Veritas High Availability Agent 5.0 for WebLogic Server Installation and Configuration Guide

HP-UX, Linux, Solaris

N19297C

Veritas High Availability Agent 5.0 for WebLogic Server Installation and Configuration Guide

Symantec Corporation
20330 Stevens Creek Blvd.
Cupertino, CA 95014
www.symantec.com

Third-party legal notices

Technical support

For technical assistance, visit http://support.veritas.com and select phone or email support. Use the Knowledge Base search feature to access resources such as TechNotes, product alerts, software downloads, hardware compatibility lists, and our customer email notification service.

Contents

Chapter 1 Introducing the Veritas Agent for WebLogic Server

What's new in this agent 7
Supported software 8
About the agent for WebLogic 8
  WebLogic Server categories 9
  Agent operations 10
    Online operation 10
    Offline operation 10
    Monitor operation 10
    Clean operation 11

Chapter 2 Installing the Veritas Agent for WebLogic Server

Prerequisites for installing the agent for WebLogic 13
  Prerequisites for installing the agent in a VCS environment 13
  Prerequisites for installing the agent in a VAD environment 13
  About the ACC library 14
Upgrading the Veritas agent for WebLogic Server 14
Installing the agent in a VCS environment 15
Installing the agent in a VAD environment 16

Chapter 3 Configuring the Veritas Agent for WebLogic Server

Importing the agent types files 17
Sample agent type definition 19
Agent attributes 20
  Required attributes 20
  Optional attributes 24
Uniquely identifying WebLogic Server instances 26
Delaying Managed Server startup process 26
Executing a custom monitor program 27

Chapter 4 Clustering WebLogic Servers

Overview of the clustering process 29

Chapter 5 Uninstalling the Veritas Agent for WebLogic Server

Chapter 6 Troubleshooting the Veritas Agent for WebLogic Server

Testing WebLogic Servers outside a cluster 35
Using correct software and operating system versions 37
Meeting prerequisites 38
Configuring WebLogic resources 38
Reviewing log files 38
    Using WebLogic agent log files 38
    Using cluster log files 39
    Using WebLogic Server log files 39
        Using trace level logging 39

Appendix A Sample Configurations

Sample service group configuration 41
    Service Group dependencies 43
    Sample configuration in a VCS environment 43
    Sample configuration in a VAD environment 44

Index 45

Introducing the Veritas Agent for WebLogic Server

Welcome to the Veritas High Availability Agent for WebLogic Server. This guide describes the agent, agent operations, and agent attributes. The guide assumes the reader understands the primary components and basic functionality of Veritas Cluster Server or Veritas Application Director. It also assumes a basic understanding of the WebLogic Server architecture and its configuration options.

What's new in this agent

- Integrated with the enhanced version of ACC library, that includes numerous fixes for improved functionality.
- Fixed the default `csh` shell issue. Previously, if the user had set the `csh` shell as default, the agent was unable to run the `start` command in the background and was unable to redirect the output of the agent operations.
- Fixed issue that arose with the SecondLevelMonitor attribute when users used the `csh` shell as default. The users previously could not run the second level check if the file specified in the EnvFile attribute contained `csh` syntax. The operation failed and reported errors to the cluster engine log.
- Fixed the negative timeout value that SecondLevelMonitor used when online.
- Fixed issue that arose during first level monitor check. Previously, if the first level monitor check failed, the agent was unable to bring the resource offline. Instead, the agent reported the resource state as UNKNOWN.
- Fixed issue that arose due to the format of the ListenAddressPort attribute. Previously, if the format of ListenAddressPort was `IPAddress:Port`, the agent was unable to interpret the value correctly.

