BEA WebLogic Server®

Understanding Domain Configuration

Version: 9.0 BETA
Revised: December 15, 2004

Copyright

Restricted Rights Legend

Trademarks or Service Marks

Contents

Managing Configuration Changes

CHAPTER 1

Introduction and Roadmap

This document describes WebLogic Server® domains and how they are configured. A *domain* is the basic administration unit for WebLogic Server. A domain consists of one or more WebLogic Server instances (and their associated resources) that you manage with a single Administration Server.

The following sections describe the contents and organization of this guide—*Understanding Domain Configuration*.

Document Scope and Audience

This document is written mainly for J2EE system architects, application developers, and system administrators who are developing or deploying Web-based applications on one or more WebLogic server domains.

The topics in this document are relevant during the design and development phases of a software project. This document does not address production phase administration, monitoring, or performance tuning topics. For links to WebLogic Server documentation and resources for these topics, see "Related Documentation" on page 1-2.

It is assumed that the reader is familiar with J2EE, basic concepts of XML, and general application management concepts.

Guide to this Document

- This chapter, "Introduction and Roadmap", introduces the purpose, organization, and context of this guide.
- Chapter 2, "Understanding WebLogic Server Domains," introduces WebLogic Server domains.
- Chapter 3, "Using WebLogic Tools to Configure a Domain," lists the various tools you can use to modify the configuration of a domain.
- Chapter 4, "Domain Configuration Files," describes the configuration and directories that maintain the on-disk representation of a domain and its contents.
- Chapter 5, "Managing Configuration Changes," describes change management features in WebLogic Server.

Related Documentation

For information about tools you can use to create and configure WebLogic Server domains, see:

- *Creating WebLogic Domains Using the Configuration Wizard*
- *WebLogic Scripting Tool*
- *Developing Manageable Applications with JMX*
- *WebLogic Server Command Reference*
- Administration Console Online Help

For information about other system administration tasks, see the System Administration documentation page, and in particular:

- *Designing and Setting Up WebLogic Server Environments*
- *Using WebLogic Server Clusters*

Samples and Tutorials

In addition to this document, BEA Systems provides the following code sample and tutorial that is relevant to domain configuration and administration:

- The BEA WebLogic Server Examples are optionally installed at `WL_HOME/samples/server/examples/src/examples`, where `WL_HOME` is the top-level directory of your WebLogic Server installation. These examples are also available from the Start menu on Windows machines. The Clustering example is described in the *BEA WebLogic Server Examples Clustering Guide*, which leads you through the process of creating and configuring a new cluster of server instances in a domain using the WebLogic Configuration Wizard and Administration Console.

New and Changed Domain Features in This Release

WebLogic Server 9.0 introduces several important changes to WebLogic Server domain configuration:

- "XML Schema for config.xml" on page 1-3
- "Domain Directory Structure" on page 1-4
- "Configuration Change Management" on page 1-4

XML Schema for config.xml

The on-disk representation of the configuration of WebLogic Server domains and instances has changed in this release. In previous releases, configuration information was persisted in a single XML repository file named `config.xml`, which was located by default in the `user_projects/domains/`*`domain_name`* folder. In this release of WebLogic Server, the `config.xml` file adheres to an XML Schema definition that can be used to validate the domain configuration file format. In addition, `config.xml` incorporates configuration information from other configuration files, which adhere to their own XML Schemas. In this release, `config.xml` is located by default in the `user_projects/domains/`*`domain_name`*`/config` folder, while the subsidiary configuration files to which the central `config.xml` file refers are located in subfolders under the `user_projects/domains/`*`domain_name`*`/config` folder. For more information, see Chapter 4, "Domain Configuration Files."

Domain Directory Structure

The directory structure that holds the on-disk representation of WebLogic Server domains has changed in this release. The parent directory for a domain is a directory named `domains`. Configuration information for the domain is kept in the `domains/domain_name/config` directory, and in subdirectories of the `config` directory. For more information, see "Domain Directory Contents" on page 4-4.

Configuration Change Management

WebLogic Server provides new features to manage changes to server configuration, which enable you to implement a secure, predictable means for distributing configuration changes in a domain. It also requires you first to obtain a lock in the Administration Console before you use the console to make any configuration changes.

The change management process in WebLogic Server loosely resembles a database transaction. The Administration Server maintains a separate, editable representation of the domain's configuration, which is referred to as the edit hierarchy. Server instances do not refer to the edit hierarchy. Instead, server instances use a read-only hierarchy to discover their configuration. To start the edit process, you obtain a lock on the edit hierarchy to prevent other people from making changes. When you finish making changes, you save and distribute them to all server instances in the domain. When you distribute changes, each server determines whether it can accept the change. If all servers are able to accept the change, they update their working configuration hierarchy and the change is completed.

