Oracle® E-Business Suite Concepts

Architecture

Introduction

This chapter describes the Oracle E-Business Suite architecture and the key features the architecture supports, plus some related points. Topics include:

- Overview
- The Client (Desktop) Tier
- The Application Tier
- The Database Tier
- The Oracle E-Business Suite Technology Layer
- Oracle Configuration Manager
- Oracle E-Business Suite Patch Nomenclature

The *Oracle E-Business Suite Architecture* is a framework for multi-tiered, distributed computing that supports Oracle E-Business Suite products. In this model, various *servers* or *services* are distributed among three levels, or *tiers*.

A server (or services) is a process or group of processes that runs on a single machine and provides a particular functionality. For example, *Web services* process HTTP requests, and *Forms services* process requests for activities related to Oracle Forms. The *Concurrent Processing server* supports data-intensive programs that run in the background.

Important: The term *server*, in the sense of a single process, is less appropriate in the Release 12 architecture. Where applicable, replacement terms such as *services* are used.

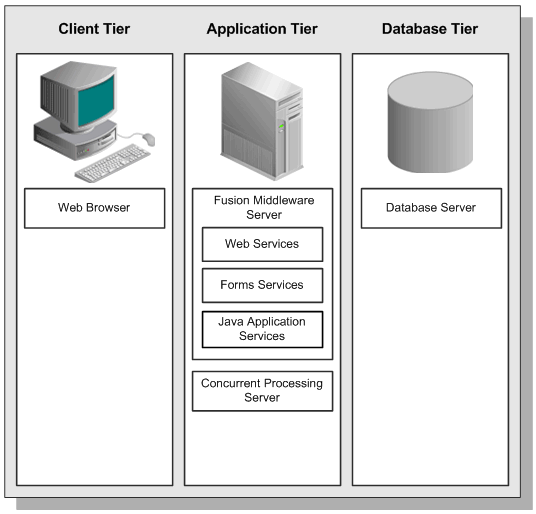
A tier is a logical grouping of services, potentially spread across more than one physical machine. The *three-tier architecture* that comprises an Oracle E-Business Suite installation is made up of the *database tier*, which supports and manages the Oracle database; the *application tier*, which supports and manages the various Oracle E-Business Suite components, and is sometimes known as the middle tier; and the *client (desktop) tier*, which provides the user interface via an add-on component to a standard web browser.

A machine may be referred to as a *node*, particularly in the context of a group of computers that work closely together in a *cluster*. Each tier may consist of one or more nodes, and each node can potentially accommodate more than one tier. For example, the database can reside on the same node as one or more application tier components. Note, however, that a node is also a software concept, referring to a logical grouping of servers.

Centralizing the Oracle E-Business Suite software on the application tier eliminates the need to install and maintain application software on each client PC, and also enables Oracle E-Business Suite to scale well with an increasing load. Extending this concept further, one of the key benefits of using the *Shared Application Tier File System* model (originally *Shared APPL_TOP*) is the need to maintain only a single copy of the relevant Oracle E-Business Suite code, instead of a copy for every application tier machine.

On the database tier, there is increasing use of *Oracle Real Application Clusters* (Oracle RAC) , where multiple nodes support a single database instance to give greater availability and scalability.

Oracle E-Business Suite Three-Tier Architecture

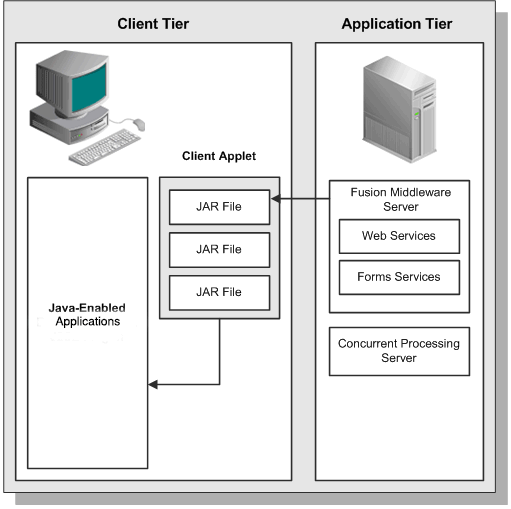


The Client Tier

The client interface is provided through HTML for the large number of HTML-based applications, and via Java applets for the smaller number of Forms-based applications and a few product-specific features.

Note: The client tier is sometimes referred to as the *desktop tier*.

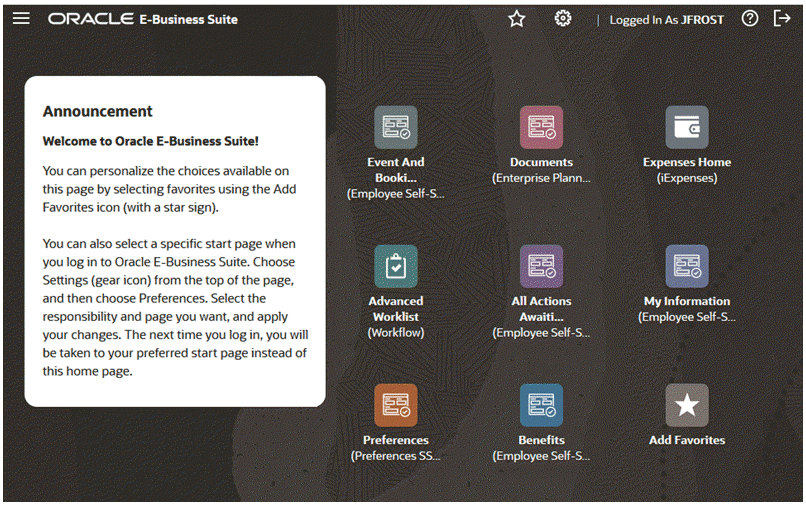
Client Tier and Application Tier Components



You log in via the Oracle E-Business Suite Home Page on a desktop client web browser. The Home Page provides a single point of access to HTML-based applications, Forms-based applications, and Business Intelligence applications.

Once successfully logged in via the E-Business Suite Home Page, you are not prompted for your user name and password again, even if you navigate to other tools and products. Oracle E-Business Suite also retains preferences as you navigate through the system. For example, if you registered in the Home Page that German is your preferred language, this preference carries over whether you access Forms-based or HTML-based applications.