Supported software

The Veritas High Availability Agent for WebLogic Server is supported in the following environments:

Environment	Supported Versions
Veritas Cluster Server	HP-UX—VCS 4.1, 5.0
	Linux—VCS 4.0, 4.1, 5.0
	Solaris—VCS 4.0, 4.1, 5.0
Veritas Application Director	Linux—VAD 1.0
	Solaris—VAD 1.0
ACC Library	5.0
Operating Systems	Red Hat Enterprise Linux 3.0, 4.0 on Intel
	Solaris 8, 9 on SPARC
	HP-UX 11i V2 on PA-RISC
WebLogic Server	9.0, 9.1

About the agent for WebLogic

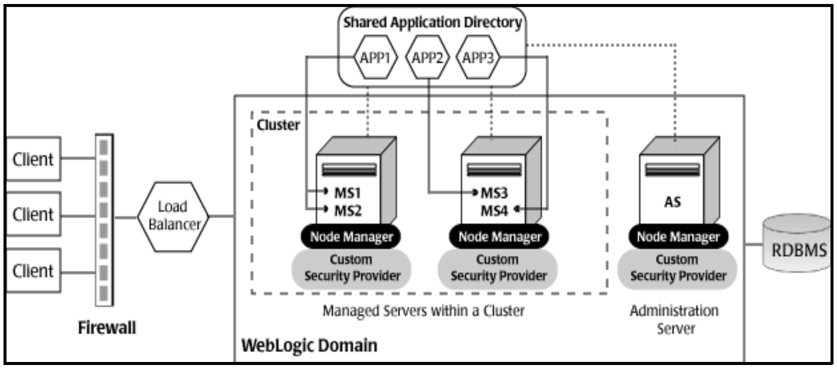
The Veritas High Availability Agent for WebLogic Server consists of resource type declarations and agent executables. The agent is responsible to bring a WebLogic Server (WLS) online, to monitor the processes and state of the WLS on all nodes in the cluster, to detect failure of a WLS and to shutdown a WLS when directed or when circumstances indicate that a failover is required. The agent executables are logically organized into separate agent entry points to include: *online*, *offline*, *monitor*, and *clean*.

WebLogic Server categories

WebLogic Servers fall into two categories: *Administrative* and *Managed*. The Administrative Server provides a central point from which you can manage the domain, and it provides access to WebLogic Server administration tools [WLS05: *Introduction to BEA WebLogic Server and BEA WebLogic Express*, July 2005]. All other servers are considered Managed Servers.

A *Node Manager* is a WebLogic Server utility that enables you to start, shut down, and restart Administration Server and Managed Server instances from a remote location.

This agent exclusively uses the WebLogic Node Manager to manage both Administrative and Managed Servers. Therefore, you must configure a WebLogic resource as a WebLogic Node Manager, prior to configuring resources for either Administrative or Managed Servers. If you do not configure a Node Manager resource, this agent is unable to manage Administrative or Managed WebLogic Servers.



The WebLogic agent supports both Administrative and Managed Servers. The agent recognizes the *startup* server dependency that exists between Managed and Administrative Servers and provides the cluster administrator with the choice of enforcing or not enforcing this startup restriction. In like manner, the agent is WebLogic Cluster agnostic. In other words, this agent can provide clustering services for stand-alone WebLogic Servers and can support Managed Servers that participate in a WebLogic Cluster.

Agent operations

The following sections elaborate the steps performed in each agent operation.

Online operation

The online operation performs these tasks:

- Performs a preliminary check to ensure that the resource is fully offline.
- Checks the value of the ServerRole attribute set for the resource. If the value of the attribute is *Managed*, the online operation may delay the Managed Server startup process till the *Administrativer* Server is initialized. For details, refer to “Delaying Managed Server startup process” on page 26.
- Starts the server instance. To start the server instance, the agent uses the `wlst.sh` utility that BEA provides.
- Ensures that the instance is up and running successfully. The operation uses the wait period that the OnlineTimeout attribute specifies, to enable the instance to initialize fully before allowing the monitor operation to probe the newly running server instance.

Offline operation

The offline operation performs these tasks:

- Performs a preliminary check to ensure that the resource running the WebLogic instance, is not already offline.
- Stops the server instance.
  - For Administrative and Managed Servers, the operation uses the `wlst.sh` utility that BEA provides.
  - For Node Manager, the operation sends a KILL signal to the `nodemanager` process.
- Ensures that the resource is given enough time to go offline successfully. The operation uses a wait period that the OfflineTimeout attribute specifies, to allow the WebLogic Server instance to complete the offline sequence before allowing further probing of the resource.

Monitor operation

The monitor operation performs these tasks:

- Conducts a first level check on the WebLogic Server instance to ensure that the processes are running smoothly. The first level check performs a socket connection for the instance.