The Administration Console now includes a region named the Change Center. When you use the Administration Console to make a configuration change, you must first obtain a lock by clicking Lock & Make Changes in the Change Center. Make any desired configuration changes, then in the Change Center either:

- click Activate Changes to accept the changes and distribute them to the server instances in the domain, or
- click Undo All Changes to roll back the changes and release the lock.

WebLogic Server controls configuration changes in generally the same manner, whether the changes are implemented using the Administration Console, the WebLogic Scripting Tool, or the Configuration Manager service and JMX APIs.

For more information, see Chapter 5, "Managing Configuration Changes."

CHAPTER 2

Understanding WebLogic Server Domains

The following sections introduce WebLogic Server domains and their contents:

What Is a Domain?

A WebLogic Server administration **domain** is a logically related group of WebLogic Server resources. Domains include a special WebLogic Server instance called the **Administration Server**, which is the central point from which you configure and manage all resources in the domain. Usually, you configure a domain to include additional WebLogic Server instances called **Managed Servers**. You deploy Web applications, EJBs, and other resources onto the Managed Servers and use the Administration Server for configuration and management purposes only.

Multiple Managed Servers can be grouped into **clusters**, which enable you to balance loads and provide failover protection for critical applications, while using a single Administration Server simplifies the management of the Managed Server instances.

Organizing Domains

How you organize your WebLogic Server installations into domains depends on your business needs. You can define multiple domains based on different system administrators' responsibilities, application boundaries, or geographical locations of the machines on which servers run. Conversely, you might decide to use a single domain to centralize all WebLogic Server administration activities.

Depending on your particular business needs and system administration practices, you might decide to organize your domains based on criteria such as:

- Logical divisions of applications. For example, you might have one domain devoted to end-user functions such as shopping carts and another domain devoted to back-end accounting applications.
- Physical location. You might establish separate domains for different locations or branches of your business.
- Size. You might find that domains organized in small units that can be managed more efficiently, perhaps by different system administrators. Contrarily, you might find that maintaining a single domain or a small number of domains makes it easier to maintain a consistent configuration.

A domain can consist of an Administration Server and one or more Managed Servers, or of a single standalone server that acts both as Administration Server and has application host.

- **Domain with Separate Managed Servers**: A simple production environment can consist of a domain with several Managed Servers that host applications, and an Administration Server to perform management operations. In this configuration, applications and resources are deployed to individual Managed Servers; similarly, clients that access the application connect to an individual Managed Server.

  Production environments that require increased application performance, throughput, or availability should configure two or more of Managed Servers as a cluster. Clustering allows multiple Managed Servers to operate as a single unit to host applications and resources. For more information about the difference between standalone and clustered Managed Servers, see "Managed Servers and Managed Server Clusters" on page 2-5.

- **Standalone Server Domain**: For development or test environments, you may want to deploy a single application and server independently from servers in a production domain. In this case, you can deploy a simple domain consisting of a single server instance that acts as an Administration Server that, and also hosts the applications you are developing. The

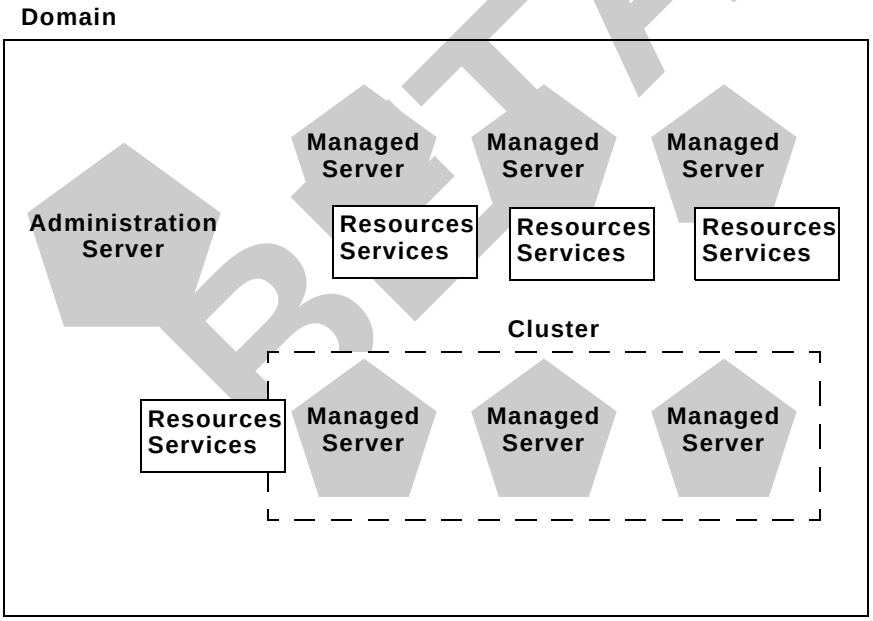
`wl_server` domain that you can install with WebLogic Server is an example of a standalone server domain.