Oracle E-Business Suite Home Page



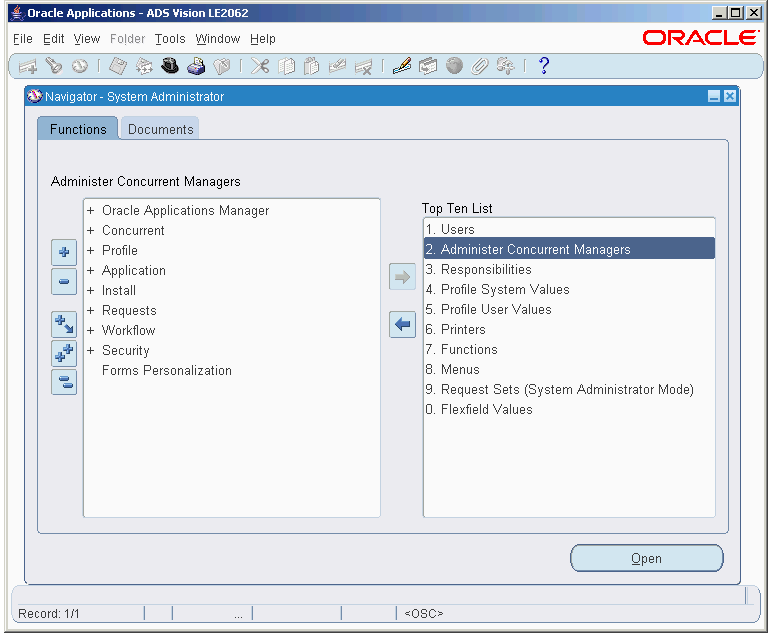
Forms Client Applet

The *Forms client applet* is a general-purpose presentation applet that supports all Oracle E-Business Suite Forms-based products, including those with customizations and extensions. The Forms client applet is packaged as a collection of *Java Archive* (JAR) files. The JAR files contain all Java classes required to run the presentation layer of Oracle E-Business Suite forms.

Desktop Java Runtime Environment

The Forms client applet must run within a Java Virtual Machine (JVM) on the desktop client. It is launched on the user's desktop using a Java Deployment technology, either Java Web Start or Java Plug-in. These are included as part of the Java Runtime Environment (JRE) software. Java Deployment technologies require the JRE software to be installed on the client, so that Java-based applications can run on it. After you download and install the JRE software, you will be able to run Forms-based applications as shown in the example on the following screenshot.

Forms-Based Oracle E-Business Suite Interface



The Forms client applet and commonly used JAR files are downloaded from the Web server at the beginning of the client's first session. Less commonly used JAR files are downloaded as needed. All downloaded JAR files are cached locally on the client, ready for future sessions. This eliminates the network traffic that would be involved in downloading them whenever they were required.

In Release 12, the cache directory path is of the form:

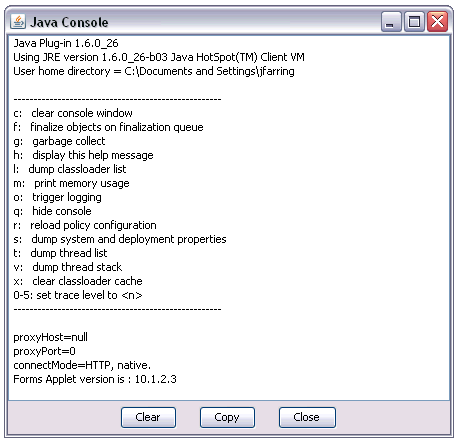
```
<HOMEDRIVE>\Documents and Settings\<Windows User Name>\Application Data\Sun\Java\Deployment\cache
```

For example:

```
C:\Documents and Settings\jsmith\Application Data\Sun\Java\Deployment\cache
```

Selecting "Show console" on the "Advanced" tab of the JRE Plug-in control panel will allow you to observe downloading of JAR files, to confirm they are being downloaded when they should be. The Java console is shown in the following screenshot.

Java Console



All updates to JAR files are installed on the application tier and downloaded to the client automatically, via the caching mechanism outlined above.

Java Web Start

Java Web Start launches Oracle E-Business Suite Java-based functionality as Java Web Start applications instead of as Java applets. Java Web Start is a Java Deployment technology that is installed as part of the Java Runtime Environment (JRE) software. It provides a browser-independent architecture for deploying Java technology-based applications to the client desktop, and is the recommended configuration for Oracle E-Business Suite.

Additional Information: For details about how you can migrate an existing Oracle E-Business Suite Release 12.2 or 12.1 environment to use Java Web Start, refer to My Oracle Support Knowledge Document 2188898.1, *Using Java Web Start with Oracle E-Business Suite*.

Java Plug-in

Running Oracle E-Business Suite's Java-based content using the Java Plug-in requires a browser that supports Netscape Plug-in Application Programming Interface (NPAPI) plug-ins. Most browsers have now phased out NPAPI plug-in support, preventing he Java Plug-in from working. Oracle E-Business Suite supports launching Java applets (such as the Forms applet) using the Java Plug-in for browsers that support it.

Additional Information: For further details, refer to My Oracle Support knowledge documents 393931.1, *Deploying JRE (Native Plug-in) for Windows Clients in Oracle E-Business Suite Release 12*, and 2510500.1, *FAQ: Essentials of Java Usage in Oracle E-Business Suite*.

The Application Tier

The *application tier* has a dual role: hosting the various servers and service groups that process the business logic, and managing communication between the desktop tier and the database tier. This tier is sometimes still referred to as the *middle tier*.

Several service groups or servers comprise the basic Oracle E-Business Suite application tier:

- HTTP services
- Java services
- Forms services
- Concurrent Processing server

As well as these, some less visible application tier services provide further infrastructure support.

In Release 12.2, Web and Forms services are provided by *Oracle Application Server* and Oracle Fusion Middleware. They are no longer servers in the sense of being a single process, as was the case in previous releases.

Tip: It is advisable to avoid using a mixture of different platforms on your application tier. This makes maintenance easier, since only one set of patches needs to be downloaded.

Load Balancing

The application tier supports load balancing among many of its servers and services to help provide higher availability, fault tolerance, reliability, and optimal scalability. If you have more than one of any of the following types of server, load balancing can be employed:

- Web services
- Forms services
- Concurrent Processing server

Load balancing is discussed in more detail in a later chapter.

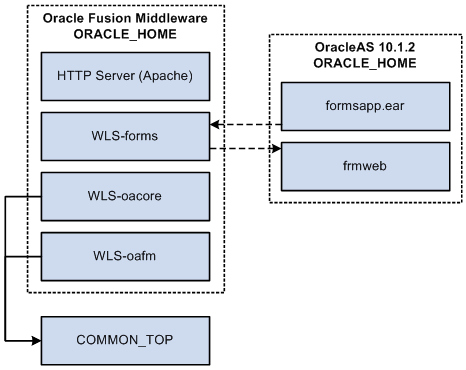
Application Tier ORACLE_HOMEs in Release 12.2

Oracle E-Business Suite Release 12.2 uses two application tier ORACLE_HOMEs.

- An OracleAS 10.1.2 ORACLE_HOME that was used in previous 12.x releases.
- An Oracle Fusion Middleware (FMW) ORACLE_HOME that supports Oracle WebLogic Server (WLS) and supersedes the Java (OracleAS 10.1.3) ORACLE_HOME that was used in previous releases.

The use of these two ORACLE_HOMEs enables Oracle E-Business Suite to take advantage of the latest Oracle technologies.

