin/
```



NOTE

To locate all of the command-line programs, you should always include the preceding directories in your \$PATH variable. In UNIX, you can place in your .profile the ksh command (if you are using the Korn shell) or the csh command (if using the C shell).

The Oracle command-line utilities will be mentioned throughout this text, but let's just take a quick tour so you can see how they are used to manage Application Server 10g. Table 1-1 shows all 59 of the command-line executables with Application Server 10g.

Category	Command	Usage
Application Server 10g	iasua.sh	This is the upgrade assistant executable.
DCM	dcmctl	The Distributed Configuration Manager is for managing Application Server 10g components.
Discoverer	eulbuilder.jar	This is the Discoverer end-user layer Java command-line interface.
DMS	dmstool	This is used for viewing performance metrics.
Forms	fplsqlconv90	This is used to update PL/SQL for Forms6i.
Forms	ifbld90	This is used to start Forms Developer.
Forms	ifcmp90	This starts the Forms Compiler.
Forms	iff2xml90	This will traverse a module object hierarchy and produce an XML representation.
Forms	ifweb90	This allows you to preview a form in a web browser.
Forms	ifxml2f90	This will take a Forms XML format and convert it back into a module.
Forms	ifxmlv90	This is the XML validator to validate .xml files against the Forms XML Schema.
J2EE	ojspc	This is the JSP back precompiler.
J2EE	admin.jar	This tool is only used in the sample OC4J standalone installation. Do not use this command.

TABLE 1-1. *Application Server 10g Command-Line Utilities*

Chapter 1: Oracle Application Server 10g Architecture and Administration **25**

Category	Command	Usage
J2EE	jazn.jar	This JAR file manages both XML-based and LDAP-based JAAS data.
LDAP	ldapadd	This is the OID add utility for adding entries, their object classes, and attributes.
LDAP	ldapaddmt	This is just like ldapadd, but with support for multiple threads for adding entries concurrently.
LDAP	ldapbind	This determines whether you can authenticate a client to a server.
LDAP	ldapcompare	This compares your attribute values with an OID entry.
LDAP	ldapdelete	This removes entries from OID.
LDAP	ldapmoddn	This modifies the DN or RDN of an Oracle Internet Directory entry.
LDAP	ldapmodify	This modifies OID attributes.
LDAP	ldapmodifymt	This modifies several OID entries concurrently.
LDAP	ldapsearch	This will search and retrieve specific OID entries.
LDAP	bulkdelete.sh	This deletes a whole OID subtree.
LDAP	bulkload.sh	This creates OID entries from data residing in or created by other applications.
LDAP	catalog.sh	This will add and delete OID catalog entries.
LDAP	hiqpurge.sh	This moves OID changes from the human intervention queue to the purge queue.
LDAP	hiqretry.sh	This moves OID changes from the human intervention queue to the retry queue.
LDAP	oidstats.sh	This analyzes Operational Data Store (ODS) schema objects to estimate statistics.
LDAP	remtool	This is used with an OID replication failure.
OCA	ocactl	This is the Certificate Authority administration tool.
OEM utility	emctl	This starts, stops, and manages security for OEM.
OID	bulkmodify	This will modify a large number OID entries.
OID	dipassistant	This is used with the Oracle Directory Integration and Provisioning platform.
OID	ldifmigrator	This is used to migrate data from application-specific repositories into OID.

TABLE 1-1. *Application Server 10g Command-Line Utilities (continued)*

26 Oracle Application Server 10g Administration Handbook

Category	Command	Usage
OID	ldifwrite	This converts OID data to LDIF and makes that information available for loading into a new node in a replicated directory or into another node for backup storage.
OID	oidctl	This will start and stop Oracle Internet Directory.
OID	oidmon	This manages OID processes.
OID	oidpasswd	This changes the OID database password.
OID	oidprovtool	This is used to administer provisioning profile entries in OID.
OID	oidreconcile	This synchronizes OID entries.
OID	resetIASpasswd.sh	This resets the internal password that instances use to authenticate themselves with OID. It resets the password to a randomly generated password.
OID	schemasync	This synchronizes schema elements between OID and third-party LDAP directories.
OID	stopodis.sh	This starts the directory integration and provisioning server without using the oidctl executable.
OPM	opmnctl	This will start, stop, and get status on OPMN-managed processes. This is the main tool for starting and stopping an instance.
OSSA	ossoca.jar	This configures additional languages for Application Server 10g Single Sign-On.
Reports	rwbuilder	This invokes the Reports Builder.
Reports	rwsgi	This translates and delivers information between HTTP and the Reports Server.
Reports	rwclient	This parses and transfers a command line to the specified (or default) Reports Server.
Reports	rwconverter	This converts report definitions or PL/SQL libraries from one storage format to another.
Reports	rwrn	This runs a report using the Application Server 10g Reports Services in-process server.
Reports	rwserver	This invokes the Reports Server.
SSO	ssocfg.sh	This updates host, port, and protocol of SSO URL.
SSO	ssooconf.sql	This points SSO to a different OID.
SSO	ossoreg.jar	This is the mod_osso registration tool.

TABLE 1-1. *Application Server 10g Command-Line Utilities (continued)*

Chapter 1: Oracle Application Server 10g Architecture and Administration 27

Category	Command	Usage
Web Cache	webcachectl	This manages Web Cache processes, including the administration server process, cache server process, and auto-restart process.
Web services	uddiadmin.jar	This manages the UDDI registry.
Wireless	portalRegistrar.sh	This reregisters the mobile gateway parameter with Application Server Portal 10g.
Wireless	reRegisterSSO.sh	This reregisters the Wireless Single Sign-On partner application with the Single Sign-On server.

TABLE 1-1. *Application Server 10g Command-Line Utilities (continued)*

Knowing these commands and their parameters for Application Server 10g is very useful for automating administrative functions and creating batch scripts. These commands can easily be placed into scripts (shell scripts in Linux and UNIX) that can be executed to automate routine management tasks.

While each product with Application Server 10g has control files, there are three main command-line interfaces:

- **opmnctl** This is the control interface for the Process Management Notification (OPM) component. The opmnctl interface is located at \$ORACLE_HOME/opmn/bin/opmnctl. The opmnctl interface provides a startall and stopall argument that will manage all of the Application Server 10g server processes.
- **dcmctl** This is the control interface for the Distributed Configuration Manager (DCM) component. The dcmctl interface is located at \$ORACLE_HOME/dcm/bin/dcmctl.
- **emctl** This is the Enterprise Manager console utility. The emctl executable is located in \$ORACLE_HOME/bin/emctl. It is used for managing the OEM agents, changing OEM passwords, starting the OEM console, and other miscellaneous tasks.

These command-line interfaces are critical for Application Server 10g administrative scripts. Let's take a look at how command-line interfaces are used as scripts.

Using Scripts to Manage Application Server 10g

You can automate many areas of Application Server 10g administration using scripts. Here is an example of a command list to start the iasdb database, the listener, the infrastructure instance, a midtier instance, and the Enterprise Manager web site on both instances.

echo Setting Env for Infrastructure