- For Administrative and Managed Servers, the first level check performs a socket connection using the value of the ListenAddressPort attribute. If the socket connection is successful, the first level check is considered successful for the WebLogic instance.
- For a Node Manager, the first level check performs a socket connection using the value of the nmListenAddressPort attribute. If the socket connection is successful, the operation reviews the contents of the `pid` file that the agent creates for the online Node Manager instance. If a valid `pid` file is not available, the agent creates a correct `pid` file for the running Node Manager instance.

- Depending on what settings you make, the monitor operation can conduct a second level check on the server instance.
  The second level check uses the `wlst.sh` scripting utility to attempt to connect to the server that is running the WebLogic instance.
  - If the value of the ServerRole attribute is **`NodeManager`**, the agent attempts to connect to the server using the `nmConnect()` API.
  - If the value of the ServerRole attribute is **`Managed`** or **`Administrative`**, the agent attempts to connect to the server using the `connect()` API.

  If the connection is successful, the second level check is considered successful for the WebLogic instance.
- Depending upon the MonitorProgram attribute, the monitor operation can perform a customized check using a user-supplied monitoring utility. For details about executing a custom monitor program, refer to “Executing a custom monitor program” on page 27.

Clean operation

The clean operation performs these tasks:

- Attempts to gracefully shut down the Administrative or Managed Servers.
- Generates a list of running processes that are relevant to the WebLogic Server instance. The ServerName and DomainName attributes determine the list of processes running for the instance.
- Kills all the process determined in the previous step.

Installing the Veritas Agent for WebLogic Server

This chapter describes how to install the Veritas High Availabilty Agent for WebLogic Server. You must install the WebLogic agent on all the systems that will host a WebLogic service group.

Prerequisites for installing the agent for WebLogic

Ensure that you meet the prerequisites before installing the Veritas High Availabilty Agent for WebLogic Server.

Prerequisites for installing the agent in a VCS environment

- Install and configure Veritas Cluster Server.
- If the operating system is HP-UX 11.11, install patch `PHCO_29042`.
- If you have installed a 4.x version of VCS, install the ACC library 5.0 (`VRTSacclib`). For more information, refer to "About the ACC library" on page 14.
- Remove any prior version of this agent.

Prerequisites for installing the agent in a VAD environment

- Install and configure Veritas Application Director.
- Remove any prior version of this agent.

About the ACC library

The operations for Veritas High Availability Agent for WebLogic Server depend on a set of Perl modules known as the ACC Library. The library must be installed on each system in the cluster that will run the WebLogic Server agent. The ACC library contains common, reusable functions that perform tasks such as process identification, logging, and system calls.

Note: If you are installing the agent for WebLogic in a VAD or VCS 5.0 environment, do not install the ACC library package separately. But, if you are installing the agent in a VCS 4.x environment, you must install the ACC library package before installing the agent.

To install or update the ACC library package, locate the library and related documentation on the agent CD and in the compressed agent tar file.

Upgrading the Veritas agent for WebLogic Server

To upgrade the agent, first remove the older version of the agent. Refer to "Uninstalling the Veritas Agent for WebLogic Server" on page 33 for the uninstallation procedure. Then follow the instructions below to install the new agent software.

Installing the agent in a VCS environment

For each platform, perform the installation steps on each system in the cluster.

To install the agent on HP-UX systems

1. Log in as `root`.
2. Go to the `<cd_mount>/hpux/application/weblogic_agent/<vcs_version>/pkgs` directory.
3. Install the package:

   ```
   # swinstall -s . 'pwd' VRTSwls9
   ```

To install the agent on Linux systems

1. Log in as `root`.
2. Go to the `<cd_mount>/linux/application/weblogic_agent/<vcs_version>/rpms` directory.
3. Install the package:

   ```
   # rpm -ihv PackageName-VersionNumber.rpm
   ```

To install the agent on Solaris systems

1. Log in as `root`.
2. Go to the `<cd_mount>/solaris/sparc/application/weblogic_agent/<vcs_version>/pkgs` directory.
3. Install the package:

   ```
   # pkgadd -d . VRTSwls9
   ```

Installing the agent in a VAD environment

This section describes the procedure to install the agent for WebLogic in a VAD environment.

To install the agent

1. Set up an `ssh` communication from the system where you are running the agent installation process, to the systems on which you want to install the agent.