Application Tier Structure



Notable features of this architecture include:

- The Oracle E-Business Suite modules (packaged in the file formsapp.ear) are deployed out of the OracleAS 10.1.2 ORACLE_HOME, and the `frmweb` executable is also invoked out of this ORACLE_HOME.
- All major services are started out of the Fusion Middleware ORACLE_HOME.

Key changes from earlier releases include:

- The Oracle Application Server 10.1.2 ORACLE_HOME (sometimes referred to as the Tools, C, or Developer ORACLE_HOME) replaces the 8.0.6 ORACLE_HOME provided by Oracle9i Application Server 1.0.2.2.2 in Release 11i.
- The FMW ORACLE_HOME (sometimes referred to as the Web or Java ORACLE_HOME) replaces the OracleAS 10.1.3.-based ORACLE_HOME used in Oracle E-Business Suite 12.x releases prior to 12.2.

Web Services

The Web services component of Oracle Application Server processes requests received over the network from the desktop clients, and includes the following major components:

- Web Listener (Oracle HTTP Server powered by Apache)
- Java Servlet Engine (Oracle WebLogic Server, WLS)

The Web listener component of the Oracle HTTP server accepts incoming HTTP requests (for particular URLs) from client browsers, and routes the requests to WLS.

If possible, the Web server services the requests itself, for example by returning the HTML to construct a simple Web page. If the page referenced by the URL needs advanced processing, the listener passes the request on to the *servlet engine*, which contacts the database server as needed.

Note: Oracle WebLogic Server integration is new in Release 12.2.

HTML-Based Applications and the Oracle Application Framework

The Oracle HTML-based applications (originally known as Self-Service applications) add a browser-based, walk-up-and-use functionality to Oracle E-Business Suite. They include numerous products such as Self-Service Expenses, Self-Service Human Resources, Internet Procurement, Internet Receivables, Self-Service Time, Web Suppliers, iStore, iPayment, iSupport, and iMarketing, and have the following characteristics:

- Do not use Oracle Forms for the interface
- Use HTML documents, JavaScript, Java Server Pages, JavaBeans, and Servlets
- Dynamically generate HTML pages by executing Java code
- Use a metadata dictionary for flexible layout
- Operate via a direct connection to the Web server

The Oracle HTML-based applications can be either inquiry or transactional. Inquiry modules only read the Oracle E-Business Suite database. In contrast, transactional modules both read and update the database.

The *Oracle Application Framework* is the development platform for HTML-based applications. It consists of a Java-based application tier framework and associated services, designed to facilitate the rapid deployment of HTML-based applications.

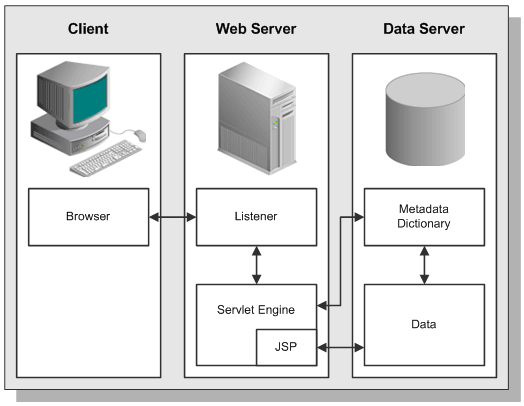
Notable Oracle Application Framework components include:

- *Business Components for Java* (BC4J), included in Oracle JDeveloper, is used to create Java business components for representing business logic. It also provides a mechanism for mapping relational tables to Java objects, and allows the separation of the application business logic from the user interface.

- *Oracle WebLogic Server* supplies the Oracle Application Framework with underlying security and applications Java services. It provides the Oracle Application Framework with its connection to the database, and with application-specific functionality such as flexfields.

The Framework-based applications logic is controlled by procedures that execute through the Java servlet engine, which is provided by the Apache JServ module. The servlet engine uses the metadata dictionary in constructing the Framework UI.

HTML-Based Applications Architecture

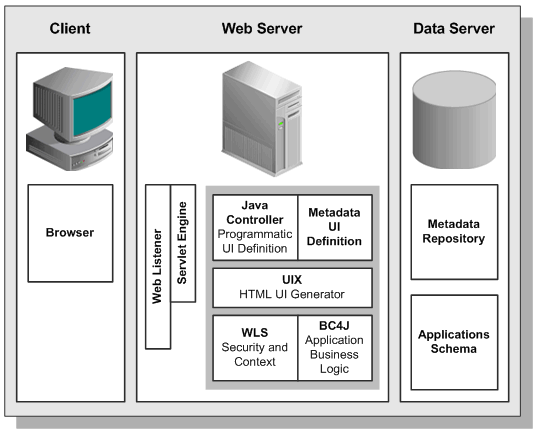


Java Servlet Access with HTML-Based Applications

An HTML-based Applications module uses the following access path:

1. The user clicks the hyperlink of a function from a browser.
2. The browser makes a URL request to the Web listener.
3. The Web listener contacts the Servlet engine (Oracle WebLogic Server), where it runs a JSP.
4. The JSP obtains the content from the Oracle E-Business Suite tables and uses information from the metadata dictionary to construct the HTML page.
5. The resulting HTML page is passed back to the browser, via the Web server.

Oracle Application Framework Architecture



Oracle Application Framework Processing Details

The following is a more detailed explanation of how the JSP obtains the content from the Oracle E-Business Suite tables and uses information from the metadata dictionary to construct the HTML page.

1. Oracle WebLogic Server validates user access to the page.
2. The page definition (metadata UI definition) is loaded from the metadata repository on the database tier into the application tier.
3. The BC4J objects that contain the application logic and access the database are instantiated.
4. The Java Controller programmatically manipulates the page definition as necessary, based on dynamic UI rules.
5. UIX (HTML UI Generator) interprets the page definition, creates the corresponding HTML in accordance with UI standards, and sends the page to the browser.

Forms Services

By default, Forms services in Oracle E-Business Suite Release 12.2 are provided by the *Forms listener servlet*, which, as described further below, facilitates the use of firewalls, load balancing, proxies, and other networking options.

Benefits of using the Forms listener servlet include:

- Ability to re-establish dropped network connections
- Fewer machines and ports need to be exposed at the firewall
- Easier firewall/proxy server configuration
- More robust and secure deployment over the Internet

Forms Listener Servlet Architecture

The Forms listener servlet is a Java servlet that delivers the ability to run Oracle Forms applications over HTTP or HTTPS connections. It hosts the Oracle E-Business Suite forms and associated runtime engine, mediating the communication between the desktop client and the Oracle database server, displaying client screens, and initiating changes in the database according to user actions.

The Forms listener servlet caches data and provides it to the client as needed, for example when scrolling through multiple order lines that exceed the limitations of a single screen.

The Forms listener servlet can communicate with the desktop client using either a standard HTTP network connection or secure HTTPS network connection. In contrast, Forms services (formerly known as Forms server) communicates with the desktop client using the TCP/IP network protocol, on top of which it layers its own protocol.