```
source envInfra.sh
echo Starting Listener
$ORACLE_HOME/bin/lsnrctl start
echo Starting Database
```

28 Oracle Application Server 10g Administration Handbook

```

$ORACLE_HOME/bin/sqlplus /nolog<<EOF
connect / as sysdba
startup
EOF
echo Starting all opmnctl controlled processes
$ORACLE_HOME/opmn/bin/opmnctl startall
echo Starting the EM website
#$ORACLE_HOME/bin/emctl start em
echo Setting Env for MidTier Instance
source envMidtier.sh
echo Starting all opmnctl controlled processes
$ORACLE_HOME/opmn/bin/opmnctl startall
echo Starting the EM website
#$ORACLE_HOME/bin/emctl start em
echo Startup Completed
    
```

By themselves, the command list is not very useful, but it becomes very powerful when embedded into a shell script. The source envMidtier.sh statement changes the ORACLE_HOME environmental variable. Each instance of Application Server 10g must be installed in its own ORACLE_HOME. This is covered in the Chapter 2. Because the Application Server 10g command-line utilities exist in many locations, it is critical that you set up your OS environment so that your scripts can locate all of the utilities. Here are examples of the proper PATH commands for UNIX and Windows. These are normally placed in the startup shell script to be executed at sign-on time.

UNIX PATH Setup