   Note: After installing the agent, you must add the agent types file to the Policy Master database. So, either set up the communication on a running PM system, or set up a communication from the system where you are running the installation to the PM system.

2. Depending upon the environment in which you are installing the agent, choose either the VAD installation software disk or the agent pack disk.
3. Mount the software disc on the system from where you are running the agent installation process.
4. Go to this directory:

Software disc	Directory
Agent pack CD	`/<platform>/application/weblogic_agent/1.0`
VAD installation software disc	`high_availability_agents`

5. Enter this command to begin installing the agent:
   ```
   ./installagpack
   ```
6. Enter the names of the systems on which you want to install the agent.

The installer proceeds to install the agent on the systems. You can view the installation logs in the `/var/VRTS/install/logs` directory.

Configuring the Veritas Agent for WebLogic Server

After installing the agent, you must import the configuration file. After importing this file, you can create and configure a WebLogic Server resource. Before you configure a resource, review the attributes table that describes the WebLogic resource type and its attributes. This chapter includes resource type definition files, and sample `main.cf` and `main.xml` configuration files for reference.

To view sample configuration service groups, refer to “Sample Configurations” on page 41.

Importing the agent types files

To use the agent for WebLogic server, you must import the agent configuration file into the cluster engine.

To import the WebLogic9Types.cf configuration file to work with VCS

Perform the following steps using the Veritas Cluster Server graphical user interface.

1. Start the Veritas Cluster Manager GUI and connect to the cluster on which the agent is installed.
2. Click **File** > **Import Types**.
3. In the **Import Types** dialog box, select the following file:

Version	Directory structure
VCS 4.x	`/etc/VRTSvcs/conf/sample_WebLogic9/WebLogic9Types.cf`
VCS 5.0	`/etc/VRTSagents/ha/conf/WebLogic9/WebLogic9Types.cf`

4 Click **Import**.

5 Save the VCS configuration.

The WebLogic type is now imported to the VCS engine. You can now create WebLogic Server resources. For additional information about using the VCS GUI, refer to the *Veritas Cluster Server User's Guide*.

To import the WebLogic9Types.*<platform>***.xml configuration file to work with VAD**

Before beginning to work with the agent for WebLogic, you must add the Veritas High Availability agent resource types to the Policy Master database configuration.

1 If you are running the agent installation on the PM system, go to step 2.
If you are running the agent installation on a system other than the PM system, set up `ssh` communication between the system on which you are running the agent installation and the PM system.

2 Ensure that the PM daemon is running:
```
haadmin -state
```
The output must show ClusterState as RUNNING.

3 Depending upon the platform on which you are installing the agent and the environment, mount either the VAD installation disk or the agent pack disk on the system.

4 Go to this directory:

Software disc	Directory
Agent pack CD	`/<platform>/application/weblogic_agent/1.0`
VAD installation software disc	`high_availability_agents`

5 Enter this command to add resource types to the PM configuration database:
```
./installagpack -addtypes
```

6 When prompted, enter the virtual IP address of the PM system.

The installer adds the agent types to the PM database configuration and copies the appropriate `types.xml` files to the PM system. You can view the installation logs in the `/var/VRTS/install/logs` directory.

Sample agent type definition

Examples of agent type definition files follow.

While working with VCS 4.x

After importing the agent types into the cluster, if you save the configuration on your system disk using the `haconf -dump` command, you can find the `WebLogic9Types.cf` file in the `/etc/VRTSvcs/conf/config` cluster configuration directory.

An excerpt from this file follows.

```
type WebLogic9 (
    static str ArgList[] = { ResLogLevel, State, IState, AdminURL,
BEA_HOME, DomainName, ListenAddressPort, MonitorProgram,
nmListenAddressPort, nmType, ServerName, ServerRole, User, WLSUser,
WLSPassword, RequireAdminServer, AdminServerMaxWait,
SecondLevelMonitor }
    str ResLogLevel = INFO
    str AdminURL
    str BEA_HOME
    str DomainName
    str ListenAddressPort
    str MonitorProgram
    str nmListenAddressPort
    str nmType
    str ServerName
    str ServerRole
    str User
    str WLSUser
    str WLSPassword
    boolean RequireAdminServer = 0
    int AdminServerMaxWait
    int SecondLevelMonitor = 0
)
```

While working with VCS 5.0

After importing the agent types into the cluster, if you save the configuration on your system disk using the `haconf -dump` command, you can find the `WebLogic9Types.cf` file in the `/etc/VRTSagents/ha/conf/config` cluster configuration directory.