Note: In production environments, BEA recommends that you deploy applications only on Managed Servers in the domain; the Administration Server should be reserved for management tasks.

Contents of a Domain

Although the scope and purpose of a domain can vary significantly, most WebLogic Server domains contain the components described in this section.

The following figure shows a production environment that contains an Administration Server, three standalone Managed Servers, and a cluster of three Managed Servers.



Administration Server

Each WebLogic Server domain must have one server instance that acts as the Administration Server. You use the Administration Server, programmatically or via the Administration Console, to configure all other server instances and resources in the domain.

Role of the Administration Server

Before you start the Managed Servers in a domain, you start the Administration Server. When you start a standalone or clustered Managed Server, it contacts the Administration Server for its configuration information. In this way, the Administration Server operates as the central control entity for the configuration of the entire domain.

When the Administration Server starts, it loads the `config.xml` file for the domain. Unless you specify another directory when you create a domain, `config.xml` is stored in:

`BEA_HOME/user_projects/domains/`*`mydomain`*`/config`

where *`mydomain`* is a domain-specific directory, with the same name as the domain. The `config.xml` file refers to other configuration files, which are located in subdirectories of the domain's `config` directory.

Each time the Administration Server starts successfully, a backup configuration file named `config-booted.jar` is created in the domain directory. In the unlikely event that the configuration files should be corrupted during the lifetime of the server instance, it is possible to revert to this previous configuration.

What Happens if the Administration Server Fails?

The failure of an Administration Server for a domain does not affect the operation of Managed Servers in the domain. If an Administration Server for a domain becomes unavailable while the server instances it manages—clustered or otherwise—are up and running, those Managed Servers continue to run. If the domain contains clustered server instances, the load balancing and failover capabilities supported by the domain configuration remain available, even if the Administration Server fails. If an Administration Server stops running while the Managed Servers in the domain continue to run, each Managed Server periodically attempts to reconnect to the Administration Server, at the interval specified by the `ServerMBean` attribute `AdminReconnectIntervalSecs`. By default, `AdminReconnectIntervalSecs` is ten seconds.

If an Administration Server fails because of a hardware or software failure on its host machine, other server instances on the same machine may be similarly affected. However, the failure of an Administration Server itself does not interrupt the operation of Managed Servers in the domain. Also, you can start a Managed Server even if the Administration Server is not running. In this

case, the Managed Server uses a local copy of its configuration files for its starting configuration and then periodically attempts to connect with the Administration Server. When it does connect, it synchronizes its configuration state with that of the Administration Server.

For instructions on re-starting an Administration Server, see *Managing Server Startup and Shutdown*.

Managed Servers and Managed Server Clusters

In a domain, server instances other than the Administration Server are referred to as Managed Servers. Managed Servers host the components and associated resources that constitute your applications—for example, JSPs and EJBs. When a Managed Server starts up, it connects to the domain's Administration Server to obtain configuration and deployment settings.

Note: Managed Servers in a domain can start up independently of the Administration Server if the Administration Server is unavailable. See "Avoiding and Recovering From Server Failure" in *Managing Server Startup and Shutdown* for more information.

Two or more Managed Servers can be configured as a WebLogic Server *cluster* to increase application scalability and availability. In a WebLogic Server cluster, most resources and services are deployed identically to each Managed Server (as opposed to a single Managed Server), enabling failover and load balancing. To learn which component types and services can be clustered—deployed to all server instances in a cluster—see "Understanding WebLogic Server Clustering" in *Using WebLogic Server Clusters.*

You can create a non-clustered Managed Server and add it to a cluster by configuring pertinent configuration parameters for the server instance and the cluster. You can remove a Managed Server from a cluster by re-configuring the parameters appropriately. The key difference between clustered and non-clustered Managed Servers is support for failover and load balancing. These features are available only in a cluster of Managed Servers.

Your requirements for scalability and reliability drive the decision on whether or not to cluster Managed Servers. For example, if your application is not subject to variable loads, and potential interruptions in application service are acceptable, clustering may be unnecessary.

For more information about the benefits and capabilities of a WebLogic Server cluster, see "Understanding WebLogic Server Clustering" in *Using WebLogic Server Clusters.* A single domain can contain multiple WebLogic Server clusters, as well as multiple Managed Servers that are not configured as clusters.

Resources and Services

In addition to the Administration Server and Managed Servers, a domain also contains the resources and services required by Managed Servers and hosted applications deployed in the domain.

The domain configuration includes information about the networked computer environment in which the domain runs, such as:

- Machine definitions identify a particular, physical piece of hardware. A machine definition is used to associate a computer with the Managed Servers it hosts. This information is used by Node Manager in restarting a failed Managed Server, and by a clustered Managed Server in selecting the best location for storing replicated session data. For more information about Node Manager, see Using Node Manager to Control Servers in *Designing and Configuring WebLogic Server Environments*.
- Network channels, an optional resource that can be used to define default ports, protocols, and protocol settings. After creating a network channel, you can assign it to any number of Managed Servers and clusters in the domain. For more information, see Configuring Network Resources in *Designing and Configuring WebLogic Server Environments*.