```

ORACLE_HOME=/u01/app/oracle/product/9.2.0
export ORACLE_HOME
PATH=.:$PATH:.;$ORACLE_HOME/dcm/bin/:$ORACLE_HOME/j2ee/home/:$ORACLE_HOME/ldap/bin/:$ORACLE_
HOME/ldap/odi/admin/:$ORACLE_HOME/oca/bin/:$ORACLE_HOME/opmn/bin/:$ORACLE_HOME/portal/admin/
plsql/sso/:$ORACLE_HOME/sso/lib/:$ORACLE_HOME/uddi/lib/:$ORACLE_HOME/upgrade/:$ORACLE_HOME/
wireless/bin/
    
```

Windows PATH Setup

```

Set ORACLE_HOME=c:\oracle\ora92
SETPATH=.;$PATH;%ORACLE_HOME%\dcm\bin\;%ORACLE_HOME%\j2ee\home\;%ORACLE_HOME%\ldap\bin\
;%ORACLE_HOME%\ldap\odi\admin\;%ORACLE_HOME%\oca\bin\;%ORACLE_HOME%\opmn\bin\;%ORACLE_HOME%\
portal\admin\plsql\sso\;%ORACLE_HOME%\sso\lib\;%ORACLE_HOME%\uddi\lib\;%ORACLE_HOME%\upgrade\
;%ORACLE_HOME%\wireless\bin\
    
```

Once you have established the PATH variable, you can create shell scripts that can be submitted in batch mode (in UNIX with the nohup command) to automate Application Server 10g administrative tasks. For example, an Application Server 10g management shell script could be scheduled in the UNIX crontab to perform a scheduled shutdown of all services.

Of course, the PATH variable is only a part of an Application Server 10g script, and the complete environment, including ORACLE_BASE, ORACLE_HOME, and ORACLE_SID, must be enabled. The env.ksh script shows a common environmental setting for Application Server 10g command scripts. Note that \$ORACLE_HOME is set to ORACLE_BASE/midtier for midtier command scripts and ORACLE_BASE/infra for infrastructure command scripts. Every Application Server 10g

Chapter 1: Oracle Application Server 10g Architecture and Administration 29

instance must be installed in a unique ORACLE_HOME, and startup/shutdown scripts must set the environment variables for that instance.

env.sh

```
#!/bin/ksh
export ORACLE_BASE=/private/ias
# Use this ORACLE_HOME for midtier applications
#export ORACLE_HOME=$ORACLE_BASE/midtier
# Use this ORACLE_HOME for infra applications
export ORACLE_HOME=$ORACLE_BASE/infra
SETPATH=.;$PATH;%ORACLE_HOME%\dcm\bin\;%ORACLE_HOME%\j2ee\home\;%ORACLE_HOME%\ldap\bin\
;%ORACLE_HOME%\ldap\odi\admin\;%ORACLE_HOME%\oca\bin\;%ORACLE_HOME%\opmn\bin\;%ORACLE_HOME%\
portal\admin\plsql\sso\;%ORACLE_HOME%\sso\lib\;%ORACLE_HOME%\uddi\lib\;%ORACLE_HOME%\upgrade\
;%ORACLE_HOME%\wireless\bin\
export ORACLE_SID=iasdb
export LD_LIBRARY_PATH=$LD_LIBRARY_PATH:$ORACLE_HOME/lib
#export DISPLAY=tor:2.0

echo ORACLE_HOME : $ORACLE_HOME
echo ORACLE_SID : $ORACLE_SID
echo DISPLAY : $DISPLAY
echo Set PATH and LD_LIBRARY_PATH
```

Here is a script to submit when there is a problem with OHS and you need to restart it. Some Application Server 10g administrators place Apache user-exit code to automate the bouncing of the OHS. For example, if an external connection fails to attach to an OHS listener, after ten seconds, the following code could be automatically invoked to bounce OHS.

bounce_ohs.ksh