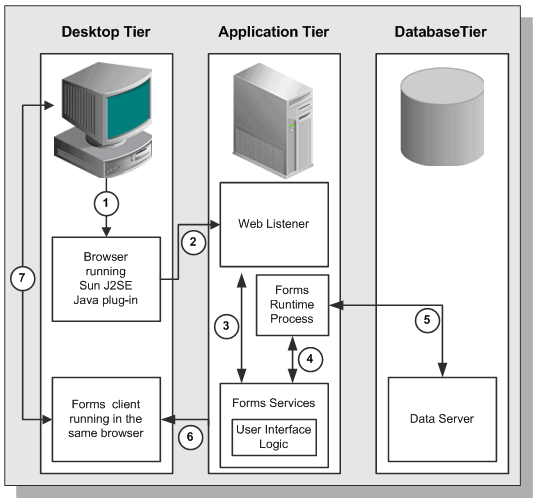
The Forms listener servlet communicates with the Oracle database server using the *Oracle Net* networking infrastructure.

The Forms listener servlet manages the creation of a Forms runtime process for each client, as well as network communications between the client and its associated Forms runtime process. The client sends HTTP requests and receives HTTP responses from the Web services, which acts as the network endpoint for the client.

Forms Socket Mode Architecture

In the traditional Forms server *socket mode* architecture, when a user initiates an action in the Forms client applet (such as entering data into a field or clicking a button), data is passed to a Forms server on the application tier. The user interface logic runs in the Forms server, and determines the appropriate user interface effect based on the user's action. For example, a window may open, or another field value may be populated. If necessary, the database tier is contacted for any data not already cached on the application tier, or for data-intensive processing.

Forms Socket Mode Architecture



Once a connection has been made, many operations can be performed with little or no further interaction with the Forms server. For example, when a few field values change in response to a user action, there is no need to update the entire screen. In this scenario, only the changed fields are updated with the new values.

Choice of Mode

As stated, by default Oracle E-Business Suite Release 12.2 utilizes Forms listener servlet mode. However, socket mode is still supported, and may be useful in high-latency, bandwidth-constrained WAN environments.

Additional Information: For more details of utilizing Forms Socket Mode, see My Oracle Support Knowledge Document 384241.1, *Using Forms Socket Mode with Oracle E-Business Suite Release 12*.

Java APIs for Forms

Building on top of Oracle Fusion Middleware and service-oriented architecture (SOA) technology, *Oracle E-Business Suite Integrated SOA Gateway* (ISG) provides a robust communication and integration infrastructure between independently managed components and loosely coupled applications. This promotes more effective integration between heterogeneous applications, and facilitates the development and execution of complex business processes into flexible and reusable Web services. With this standardized Web service platform, Oracle E-Business Suite Integrated SOA Gateway provides a framework that efficiently supports service integration between Web applications.

Note: For further details, see My Oracle Support Knowledge Document 556540.1, *Installing Oracle E-Business Suite Integrated SOA Gateway, Release 12*.

Within this integration environment, Java APIs for Forms are XML document-based integration points wrapped in Java classes, and used to execute business logic in Oracle Forms. These specialized Java classes are categorized as a subtype of Java interface, and can be service-enabled through SOA Provider.

Note: For further details of Java APIs and Web services, see Oracle E-Business Suite Integrated SOA Gateway User's Guide, Development Guide, and Implementation Guide.

Concurrent Processing Server

As described previously, user interactions with Oracle E-Business Suite data can be conducted via HTML-based applications or the more traditional Forms-based applications. However, there are also reporting programs and data updating programs that need to run either periodically, or on an ad hoc basis. These programs, which run in the background while users continue to work on other tasks, may require a large number of data-intensive computations, and are run using the *Concurrent Processing* architecture.

Concurrent Processing is an Oracle E-Business Suite feature that allows these non–interactive and potentially long-running functions to be executed efficiently alongside interactive operations. It uses operating system facilities to enable background scheduling of data- or resource-intensive jobs, via a set of programs and forms.To ensure that resource-intensive concurrent processing operations do not interfere with interactive operations, they are run by a specialized component , the *Concurrent Processing server*.

Concurrent Processing Server Implementation

The server is located on the Oracle E-Business Suite application tier, and implemented via the *Batch Processing Services* service group.

Note: For further information about application tier services and service groups, see *Oracle E-Business Suite Installation Guide: Using Rapid Install*, Chapter 2.

Concurrent Requests and Managers

Processes that run on the Concurrent Processing server are called *concurrent requests*. When you submit such a request, either through HTML-based or Forms-based applications, a row is inserted into a database table specifying the program to be run. A *concurrent manager* then reads the applicable requests in the table, and starts the associated concurrent program.

Concurrent Manager Location

Traditionally, the concurrent managers were installed on the same machine as the database. However, with the much faster local networks that are now common, and the typical co-location of database and application tier machines, it is generally preferable to run them on a separate machine. This greatly faciliates ease of management and maintenance, as well as providing deployment flexibility.

Concurrent Manager Characteristics

Concurrent managers are fundamental to concurrent processing. Acting as a job scheduling and execution system, a concurrent manager:

- Is an executable that is registered as a program library within Oracle E-Business Suite, and which runs in its own operating system process
- Runs operating system processes called *target processes* (often referred to as *workers*), each of which can start one concurrent program at a time
- Can optionally run an *immediate program* that runs as part of the concurrent manager's own operating system process
- Can be allowed to run any concurrent program, or be specialized to run certain programs
- Operates during the days and times defined by a *work shift*

Specialist Concurrent Managers

As well as the general-purpose concurrent managers described in the previous section, there are a number of specialized concurrent managers.

The *Internal Concurrent Manager* (ICM) controls all other concurrent managers. It administers the startup and shutdown of managers as defined by their work shift, monitors for process failure, and cleans up if a failure occurs. The ICM does not process concurrent requests itself (except for queue control requests, such as ACTIVATE, DEACTIVATE, or ABORT).

While the basic ICM definition should not be changed, you can if required modify the *sleep time* (number of seconds the ICM waits between checking for new concurrent requests), *PMON (process monitor) cycle time* (number of sleep cycles the ICM waits between checking for failed workers), and *queue size* (duration between checks for number of active workers, measured in PMON cycles). If Parallel Concurrent Processing (described below) is being used, you can also set some options for this.

The *Conflict Resolution Manager* (CRM) enforces rules designed to ensure that incompatible concurrent requests do not run in the same *conflict domain* (an abstract representation of the groupings used to partition data). As with the Internal Concurrent Manager, the basic CRM definition should not be changed, but you can modify the sleep time for each work shift, as well as some Parallel Concurrent Processing options.