The domain configuration also includes information about resources and services associated with applications hosted on the domain. Examples of these resources and services include:

- application components, such as EJBs
- connectors
- JDBC connection pools
- JMS servers
- startup classes

Resources and services can be limited to one or more Managed Servers in the domain, rather than being available to the domain as a whole. You can deploy resources and services to selected Managed Servers or to a cluster.

Domain Restrictions

A WebLogic Server environment may consist of a single domain that includes all the Managed Servers required to host applications, or multiple domains. You might choose to create multiple domains divided by organizational units, system administrator responsibilities, application boundaries, or other considerations. In designing your domain configuration, note the following restrictions:

- Each domain requires its own Administration Server for performing management activities. When you use the Administration Console to perform management and monitoring tasks, you can switch back and forth between domains, but in doing so, you are connecting to different Administration Servers.
- All Managed Servers in a cluster must reside in the same domain; you cannot split a cluster over multiple domains.
- All Managed Servers in a domain must run the same version of the WebLogic Server software. The Administration Server may run either the same version as the Managed Servers in the domain, or a later service pack.

You cannot share a configured resource or subsystem between domains. For example, if you create a JDBC connection pool in one domain, you cannot use it with a Managed Server or cluster in another domain. Instead, you must create a similar connection pool in the second domain.

CHAPTER 3

Using WebLogic Tools to Configure a Domain

WebLogic includes a variety of tools you can use to create, modify, or replicate a domain configuration. These tools include the following:

- Domain Configuration Wizard—The Domain Configuration Wizard is the recommended tool for creating a new domain or cluster. For information about using the Domain Configuration Wizard, see *Creating WebLogic Domains Using the Configuration Wizard*.
- WebLogic Server Administration Console—The Administration Console is a graphical user interface (GUI) to the Administration Server. The Administration Console is described in the *Administration Console Online Help*.
- WebLogic Scripting Tool (WLST)—You can use this command-line scripting interface to initiate, manage, and persist WebLogic Server configuration changes. The WebLogic Scripting Tool is described in *WebLogic Scripting Tool*.
- WebLogic Server Application Programming Interface (API)—You can write a program to modify configuration attributes using the API provided with WebLogic Server. The JMX API is described in *Developing Manageable Applications with JMX*.
- WebLogic Server command-line utility—This utility allows you to create scripts to automate domain management. For more information about this utility, see *WebLogic Server Command Reference*.

For most of these methods, the Administration Server for a domain must be running to modify the domain configuration. However, you can use WLST to make configuration changes to a domain even if its Administration Server is not running. In this case, changes made with WLST won't take effect until the Administration Server and Managed Servers are restarted.

CHAPTER 4

Domain Configuration Files

This section describes how a domain is represented in the file system. It includes the following sections:

Overview of Domain Configuration Files

WebLogic Server management and configuration services are accessed with the Java Management Extensions (JMX) API. The configuration of a domain is stored in the configuration directories under the domain directory. The files in these configuration directories act as a persistent store for the managed objects that WebLogic Server creates and modifies during its executing using the JMX API. The purpose of `config.xml` is to store changes to managed configuration objects so that they are available when WebLogic Server is restarted.

The central configuration file for the domain is *`domain_name`*`/config/config.xml` file. It specifies the name of the domain and the configuration parameter settings for each server instance, cluster, resource, and service in the domain. The configuration for some major subsystems of the domain is stored in subdirectories of the *`domain_name`*`/config` directory in separate configuration files that are incorporated by reference into the central `config.xml` file.

The domain directory also contains default script files that you can use to start the domain's Administration Server and Managed Servers.

config.xml

The central configuration file for the domain is the `/domains/domain_name/config/config.xml` file. It specifies the name of the domain and the configuration parameter settings for each server instance, cluster, resource, and service in the domain.

The `config.xml` file complies with an XML Schema whose URL is `http://www.bea.com/ns/weblogic/config`. This schema is located in a JAR file in the file system at `BEA_HOME/weblogic90/server/lib/schema/configuration-binding.jar` and within the JAR file at `META-INF/schemas/schema-0.xsd`. The XML Schema enables the use of XML editing tools to modify and validate the `config.xml` file.

Editing Configuration Files

In most circumstances, you should not directly modify the `config.xml` file or the other configuration files. Instead, use the Administration Console or one of the other tools listed in Chapter 3, "Using WebLogic Tools to Configure a Domain," to modify the domain's configuration. The configuration changes will then be reflected in the configuration files.

Directly modifying configuration files may be appropriate if you choose to place configuration files and other components of your installation under source control, managing them using WLST.