```
*****
# Copyright (c) 2003 by Donald K. Burleson
#
#*****
# Exit if no first parameter $1
if [ -z "$1" ]
then
    echo "ERROR: Please pass a valid ORACLE_SID to this script"
    exit 99
fi
# Validate Oracle
TEMP=`cat /etc/oratab|grep `^$1:|cut -f1 -d'|wc -l`
tmp=`expr $TEMP`          # Convert string to number
if [ $tmp -ne 1 ]
then
    echo
    echo "ERROR: Your input parameter $1 is invalid. Please Retry"
    echo
    exit 99
fi

if [ `whoami` != 'oracle' ]
then
    echo "Error: You must be oracle to execute the script. Exiting."
    exit
fi
```

30 Oracle Application Server 10g Administration Handbook

```
# First, we must set the environment . . . .
export ORACLE_BASE=/private/ias
# Use this ORACLE_HOME for midtier applications
#export ORACLE_HOME=$ORACLE_BASE/midtier
# Use this ORACLE_HOME for infra applications
export ORACLE_HOME=$ORACLE_BASE/infra
SETPATH=.;$PATH;%ORACLE_HOME%\dcm\bin\;%ORACLE_HOME%\j2ee\home\;%ORACLE_HOME%\ldap\bin\
;%ORACLE_HOME%\ldap\odi\admin\;%ORACLE_HOME%\oca\bin\;%ORACLE_HOME%\opmn\bin\;%ORACLE_HOME%\
portal\admin\plsq\l\sso\;%ORACLE_HOME%\sso\lib\;%ORACLE_HOME%\uddi\lib\;%ORACLE_HOME%\upgrade\
;%ORACLE_HOME%\wireless\bin\
export ORACLE_SID=iasdb
export LD_LIBRARY_PATH=$LD_LIBRARY_PATH:$ORACLE_HOME/lib
#export DISPLAY=tor:2.0
#*****
# Execute the DCM commands to bounce the OHS
#*****
$ORACLE_HOME/dcm/bin/dcmctl stop -ct ohs
$ORACLE_HOME/dcm/bin/dcmctl start -ct ohs
$ORACLE_HOME/dcm/bin/dcmctl start -co OC4J-Portal
```

As you can see, these shell scripts with embedded Application Server 10g commands are extremely useful for automatic administration. As each component is discussed in later chapters, detailed scripts will be introduced to assist with the administration of that component.

The next few sections give examples of commands that are used to perform frequent Application Server 10g administrative functions.

EM Commands with emctl

The emctl utility is used to manage all aspects of the Enterprise Manager console. While the EM console is greatly useful for managing components of Application Server 10g, the console itself must be managed. For example, all EM servers must have a running EM agent, and the emctl command-line utility can be used to start or stop OEM, OMS, or any OEM agent. As mentioned earlier in the chapter, OMS (Oracle Management Server) is used with Oracle Enterprise Manager for centralized management of all Oracle products installed.

EM Console Commands for Application Server 10g

```
emctl start em
emctl stop em
emctl status em
```

OEM Agent Commands

```
emctl start agent
emctl stop agent
emctl status agent
```

OMS Commands

```
emctl start oms
emctl stop oms
emctl status oms
```

Chapter 1: Oracle Application Server 10g Architecture and Administration 31

Managing Application Server 10g with opmnctl

The Oracle Process Manager and Notification (OPMN) uses the opmnctl utility to manage all Oracle Application Server 10g server processes. The powerful startall and stopall commands will manage all server components. Unless a tier consists of a standalone component such as the Web Cache, opmnctl should be used rather than the separate component control program.

Start OPMN, DCM, and All Components

```
opmnctl startall
```

Stop OPMN, DCM, and All Components

```
opmnctl stopall
```

There may be times when you want to stop and restart all OPMN and DCM processes on your servers. The following shell script will perform this function:

```
#!/bin/ksh
#*****
#
# Copyright (c) 2003 by Donald K. Burlison
#
# Bounce all Oracle Application Server 10g server processes
#
#*****
# First, we must set the environment . . .
export ORACLE_BASE=/private/ias
# Use this ORACLE_HOME for midtier applications
#export ORACLE_HOME=$ORACLE_BASE/midtier
# Use this ORACLE_HOME for infra applications
export ORACLE_HOME=$ORACLE_BASE/infra
SET PATH=.:$PATH;%ORACLE_HOME%\dcm\bin\;%ORACLE_HOME%\j2ee\home\;
%ORACLE_HOME%\dap\bin\;%ORACLE_HOME%\ldap\odi\admin\;%ORACLE_HOME%\oca\bin\;%ORACLE_HOME%\opmn\
bin\;%ORACLE_HOME%\portal\admin\plsql\sso\;%ORACLE_HOME%\sso\lib\;%ORACLE_HOME%\uddi\lib\
;%ORACLE_HOME%\upgrade\;%ORACLE_HOME%\wireless\bin\
export ORACLE_SID=iasdb
export LD_LIBRARY_PATH=$LD_LIBRARY_PATH:$ORACLE_HOME/lib
#export DISPLAY=tor:2.0
# Loop through each host name . . .
for host in `cat -oracle/.rhosts|cut -d"." -f1|awk '{print $1}'|sort -u`
do
    # Get the ORACLE_HOME on each Oracle Application Server 10g server
    home=`rsh $host "cat /etc/oratab|egrep ':N|:Y'|grep -v \*|cut -f1 /
d':"``
    # Execute opmnctl to bounce all Oracle Application Server 10g server processes:
    rsh $host "$home/opmn/bin/opmnctl stopall"
    rsh $host "$home/opmn/bin/opmnctl startall"
done
```

Managing Application Server 10g with dcmctl

The Distributed Configuration Manager (DCM) is the master utility for Application Server 10g. The DCM is responsible for maintaining configuration by updating the configuration files on each server. DCM also stores the values of the parameters within each configuration file on each server.

32 Oracle Application Server 10g Administration Handbook

within isadb. Note that if you choose not to implement the infrastructure (not recommended), the parameter files will exist as flat files on each server.

The dcmctl utility has two important argument settings, verbose (-v) and diagnostic (-d). These are important options because they provide additional diagnostic information about the state of your dcmctl commands. Starting in Application Server 10g release 9.0.4, you can use set commands to enable and disable these options:

```
dcmctl set -v on
dcmctl set -d on
```

Once you have established the settings, you can use dcmctl for a variety of Application Server 10g administrative functions. Here is an example of using dcmctl to start the HTTP server (OHS):

```
dcmctl start -ct ohs
http://diogenes:7777
http://diogenes:1080
```

You can also use the dcmctl command to deploy OC4J applications, and the dcmctl commands can be embedded into command lists for the purpose of deploying them on many servers. For example, let's create a list of dcmctl commands to deploy an OC4J application on multiple servers. Assume that we have saved this file as /home/oracle/dcm_dep.cmd on our main server:

dcm_dep.cmd

```
echo "creating testcluster"
createcluster testcluster
echo "joining testcluster"
joincluster testcluster
echo "creating component component1"
createcomponent -ct oc4j -co component1
echo "starting component to deploy application"
start -co component1
echo " deploying application"
deployapplication -f /stage/apps/appl.ear -a appl -co component1
echo "starting the cluster"
start -cl testcluster
echo "verifying everything started "
getstate
exit
```

So, how can you execute this script on all 20 of your OC4J servers? You can create a shell script to loop through a list of all servers and deploy the J2EE application on each OC4J server. This script requires the remote shell (rsh) and remote copy (rcp) UNIX commands. The rsh command is enabled by placing server host names in your .rhosts file. The rsh facility should only be implemented if all Application Server 10g servers are safe behind a firewall, because rsh allows a hack who gains access to one server to access all other servers in the .rhosts file.

Chapter 1: Oracle Application Server 10g Architecture and Administration 33

deploy_oc4j.ksh

```
#!/bin/ksh
#*****
#
# Copyright (c) 2003 by Donald K. Burlison
#
# Deploy Oracle Application Server 10g application on multiple servers
#
#*****