The *Standard Manager* as shipped with Oracle E-Business Suite will accept and run any concurrent requests, as it has no specialization rules that would restrict its activities. Consequently, the definition of the Standard Manager should not be altered without careful planning, otherwise some programs might not be able to run at all. Jobs should only be excluded from the Standard Manager after ensuring they can be run by an alternative manager, such as a product-specific manager or user-defined manager.

Transaction Managers support synchronous request processing, whereby a pool of server processes responds to requests from client programs. Instead of polling the concurrent requests table to obtain instructions, a transaction manager waits to be signaled by a client. An example is approval of an order, where execution of the request must take place quickly.

The relevant transaction manager program runs on the server, transparently to the client. All transaction programs for a given manager process run in the same database session. Communication between the client and the server is conducted synchronously via Advanced Queueing. At the end of program execution, the client program receives a completion message and a return value, for example denoting approval of the order. This strategy of using non-persistent connections between the client and Transaction Manager processes enables a small pool of server processes to service a large number of clients with near real-time response.

Setting Up Concurrent Managers

Oracle E-Business Suite Setup Guide gives full details of the steps and options involved in setting up and monitoring concurrent managers. Some of the key steps include:

- Name and description of the manager
- Assignment of a concurrent program library
- Assignment of work shifts to the manager
- Definition of the maximum number of workers (target processes) the manager can run concurrently
- Optionally specializing the manager to run certain types of requests

Tip: It is easier to identify the optimum number of workers by being conservative initially, and defining additional workers later if needed (subject to availability of system resources).

Multiple managers can be run on multiple nodes using *Parallel Concurrent Processing*, as described below.

Concurrent Processing Architecture

In Concurrent Processing, programs are run as operating system background processes. These programs may be written using a variety of Oracle tools, programming languages for executables, or the host operating system scripting language.

As noted above, a concurrent program that runs in the concurrent manager's own operating system process is known as an immediate program. Immediate programs run as a function within the concurrent manager's program library. Examples include PL/SQL programs. In contrast, a concurrent program that runs in a child process of the concurrent manager process is known as a *spawned program*. Examples include SQL programs, SQL Loader programs, Oracle Reports programs, spawned C programs, and host language programs such as UNIX shell scripts or Windows command files.

All reports are run through the Concurrent Processing server manager via the `rwrun` executable, which spawns an in-process server. This is in contrast to earlier releases of Oracle E-Business Suite, which used the now-obsolete Reports server.

While C programs can be run as immediate programs, it is advisable to run them as spawned programs. This simplifies maintenance, without introducing any disadvantages.

A concurrent request has a life cycle, which consists of three or possibly four phases.

Concurrent Request Life Cycle

Phase	Activity
Pending	The request is waiting to be run
Running	The request is running
Completed	The request has finished
Inactive	The request cannot be run

A *concurrent program library* contains concurrent programs that can be called by a concurrent manager. An important example is the Oracle Application Object Library program library (FNDLIBR), which contains Oracle E-Business Suite immediate concurrent programs, and is assigned to the standard concurrent manager. Although each concurrent manager can only run immediate concurrent programs from its own concurrent program library, it can also run spawned or Oracle tool concurrent programs.

Various database tables are employed by the concurrent processing architecture:

Concurrent Processing Database Tables

Table	Content
FND_CONCURRENT_REQUESTS	Details of user requests, including status, start date, and completion date
FND_CONCURRENT_PROGRAMS	Details of concurrent programs, including execution method, whether the program is constrained, and whether it must be run alone.
FND_CONCURRENT_PROCESSES	Cross-references between concurrent requests and queues, and a history of concurrent manager processes
FND_CONCURRENT_QUEUES	Information about each of the concurrent manager queues

Caution: Do not update these tables manually. You can (subject to your organization's archiving requirements) periodically run the "Purge Concurrent Requests and/or manager data" program to prevent these tables growing too large. See *Oracle E-Business Suite Setup Guide* for details.

Concurrent Processing Operations

Because the Internal Concurrent Manager controls all the other managers, it must be running before any other manager can be activated. Once the ICM has been activated, it starts a Service Manager on each node that is enabled for concurrent processing. Acting as an agent of the ICM, the Service Manager starts the concurrent managers on its node, excluding any managers that have been deactivated, or that have no current work shift. The ICM can be activated and deactivated from the operating system prompt, or Oracle Applications Manager. It can also be deactivated (but not activated) from the Administer Concurrent Managers form.

When the ICM is initiated on UNIX, the $FND_TOP/bin/startmgr program is invoked. This calls $FND_TOP/bin/batchmgr, which then:

1. Starts a shell process.
2. Starts the ICM process using the command FNDLIBR, with startup parameters FND, CPMGR, and FNDCPMBR.

3. Creates log files (std.mgr and wnnn.mgr) in $APPLCSF/$APPLLOG.

Normally, startmgr is run by the user account that owns the application software (for example, applmgr). This account must have write privileges to the log and out directories where the log and output files respectively are written.

The ICM starts up a Service Manager on each node that is enabled for concurrent processing, by instructing the node's Applications Listener (which is dedicated to Concurrent Processing) to spawn a process running the Service Manager executable (FNDSM). The Applications Listener must be configured to source the Oracle E-Business Suite environment file before FNDSM is spawned. Following startup, the Service Manager acts as an agent of the ICM to start and stop concurrent managers on that node, according to their defined work shifts.

Note: The Service Manager is a component of the Generic Service Management (GSM) architecture rather than Concurrent Processing, although GSM and Concurrent Processing are closely integrated.

Concurrent manager processes on a specific node can be seen by running the UNIX commands:

```
ps –ef | grep FNDLIBR
```

```
ps –ef | grep FNDSM
```

The Service Manager PID seen in the output of the second command can then, if desired, be used to locate all concurrent manager and service processes on the node, since the Service Manager is the parent process for them:

```
ps –ef | grep <sm_pid>
```

On Windows, the Task Manager can be used to locate concurrent manager processes. An FNDLIBR process runs for the Internal Concurrent Manager and each standard manager. The ICM can be distinguished by additional details being displayed, including some of the parameters it was started with.

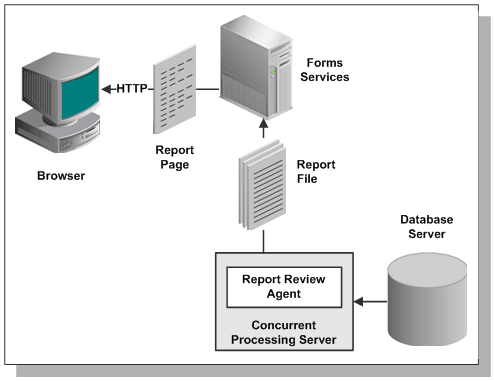
For every process that was successfully started at operating system level, the ICM inserts a row into FND_CONCURRENT_PROCESSES. It then updates the RUNNING_PROCESSES column to reflect the actual running processes as shown in FND_CONCURRENT_QUEUES.

Viewing Concurrent Processing Output

The output from a concurrent processing job goes through several stages before being displayed to the user.