Warning: You cannot edit configuration files while WebLogic Server is executing, since WebLogic Server rewrites the files periodically. Your changes will be lost and, depending on your platform, you could cause WebLogic Server failures.

Since the WebLogic Server configuration files are well-formed XML files, it is possible to script certain repetitive changes using an XML parser application such as Apache Xerces, or JDOM. Be sure to test any scripts you create thoroughly and always make a backup copy of each configuration file before you make any changes to it.

Subsidiary Configuration Files

In previous releases, the `config.xml` file was a repository for all configuration information. In this release, several WebLogic Server subsystems are configured in subsidiary configuration files that are referred to by the central `config.xml` file. These subsidiary configuration files reside in subdirectories of the `/domains/domain_name/config` directory. For more information about these subsidiary configuration files, see "Domain Directory Overview" on page 4-3 and "Domain Directory Contents" on page 4-4.

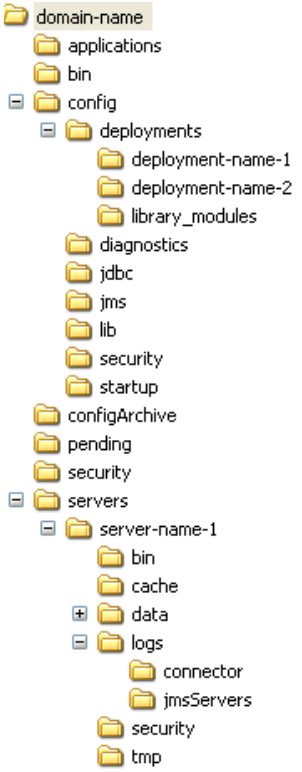
Configuration File Archiving

WebLogic Server makes backup copies of the configuration files. This facilitates recovery in cases where configuration changes need to be reversed or the unlikely case that configuration files become corrupted. When the Administration Server starts up, it saves a JAR file named `config-booted.jar` that contains the configuration files. When you make changes to the configuration files, the old files are saved in the `configArchive` directory under the domain directory, in a JAR file with a sequentially-numbered name like `config-1.jar`.

Domain Directory Overview

Figure 4-1 is an overview of the domain directory tree hierarchy. The *`domain-name`*, *`deployment-name`*, and *`server-name`* directory names are not literal, but are replaced in any actual case with whatever specific names are appropriate; the other directory names are literal. This overview shows only directories, not files within the directories. In any actual particular domain directory tree, this whole hierarchy might not be present.

Figure 4-1 Domain Directory Structure



Domain Directory Contents

This section describes the contents of the domain directory and its subfolders. Directory names in *italics* are not literal, but are replaced with whatever specific names is appropriate; names not in italics are literal.

domain-name

The name of this directory is the name of the domain.

applications

This directory provides a quick way to deploy applications in a development server. When the WebLogic Server instance is running in development mode, it automatically deploys any applications or modules that you place in this directory.

The files you place in this directory can be:

- A J2EE application
- An EAR file
- A WAR, EJB JAR, RAR, or CAR archived module
- An exploded archive directory for either an application or a module

bin

This directory contains scripts that are used in the process of starting and stopping the Administration Server and the Managed Servers in the domain. It can optionally contain other scripts of domain-wide interest, such as scripts to start and stop database management systems, full-text search engine processes, etc. For more information, see *Managing Server Startup and Shutdown*.

config

This directory contains the current configuration and deployment state of the domain. The central domain configuration file, `config.xml`, resides in this directory

config/deployments

This directory holds the domain's deployed applications.

config/deployments/library_modules

This directory holds library modules; that is, anything found in this directory will automatically be registered as a library module.

config/deployments/*deployment-name-1*

This directory contains one application, or deployable module. It contains a subhierarchy that can contain an archive file (EAR or WAR), a deployment plan file, external descriptors, and so on.

config/diagnostics

This directory contains system modules for instrumentation in the WebLogic Diagnostic Service. For more information, see *Understanding the WebLogic Diagnostic Service*.

config/jdbc

This directory contains system modules for JDBC: global JDBC modules that can be configured directly from JMX (as opposed to JSR-88). For more information, see *Database Connectivity (JDBC)*.

config/jms

This directory contains system modules for JMS: global JMS modules that can be configured directly from JMX (as opposed to JSR-88). For more information, see *Messaging and WebLogic Server (JMS)*.

config/nodemanager

This directory holds configuration information for connection to the Node Manager. For more information, see *Using Node Manager to Control Servers* in *Designing and Configuring WebLogic Server Environments*.

config/security

This directory contains system modules for the security framework. It contains one security provider configuration extension for each kind of security provider in the domain's current realm. For more information, see *Understanding WebLogic Security*.

config/startup

This directory contains system modules that contain startup plans. Startup plans are used to generate shell scripts that can be used as part of server startup.

configArchive

This directory contains a set of JAR files that save the domain's configuration state. Just before pending changes to the configuration are activated, the domain's existing configuration state, consisting of the `config.xml` file and the other related configuration files, is saved in a versioned JAR file with a name like `config.jar#1`, `config.jar#2`, etc.