An excerpt from this file follows.

```
type WebLogic9 (
    static str AgentFile = "/opt/VRTSvcs/bin/ScriptAgent"
    static str AgentDirectory = "/opt/VRTSagents/ha/bin/WebLogic9"
    static str ArgList[] = { ResLogLevel, State, IState, AdminURL,
BEA_HOME, DomainName, ListenAddressPort, MonitorProgram,
nmListenAddressPort, nmType, ServerName, ServerRole, User, WLSUser,
WLSPassword, RequireAdminServer, AdminServerMaxWait,
SecondLevelMonitor }
```

```
        str ResLogLevel = INFO
        str AdminURL
        str BEA_HOME
        str DomainName
        str ListenAddressPort
        str MonitorProgram
        str nmListenAddressPort
        str nmType
        str ServerName
        str ServerRole
        str User
        str WLSUser
        str WLSPassword
        boolean RequireAdminServer = 0
        int AdminServerMaxWait
        int SecondLevelMonitor = 0
)
```

While working with VAD

After installing the agent, go to the `/etc/VRTSagents/ha/conf/WebLogic9/` directory to view the `WebLogic9Types.<platform>.xml` agent definition file.

Agent attributes

The WebLogic agent attributes are described as below.

Required attributes

Table 3-1 Required attributes

Required Attribute	Description
BEA_HOME *String*	The absolute path to BEA home directory of WebLogic installation. BEA_HOME is used to locate programs that the agent executes, uniquely identify the ServerRole processes, and to find the location of the `wlst.sh` script and `DomainName` directory: ■ `$BEA_HOME/weblogic90/common/bin/wlst.sh` ■ `$BEA_HOME/user_projects/domains/WLS90Domain/` Example: **`/bea/wls90/mng02`** Default: ""

Table 3-1 Required attributes

Required Attribute	Description
DomainName *String*	The name of the WebLogic domain to which the server instance belongs. DomainName is used to connect to the Node Manager using the `wlst.sh` utility. Specify this attribute for *Administrative* and *Managed* Servers only. For more information about how to set this attribute, refer to "Uniquely identifying WebLogic Server instances" on page 26. Example: **`WLS90Domain`** Default: ""
ListenAddressPort *String*	The Listen Address and port of the WebLogic Server instance. The format is `ListenAddress:port`. Ensure that the `ListenAddress` string resolves to the proper IP Address, using the network name service that you used on the host. The agent connects to the `ListenAddress` on the specified port through the `wlst.sh` API. Specify this attribute for *Administrative* and *Managed* Servers only. Example: **`wls90adminsol.veritas.com:7001`** or **`wls90adminsol.veritas.com:5556`** Default: ""
nmListenAddressPort *String*	The Listen Address and port of the WebLogic Node Manager. The format is `ListenAddress:port`. The value of this attribute must match the values of `ListenAddress` and `ListenPort` that appear in the long listing of processes for a Node Manager instance. The `ListenAddress` string must resolve to a proper IP Address, using the network name service that you used on the host. The agent uses the `ListenAddress` on the specified port to connect through the `wlst.sh` API. Example: **`wlsadmin:5556`** Default: ""
nmType *String*	The WebLogic Node Manager type. This type is used while connecting to the Node Manager through the `wlst.sh` script. Valid values include: ■ **`plain`**: plain socket Java-based implementation ■ **`rsh`**: RSH implementation ■ **`ssh`**: script-based SSH implementation ■ **`ssl`**: Java-based SSL implementation Example: **`ssh`** Default: `ssl`