1. The Concurrent Processing server communicates with the database server via Oracle Net.
2. The log or output file associated with a concurrent request is passed back to the *Report Review Agent*, also known as the *Web Review Agent*.
3. The Report Review Agent passes a file containing the entire report to the Forms services.
4. The Forms services pass the report back to the user's browser one page at a time.

Viewing Concurrent Processing Output



You can cater for your network capacity and report volume by using profile options to specify the maximum size of the files and pages that can be passed through the system.

Parallel Concurrent Processing

Parallel Concurrent Processing (PCP) allows concurrent processing activities to be distributed across multiple nodes in an Oracle Real Application Clusters (Oracle RAC) environment or similar cluster system. By distributing concurrent processing in this way, hardware resources can be fully utilized, maximizing throughput and providing resilience to node failure, while retaining a central point of control.

Parallel Concurrent Processing enables you to:

- Run concurrent processes on multiple nodes to improve concurrent processing throughput
- Continue running concurrent processes on the remaining nodes when one or more nodes fail
- Administer concurrent managers running on multiple nodes from any node in the cluster

One or more concurrent managers can be specified to run on one or more nodes, to best suit your processing needs and fully utilize available hardware resources.

Parallel Concurrent Processing is enabled by default, so PCP is always available for use in environments where one or more concurrent processing nodes exist.

Note: PCP does not require an Oracle RAC environment. Conversely, you do not have to use PCP in an Oracle RAC environment, although it typically makes sense to do so.

Managing Concurrent Processing

From the command line, two commands can be entered to control the Internal Concurrent Manager: **`startmgr`**, which starts the ICM; and **`concsub`**, which is used to stop or abort the ICM, or request the ICM to check on the operating system process for each manager. In addition, an AutoConfig-enabled environment provides a number of scripts for starting and stopping application tier services from the command line. The script for concurrent processing startup and shutdown is INST_TOP/admin/scripts/adcmctl.sh.

The various components of the concurrent processing system can be managed from various forms, such as *Concurrent Manager: Administer*, or from *Concurrent Managers* or *Concurrent Requests* under Oracle Applications Manager (OAM).

Additional Information: For details of setting up and managing concurrent processing, see *Oracle E-Business Suite Setup Guide*.

Concurrent Processing and Online Patching

The introduction of online patching with Release 12.2 has involved the introduction of a new concurrent program, ADZDPATCH, to ensure that concurrent programs cannot run during an online patching cycle if they are incompatible with this new patching model.

Full, practical details of the interaction between concurrent processing and online patching are provided in the Patching Procedures chapter of *Oracle E-Business Suite Maintenance Guide*

Application Tier Administration

Any application tier node can be used to carry out the following operations:

- **Applying Oracle E-Business Suite application patches**

  In general, *Oracle E-Business Suite patches* consist of files and scripts that update the file system and database objects. In Release 12.2, a single *unified (u) driver* file combines the features of the older *copy* (c), *database* (d), and *generate* (g) driver files used in earlier releases.

  You use the *adop* utility to perform these updates. This utility may also be used to apply cumulative patches such as *mini-packs* and *maintenance packs*.

- **Maintaining Oracle E-Business Suite data**

  Some features require updates to the tables and schemas they use. The *AD Administration* utility (*adadmin*) enables you to carry out this and various other file system and database maintenance tasks.

The Database Tier

The database tier contains the Oracle database server that stores and manages all the data maintained by Oracle E-Business Suite. This includes the various types of file in which the tables, indexes, and other database objects for your system physically reside, as well as the database executables. The database also stores the Oracle E-Business Suite online help information.

The database server communicates with the services and servers on the application tier, which mediate the communications between the database and the clients: there is no direct communication between the database and clients.

Using a Mixed Platform Architecture

The Oracle database server is sometimes available on platforms where Oracle E-Business Suite is not currently certified. In such a case, it may be possible to utilize a *mixed platform architecture*, where the database is installed on one platform and the application tier on another.

This type of deployment can enable the database to utilize the specific features offered by a particular platform. It can also allow the application tier to be managed in a more cost-effective way.

Additional Information: For up-to-date details of Release 12.2 support with mixed platform architectures, see Certify on My Oracle Support.

The Oracle E-Business Suite Technology Layer

The Oracle E-Business Suite technology layer lies between the Oracle E-Business Suite technology stack and the Oracle E-Business Suite product-specific modules. It provides features common to all Oracle E-Business Suite products.

Products in the Oracle E-Business Suite technology layer include:

- Oracle Applications DBA (AD)
- Oracle Application Object Library (FND)
- Oracle Applications Utilities (AU)
- Oracle Common Modules (AK)
- Oracle Workflow (WF)
- Oracle Web Applications Desktop Integrator (BNE)
- Oracle Alert (ALR)
- Oracle Application Framework (FWK)
- Oracle BI Publisher (XDO)

Oracle Applications DBA (AD)

The Oracle Applications DBA product provides a set of tools for administration of the Oracle E-Business Suite file system and database. AD tools are used for installing, upgrading, maintaining, and patching the Oracle E-Business Suite system.

The AD utilities include:

- **AD Administration** - Performs general maintenance tasks for Oracle E-Business Suite.
- **AD Merge Patch** - Merges multiple patches into a single, integrated patch.
- **AutoConfig** - Manages configuration changes to components that are specific to an Oracle E-Business Suite system. Configuration changes to technology components that are not specific to Oracle E-Business Suite are managed via the native tools for those components.
- **AutoPatch** - Applies patches and adds new languages and products to an Oracle E-Business Suite system.

  Important: In Oracle E-Business Suite Release 12.2, the actual patching utility you run is `adop` (AD online patching), rather than `adpatch` as in previous releases.

- **Rapid Clone** - Used to copy (clone) an Oracle E-Business Suite system.
- **Rapid Install** - Sets up a fully configured Oracle E-Business Suite system, including the latest certified technology stack and all patches, mini-packs, and other updates.

MetaData Services (MDS)

MetaData Services is an active data dictionary that enables you to define Oracle E-Business Suite components for HTML-based applications, and generate many of the Oracle E-Business Suite runtime characteristics.

The *Oracle Common Modules* can be used to develop inquiry applications for the HTML-based applications, without the need for any programming. They allow storage of language-translated labels for all the attributes on a transaction page, thus assisting with the provision of support for multiple languages.

Oracle Applications Utilities (AU)

The Applications Utilities (AU) component is used to maintain the Oracle E-Business Suite system.

AU hosts a collection of files copied from other products. This allows generating on-site classes of files such as Forms and reports. Generating forms or reports may require access to shared PL/SQL libraries, so these files are copied to AU_TOP as well.

Oracle Application Object Library (FND)

The Oracle Application Object Library is a key component of the Oracle E-Business Suite technology layer. It consists of a collection of reusable code, programs, and database objects that provides common functionality across all products.