The maximum number of versioned JAR files to be kept is specified by the `archiveConfigurationCount` property of `DomainMBean`. Once this maximum number is reached, the oldest conversion archive is deleted before a new one is created.

lib

Any JAR files you put in this directory are added to the system classpath of each server instance in the domain when the server's Java virtual machine starts.

pending

This directory contains domain configuration files representing configuration changes that have been requested, but not yet activated. Once the configuration changes have been activated, the configuration files are deleted from this directory. For more information, see "Managing Configuration Changes" on page 5-1.

security

This directory holds those security-related files that are the same for every WebLogic Server instance in the domain:

- `SerializedSystemIni.dat`

This directory also holds security-related file that are only needed by the domain's Administration Server:

- `DefaultAuthorizerInit.ldift`
- `DefaultAuthenticatorInit.ldift`
- `DefaultRoleMapperInit.ldift`

For more information, see *Understanding WebLogic Security*.

servers

This directory contains one subdirectory for each WebLogic Server instance in the domain.

servers/*server-name*

This directory is the server directory for the WebLogic Server instance with the same name.

servers/*server-name*/bin

This directory holds executable or shell files that can be or must be different for each server. The server environment script (`setServerEnv.sh` or `setServerEnv.cmd`) is an example of a file that resides here because it can differ from one WebLogic Server instance to the next, for example, depending on whether the server instance has its own startup plan.

servers/*server-name*/cache

This directory holds directories and files that contain cached data. By "cached" here we mean that the data is a copy, possibly in a processed form (compiled, translated, or reformatted), of other data.

servers/*server-name*/cache/EJBCompilerCache

This directory is a cache for compiled EJBs.

servers/*server-name*/data

This directory holds files that maintain persistent per-server state used to run the WebLogic Server instance, other than security state, as opposed to temporary, cached or historical information. Files in this directory are important data that must be retained as the WebLogic Server instance is brought up, is brought down, crashes, restarts, or is upgraded to a new version.

servers/*server-name*/data/ldap

This directory holds the embedded LDAP database. The runtime security state for the WebLogic Server instance is persisted in this directory.

servers/*server-name*/data/store

This directory holds JMS persistent stores. For each persistent store, there is a subdirectory that holds the files that represent the persistent store. The name of the subdirectory is the name of the persistent store. By convention there is one store named `default`.

servers/*server-name*/logs

This directory holds logs and diagnostic information. This information is historical in nature. It is not crucial to the operation of the server, and can be deleted (while the WebLogic Server instance is down, at least) without affecting proper operation. However, the information can be quite useful for debugging or auditing purposes and should not be deleted without good reason.

servers/*server-name*/logs/diagnostic_images

This directory holds information created by the Server Image Capture component of the WebLogic Diagnostic Service. For more information, see *Understanding the WebLogic Diagnostic Service*.

servers/*server-name*/logs/jmsServers

This directory contains one subdirectory for each JMS server in the WebLogic Server instance. Each such subdirectory contains the logs for that JMS server. The name of the subdirectory is the name of the JMS server.

servers/*server-name*/logs/connector

This directory is the default base directory for connector module (JCA ResourceAdapter) logs.

servers/*server-name*/security

This directory holds security-related files that can be or must be different for each WebLogic Server instance. The file `boot.properties` is an example of a file that resides here because it can differ from one server to the next. This directory also maintains files related to SSL keys.

servers/*server-name*/tmp

This directory holds temporary directories and files that are created while a server instance is running. Files in this directory must be left alone while the server is running, but may be freely deleted when the server instance is shut down.

CHAPTER 5

Managing Configuration Changes

To provide a secure, predictable means for distributing configuration changes in a domain, WebLogic Server imposes a change management process that loosely resembles a database transaction. The configuration of a domain is represented on the file system by a set of XML configuration files, centralized in the `config.xml` file, and at runtime by a hierarchy of Configuration MBeans. When you edit the domain configuration, you edit a separate hierarchy of Configuration MBeans that resides on the Administration Server. To start the edit process, you obtain a lock on the edit hierarchy to prevent other people from making changes. When you finish making changes, you save the changes. The changes do not take effect, however, until you activate them, distributing them to all server instances in the domain. When you activate changes, each server determines whether it can accept the change. If all servers are able to accept the change, they update their working configuration hierarchy and the change is completed.

Note that WebLogic Server's change management process applies to changes in domain and server configuration data, not to security or application data.

For more detailed information about how configuration changes are carried out through JMX and Configuration MBeans, see *Understanding WebLogic Server MBeans* in *Developing Manageable Applications with JMX*.