# First, we must set the environment . . . .
export ORACLE_BASE=/private/ias
# Use this ORACLE_HOME for midtier applications
#export ORACLE_HOME=$ORACLE_BASE/midtier
# Use this ORACLE_HOME for infra applications
export ORACLE_HOME=$ORACLE_BASE/infra
SET PATH=.;$PATH;%ORACLE_HOME%\dcm\bin\;%ORACLE_HOME%\j2ee\home\;
%ORACLE_HOME%\ldap\bin\;%ORACLE_HOME%\ldap\odi\admin\;%ORACLE_HOME%\
oca\bin\;%ORACLE_HOME%\opmn\bin\;%ORACLE_HOME%\portal\admin\plsql\sso\;
%ORACLE_HOME%\sso\lib\;%ORACLE_HOME%\uddi\lib\;%ORACLE_HOME%\upgrade\;
%ORACLE_HOME%\wireless\bin\
export ORACLE_SID=iasdb
export LD_LIBRARY_PATH=$LD_LIBRARY_PATH:$ORACLE_HOME/lib
#export DISPLAY=tor:2.0

# Loop through each host name . . .
for host in `cat ~oracle/.rhosts|cut -d"." -f1|awk '{print $1}'|sort -u`
do
    # Get the ORACLE_HOME on each Oracle Application Server 10g server
    home=`rsh $host "cat /etc/oratab|egrep ':N|:Y'|grep -v \*|cut -f1/
d':"'`

    # Copy the dcm command file and ear to the remote server
    rcp -p /home/oracle/dcm_dep.cmd ${host}:~oracle/dcm_dep.cmd
    rcp -p /stage/apps/app1.ear ${host}:/stage/apps/app1.ear

    # Set and check file permissions
    rsh $host "chmod 500 ~oracle/dcm_dep.cmd "
    rsh $host "ls -al ~oracle/dcm_dep.cmd"

    # Execute dcmctl to start the dcmctl shell:
    rsh $host "$home/dcm/bin/dcmctl shell -f ~oracle/dcm_dep.cmd"

done
```

As you can see, the dcmctl command is very useful when you must deploy applications across many J2EE instances.

34 Oracle Application Server 10g Administration Handbook

Miscellaneous Application Server 10g Commands

While we will be exploring these commands in greater detail in later chapters, we want to introduce a few more common command-line utilities for Application Server 10g administration. Let's take a quick look at commands for managing the Web Cache (webcachectl) and the Oracle Internet Directory (oidctl and oidmon).

Shut Down the Application Server 10g Web Cache

```
webcachectl stop
```

Start the isadb Instance

```
oidmon connect=iasdb start  
oidctl server=oidldapd instance=s flags="-port 4032 -host myhost" start
```

Summary

This chapter has given you an overview of Application Server 10g and all of its components. Application Server 10g is now the encapsulation of many application-related products, each with unique features and functionality. Remember, many of the components are optional, and few shops use all of them. The main points of this chapter can be summarized as follows:

- Oracle has implemented a flexible architecture for Application Server 10g, allowing administrators to define multiple servers to manage the application load.
- Application Server 10g architectures may be defined as two-tiered, three-tiered, or four-tiered, and there may be many independent servers at each tier.
- Multiple Application Server 10g instances can be grouped into farms, which are instances that share a common metadata repository.
- Common Application Server 10g instances can be grouped into clusters, which are instances that share a common definition and J2EE applications.
- The Oracle Management Server (OMS) helps with management and definition of farms, clusters, and instances, making it simpler to manage complex application environments.
- TopLink is an important component of Oracle Application Server 10g that allows Java objects to be stored for future reference by other Java tasks.
- For those who install the Application Server 10g Infrastructure, the Enterprise Manager is a fast and easy way to perform administrative functions.
- Application Server 10g defines a hierarchy of components, instances, clusters, and farms. Each farm has many clusters, each cluster has many instances, and each instance has many components.
- Application Server 10g also provides 59 command-line interfaces. The most popular command-line interfaces are emctl, dcmctl, and opmnctl.

Now that we have completed the high-level tour, we are ready to move on to examine the Application Server 10g Infrastructure in more detail. The Infrastructure includes SSO, LDAP, and the all-important metadata repository.