Oracle Application Object Library offers many features to make system administration easier, such as security setup and maintenance, and management of concurrent processing. Using Application Object Library ensures that the processing of flexfields or the procedure for report submission, for example, does not vary from one product to another. Oracle Application Object Library also provides capabilities for developers to extend the operation of Oracle E-Business Suite by allowing the creation of custom programs that interact with the base modules.

End User Features

Oracle Application Object Library includes several features that help provide uniformity of function across the various Oracle E-Business Suite products.

Standard User Interface

Oracle Application Object Library supports the integration of Oracle E-Business Suite by providing standardized functionality and capabilities across all products so that the look and feel remains the same from product to product.

Shared Flexfield value sets

Flexfields allow the entry of certain important information to be standardized across all products. One example is the Accounting Flexfield, which is used by Financials products and Manufacturing products.

Standard Report Submission (SRS)

The procedure to submit a background report to the concurrent manager using SRS is the same, regardless of the product that owns the report. SRS takes advantage of shared flexfield value sets.

Online Help

The presentation of Online Help is also standardized across all Oracle E-Business Suite products.

Developer Features

Oracle Application Object Library provides many features for developers creating custom forms, reports, or programs that interface with Oracle E-Business Suite:

- The same coding and Graphical User Interface (GUI) standards used by Oracle E-Business Suite developers are available for custom development.
- Custom reports can be integrated into Standard Report Submission so that they can be submitted and monitored using the same procedures as other Oracle E-Business Suite reports, and developers can set up certain menus and responsibilities to access custom reports or standard objects.
- Flexfields used on custom forms can take advantage of existing flexfield capabilities such as value sets, validation, and security rules.
- Custom menus and responsibilities can be seamlessly integrated with Oracle E-Business Suite.

Features for System Administrators

Oracle Application Object Library provides many features to simplify administration of Oracle E-Business Suite, enabling the system administrator to carry out routine tasks quickly and easily. These features include:

- Registering new Oracle E-Business Suite users, and giving them access to only those Forms, functions, and reports they need to do their jobs.
- Deciding which users have access to each product, and within a product, which forms, functions, and reports a user can access.
- Monitoring what users do, and when, via comprehensive auditing capabilities.
- Setting user and system *profiles* to modify the look and behavior of Oracle E-Business Suite products; profiles can be set at site, application, responsibility, and user levels.
- Monitoring and controlling concurrent processing using interfaces such as Oracle Applications Manager (OAM).

Oracle Application Object Library Security

Oracle Application Object Library controls access to the data in Oracle E-Business Suite via user sign-ons and responsibilities. Each user must have a valid user name and password to gain access to Oracle E-Business Suite.

A *responsibility* is a level of authority in Oracle E-Business Suite that lets users access only those functions and data appropriate to their roles in the organization. For example, responsibilities may be used to allow access to a specific product, ledger, operating unit, or to a restricted list of windows, functions, reports, or groups of products.

Note that the Forms available from the navigation menus vary by responsibility. For example, the Purchasing User navigation menu does not include all the forms that are available to the Purchasing Superuser navigation menu.

When you install Oracle E-Business Suite, a standard user called SYSADMIN is created for you. Several default responsibilities are also created. Since the SYSADMIN sign-on is automatically assigned to the System Administrator responsibility, you can use SYSADMIN to create new user signons and assign them to responsibilities. You can also create any custom responsibilities you need.

Oracle Workflow (WF)

Oracle Workflow delivers a complete workflow management system that supports business process based integration. Its technology enables modeling, automation, and continuous improvement of business processes, routing information of any type according to user-defined business rules. Oracle Workflow also provides an infrastructure for the enterprise-wide communication of data related to defined business events, providing the capabilities needed to:

- Manage enterprise business processes that may span trading partners
- Support standard and personalized business rules
- Streamline and automate transaction flows
- Manage exceptions without manual intervention

Oracle Workflow lets you model and maintain your business processes using a graphical workflow builder. You can model and automate sophisticated business processes, defining processes that can loop, branch into parallel flows and rendezvous, decompose into sub-flows, branch on task results, time out, and more.

Acting as a *system integration hub*, Oracle Workflow can apply business rules to control objects and route them between applications and systems. It extends the reach of business process automation throughout an enterprise and beyond, to include any email user, web user, or system, enabling people to receive, analyze, and respond to *notifications* needing their attention. Users can respond to a notification via any standard email system or standard Web browser.

Workflow Components

Oracle Workflow Builder provides a graphical drag and drop process designer. You can create and evolve business processes to incorporate existing business practices between your organization and customers or suppliers, without modifying existing business processes and without changing applications code.

The *Workflow Engine*, embedded in the Oracle database, implements process definitions at runtime. The Workflow Engine monitors workflow states and coordinates the routing of activities for a process. Changes in workflow state, such as the completion of workflow activities, are signaled to the engine via a PL/SQL or Java API.

The Oracle Workflow *Business Event System* provides a workflow-enabled solution for your enterprise application integration requirements. The Business Event System is an application service delivered with Oracle Workflow that uses Oracle Advanced Queuing technology to communicate business events between systems. The Business Event System supports the following types of integration:

- Message-based point-to-point system integration
- System integration messaging hubs
- Distributed applications messaging

The Business Event System uses Oracle Advanced Queuing to propagate messages between communication points on systems, called *agents*, using a specified protocol. Events received from external systems are processed by an agent listener that runs on that agent's queue.

The *Oracle Workflow Event Manager* enables registration of significant business events for selected applications, including functions that generate the XML event messages. Users of those applications can register *subscriptions* on events that are significant to their systems, to take actions such as triggering custom code.

Features and Usage

A completed application transaction or event can initiate a workflow process by raising a business event or by calling a series of Workflow Engine APIs. The Workflow Engine drives through the process, performing all automated steps and calling the Notification System to deliver notifications for steps that involve human intervention.

You can review and respond to your business process notifications from one central page, known as the *worklist*, using a standard Web browser. This offers the flexibility to prioritize tasks and to define sort criteria, giving you the flexibility to organize your work the way you wish. For example, you can group notifications by type or subject, to avoid having to jump from one context to another. Alternatively, you can focus on time critical tasks first, sorting by priority or due date. Oracle Workflow is fully integrated with the Oracle E-Business Suite, providing the ability to drill down to any Oracle E-Business Suite or associated URL to view or complete a transaction.

When a business event occurs, the Workflow Event Manager executes any subscriptions registered on the event. For local events, the subscribing code can be executed synchronously, in the same database transaction as the code that raised the event, or asynchronously, deferring costly subscription processing to a later time, and thus allowing control to be returned more quickly to the calling application. Events can also be received asynchronously from external systems. Before producing the XML event message, the Event Manager minimizes processing by checking whether event information is required by subscriptions on the event.