As described in Chapter 3, "Using WebLogic Tools to Configure a Domain," you can use a variety of different WebLogic Server tools to make configuration changes:

- Administration Console
- WebLogic Scripting Tool
- JMX APIs

Whichever tool you use to make configuration changes, WebLogic Server handles the change process in basically the same way.

The following sections describe configuration change management:

Change Management in the Administration Console

The WebLogic Administration Console centralizes the configuration change management process in the Change Center region:

Figure 5-1 Change Center

If you want to use the Administration Console to make configuration changes, you must first click the Lock & Edit button in the Change Center. When you click Lock & Edit, you obtain a lock on the editable hierarchy of Configuration MBeans for all servers in the domain (the edit tree).

As you make configuration changes using the Administration Console, you click Save (or in some cases Finish) on the appropriate pages. This does not cause the changes to take effect immediately; instead, when you click Save, you are saving the change to the edit tree and to the *domain-name*`/pending/config.xml` file and related configuration files. The changes take effect when you click Activate Changes in the Change Center. At that point, the configuration changes are distributed to each of the servers in the domain. If the changes are acceptable to each

of the servers, then they take effect. If any server cannot accept a change, then all of the changes are rolled back from all of the servers in the domain. The changes are left in a pending state; you can then either edit the pending changes to resolve the problem or revert the pending changes.

Configuration Change Management Process

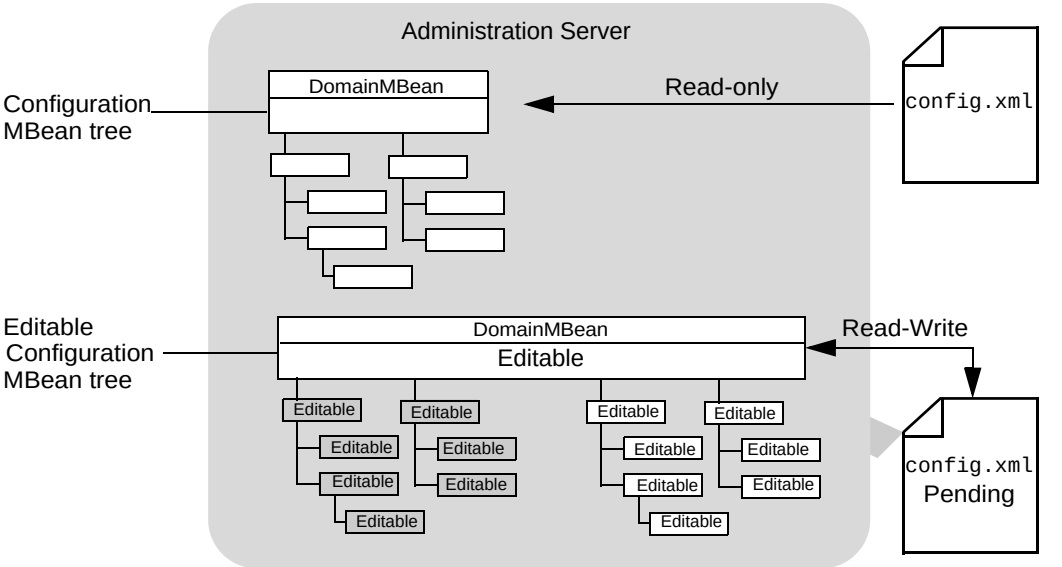
The following steps describe the process in detail, starting from when you first boot the domain's Administration Server:

1. When the Administration Server starts, it reads the domain's configuration files, including `config.xml` file and any subsidiary configuration files referred to by the `config.xml` file and uses the data to instantiate the following MBean trees:
   - A read-only tree of Configuration MBeans that contains the current configuration of resources that are on the Administration Server.
   - An editable tree of all Configuration MBeans for all servers in the domain.

   Note: The Administration Server also instantiates a Runtime MBean tree and a DomainRuntime MBean tree, but these are not used for configuration management.
2. To initiate a configuration change, you do the following:
   a. Obtain a lock on the current configuration.
   b. Make any changes you desire, using the tool of your choice (the Administration Console, WLST, the JMX APIs, etc.)
   c. Save your changes to a pending version of the `config.xml` file.
3. The Configuration Manager service saves all data from the edit MBean tree to a separate set of configuration files in a directory named `pending`. See Figure 5-2.

   The `pending` directory is immediately below the domain's root directory. For example, if your domain is named `mydomain`, then the default pathname of the pending `config.xml` file is `mydomain/pending/config.xml`.

Figure 5-2 The Administration Server's Pending config.xml File



4. Make additional changes or undo changes that you have already made.
5. When you are ready, activate your changes in the domain, using the Activate Changes button in the Administration Console's Change Center or using the `ConfigurationManagerMBean`.