Additional Capabilities

Workflow Engine event activities enable you to model business events within workflow processes. Event activities can be used to model content-based routing, transformations, error handling, and so on. A workflow process can be started or processed by an inbound message, and can send an outbound message or raise an event to the Event Manager. XML function activities give you access to event data content within workflow processes. Workflow processes based on business events give the greatest flexibility when implementing an integration solution. However, the Business Event System can also run independently of the Workflow Engine, to enable point-to-point messaging to be utilized.

You can perform complex transformations between different formats required for your business documents. Oracle Workflow allows you to apply a stylesheet to an XML event message. In addition, when queues are defined within the Business Event System, you specify the logic used to enqueue and dequeue messages. This logic, called a *queue handler*, can include transformations.

Oracle Web Applications Desktop Integrator (BNE)

Oracle Web Applications Desktop Integrator brings Oracle E-Business Suite data to the desktop, where you can use familiar spreadsheet data entry and modeling techniques in Microsoft Excel to complete your Oracle E-Business Suite tasks. You can create formatted spreadsheets on your desktop that allow you to download, view, edit, and create Oracle E-Business Suite data. To save time, you can use data entry shortcuts such as copying and pasting (or dragging and dropping) ranges of cells, or use Excel formulas to calculate amounts. You can combine speed and accuracy by invoking lists of values for fields within the spreadsheet.

After editing the spreadsheet, you can validate the data before uploading it to the Oracle E-Business Suite. Validation messages are returned to the spreadsheet, allowing you to identify and correct invalid data.

The fields that appear in the spreadsheet, their positions, and their default values can all be customized through the Oracle Web Applications Desktop Integrator layout functionality. A layout lets you create a more productive work environment by removing unnecessary fields from the spreadsheet, and by organizing the spreadsheet in a way that conforms to your practices.

Oracle E-Business Suite Desktop Integration Framework is a development tool that lets you define custom integrators for use with Oracle Web Applications Desktop Integrator. An integrator is a set of metadata that encapsulates all the information needed to integrate a particular Oracle E-Business Suite task with a desktop application. With Oracle E-Business Suite Desktop Integration Framework, you can define custom integrators for tasks in a standard Oracle E-Business Suite application that are not covered by seeded integrators, or for tasks in custom applications developed for your instance.

Oracle E-Business Suite Desktop Integration Framework provides a graphical user interface in which developers can define integrators and all associated supporting objects, without needing to work directly with the underlying Oracle Web Applications Desktop Integrator tables and APIs. Through this user interface, Oracle E-Business Suite Desktop Integration Framework reduces development time, increases developer productivity, and enhances ease of maintenance for the integrators you define.

Oracle Alert (ALR)

Oracle Alert (ALR) allows you to email system notifications to users when an exception or event has occurred. Some products are delivered with predefined alerts, which can be used to notify users about specified database exceptions as they occur, and perform routine tasks automatically according to a schedule you define.

For example, you can configure Oracle Alert to send an email to key database administrators when a tablespace in the Oracle E-Business Suite database does not have adequate free space.

Oracle BI Publisher (XDO)

Oracle BI Publisher (formerly known as Oracle XML Publisher) is a Java-based product based on the World Wide Web Consortium (W3C) *Extensible Stylesheet Language* (XSL). Specifically, Oracle BI Publisher utilizes the *XSL-FO* standard to transform XML data into a *formatting object* (FO). A formatting object contains both data and formatting information, and can be further transformed to an output format such as Portable Document Format (PDF).

Oracle BI Publisher uses *data definitions* and *templates* to produce output reports in the desired format. A data definition is a data source (or a combination of data sources) that either is XML or can produce XML. Examples include output from concurrent programs and Web services. A template is a report definition, which sets out how a report should look. The template layout can be user-specified. Supported templates currently include RTF, PDF Forms, and XSL.

Key features of Oracle BI Publisher include:

- Provides a template-based, easy-to-use publishing solution that enables end-users to develop and maintain report formats to suit specific business needs.
- Allows users to employ familiar desktop tools such as Adobe Acrobat, Microsoft Excel, and Microsoft Word to create and maintain reports in their preferred format, and then use Oracle BI Publisher to convert these documents to the XSL-FO format.
- Offers a variety of options for published documents, such as multiple output formats, multiple languages, and multiple delivery options.

Core Components

The core components of Oracle BI Publisher are a Java-based set of publishing tools accessible via Java APIs from Oracle E-Business Suite or any Java-based application.

- **FO Processor** - The formatting object that results from the application of the XSL-FO template to the XML data is used by the FO Processor to generate the required output document, for example as PDF.
- **PDF Document Merger** - The PDF Document Merger accepts XML data and a PDF document as input, and uses a mapping between the XML and the fields in the document to merge the data into the PDF document.
- **PDF Form Processor** - The PDF Form Processor is used to add attributes such as watermarks to a merged document.
- **RTF Template Parser** - Report templates can be created in the *Rich Text Format* (RTF) document format, and converted to an XSL-FO format that can be applied to XML data.

Template Manager

The Template Manager enables you to upload and maintain your templates and data sources. Key features include:

- **Data Source Registration** - Data sources that generate XML data can be registered with Oracle BI Publisher. These can be concurrent programs such as Oracle Reports, or BC4J objects that are able to generate XML data.
- **Template Registration** - Templates used to format report data must be registered with Oracle BI Publisher. They can then be applied to the report data to create the final published output. Supported template types include PDF, RTF, and XSL-FO.
- **Oracle BI Publisher APIs** - Oracle BI Publisher provides APIs to allow other Oracle E-Business Suite products to communicate directly with the underlying processes.

Network Topologies

Oracle E-Business Suite is designed to run on a local area network (LAN), with the database and application tiers co-located. This is because Oracle E-Business Suite processes generate significant network traffic between the database tier and the application tier, making performance very sensitive to the network characteristics of the connection, especially the latency. The best performance is therefore expected when the database and application tiers are located in the same data center.

However, you may be able to achieve adequate performance with the database tier and the application tier situated in separate data centers within the same geographical region if you are able to deploy a specialized network solution that guarantees high bandwidth and low latency between the data centers.

A deployment that separates the database tier and application tier over a wide area network (WAN) will not provide performance suitable for running EBS, and is not supported.

Additional Information: For a discussion of the effects of different network layers on load balancing, see My Oracle Support Knowledge Document 380489.1, *Using Load-Balancers with Oracle E-Business Suite Release 12*.

(#)

About Oracle (http://www.oracle.com/corporate/index.html) | Contact Us (http://www.oracle.com/us/corporate/contact/index.html) | Legal Notices (http://www.oracle.com/us/legal/index.html) | Terms of Use (http://www.oracle.com/us/legal/terms/index.html) | Your Privacy Rights (http://www.oracle.com/us/legal/privacy/index.html) | Cookie Preferences (#) | Ad Choices (https://www.oracle.com/legal/privacy/marketing-cloud-data-cloud-privacy-policy.html#12) |

Copyright © 2025, Oracle and/or its affiliates. (http://www.oracle.com/pls/topic/lookup?ctx=cpyr&id=en)