   When you activate changes (see Figure 5-3):

   a. For each server instance in the domain, the Configuration Manager service copies the pending configuration files to a `pending` directory in the server's root directory.

      If a Managed Server shares its root directory with the Administration Server, `ConfigurationManagerMBean` does not copy the pending configuration files; the Managed Server uses the Administration Server's pending file.

   b. Each server instance compares its current configuration with the configuration in the pending file.

   c. Subsystems within each server vote on whether they can consume the new configuration.

      If any subsystem indicates that it cannot consume the changes, the entire activation process is rolled back and the `ConfigurationManagerMBean` emits an exception. You can modify your changes and retry the change activation, or you can abandon your

lock, in which case the edit Configuration MBean tree and the pending configuration files are reverted to the configuration in the read-only Configuration MBean tree and configuration files.

d. If all subsystems on all servers can consume the change, the Configuration Manager service replaces the read-only configuration files on each server instance in the domain with the pending configuration files.

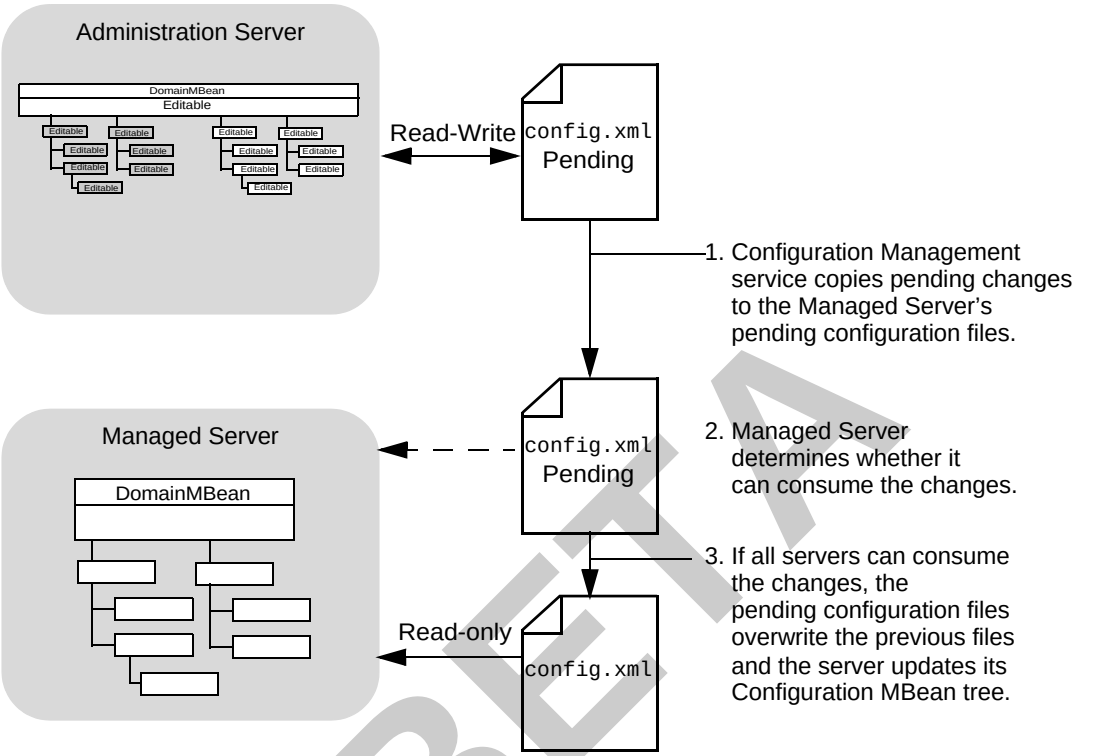
e. Each server instance updates its beans and its read-only Configuration MBean tree according to the changes in the new configuration files.

f. The pending configuration files are then deleted from the `pending` directory.

6. You can retain your lock to make additional changes or release it so that others can update the configuration. You can configure a timeout period that causes the Configuration Manager service to abandon a lock.

Note: The configuration change lock does not prevent you from making conflicting configuration edits using the same administrator user account. For example, if you obtain a configuration change lock using the Administration Console, and then use the WebLogic Scripting Tool with the same user account, you will access the same edit session that you opened in the Administration Console and you will not be locked out of making changes with the Scripting Tool. Since this can lead to confusion and conflicting configuration changes, this is not a recommended practice. You can reduce the risk that such a situation might occur by maintaining separate administrator user accounts for each person with an administrative role. Similar problems can still occur, however, if you have multiple instances of the same script using the same user account.

Figure 5-3 Activating Changes in Managed Servers



Resolving Change Conflicts

In the event that you have saved more than one change set without activating them and one change would invalidate a prior change, the Change Management service requires you to manually resolve the invalidation before it will save your changes.

Configuration Management State Diagram

The Configuration Management service follows a series of states, which are described in Figure 5-4.

Figure 5-4 Configuration Management State Diagram