Table 3-1 Required attributes

Required Attribute	Description
ResLogLevel *String*	The logging detail performed by the agent for the resource. Valid values are: **`ERROR`**: Only logs error messages. **`WARN`**: Logs above plus warning messages. **`INFO`**: Logs above plus informational messages. **`TRACE`**: Logs above plus trace messages. **`TRACE`** is very verbose and should only be used during initial configuration or for troubleshooting and diagnostic operations. Example: **`TRACE`** Default: `INFO`
ServerName *String*	The name of the WebLogic Server. You must specify this attribute for *Administrative* and *Managed* Servers only. For more information about how to set this attribute, refer to "Uniquely identifying WebLogic Server instances" on page 26. Example: **`AdminServer`** Default: ""
WLSUser *String*	The user name of the user that is connecting the `wlst.sh` utility to the server running the WebLogic instance, along with WLSPassword. Specify this attribute for *Administrative* and *Managed* Servers only. Example: **`weblogic`** Default: ""
ServerRole *String*	Type of server. Valid values are: ■ **`NodeManager`**: Online operation executes `wlst.sh` script with `startNodeManager()` API. Example: `startNodeManager(verbose='true',NodeManagerHome='/bea/wls90/admin/weblogic90/common/nodemanager',ListenPort='5556',ListenAddress='wls90adminsol')` ■ **`Administrative`**: Online operation executes `wlst.sh` script with `nmConnect()` and `nmStart()` API. Example: `nmStart ('AdminServer1')` ■ **`Managed`**: Online operation executes `wlst.sh` script with `nmConnect()` and `nmStart()` API. Example: `nmStart ('ManagedServer1')` Default: ""

Table 3-1 Required attributes

Required Attribute	Description
User *String*	The UNIX user name used to start and stop the WebLogic Server instance. If MonitorProgram is specified, the agent uses this user's credentials to run the defined program. You must synchronize the user name across the systems within the cluster. This user name must resolve to the same UID and have the same default shell on each system in the cluster. Agent entry points use the `getpwname(3C)` function system call to obtain UNIX user attributes. Hence you can define the user name locally or in a common repository such as NIS, NIS+, or LDAP). Example: **`wlsadmin`** Default: ““
WLSPassword *String*	The password of user connecting WLST to ServerRole Application Server, along with WLSUser. ■ For VCS, encrypt the value of this attribute using the `$VCS_HOME/bin/vcsencrypt` utility that VCS provides. ■ For VAD, encrypt the value of this attribute using the `$VAD_HOME/bin/vadencrypt` utility that VAD provides. Specify this attribute for *Administrative* and *Managed* Servers only. Example: **`weblogic`** Default: ““

Optional attributes

Table 3-2 Optional attributes

Optional attribute	Description
AdminUrl *String*	The URL of the Managed Server's Administrative Server. Set this attribute only for resources whose ServerRole attribute is *Managed*. Ensure that the value of this attribute is the same as `management.server` that appears in the long listing of processes for the Managed Server. If the RequireAdminServer attribute is set to `1`, AdminUrl is used to connect to the Administrative Server for the domain to determine if the server is fully online. Managed Servers also use this URL to connect to the Administrative Server and download its web applications and services (JMS, JDBC Connection Pool, etc) configuration. Example: **`http://wlsadmin:7001`** Default: ""
AdminServerMaxWait *Integer*	The maximum number of seconds that a Managed Server waits for an Administrative Server to respond to a test probe. For information about setting this attribute, refer to "Delaying Managed Server startup process" on page 26. Example: **`90`** Default: `60`
MonitorProgram *String*	The full pathname and command-line arguments for an externally provided monitor program. For information about setting this attribute, refer to "Executing a custom monitor program" on page 27. Example 1: **`/bea/wls90/admin/mymonitor.sh`** Example 2: **`/usr/local/bin/MyMonitor.sh myWLS.foo.com 8080`** Default: ""

Table 3-2 Optional attributes

Optional attribute	Description
RequireAdminServer *Boolean*	The flag that is used to control the startup behavior of a WebLogic Server instance. When the RequireAdminServer attribute is set to `1` (true), the Managed Server resource is not allowed to complete an initiated online entry point until the Administrative Server is ready to accept connections. If the RequireAdminServer attribute is set to `0` and the AdminServerMaxWait is set to a value `> 5`, the online entry point first probes the Administrative WLS instance to see if it is ready to accept connections. If server is not ready, the entry point waits for 5 seconds and then probes the server again to determine its state. This cycle of *probe and wait* repeats until either the Administrative Server is ready or the AdminServerMaxWait time expires. Specify this attribute for *Managed* Server only. Example: **1** (true) Default: `0` (false)
SecondLevelMonitor *Integer*	Used to enable second-level monitoring. Second-level monitoring is a deeper, more thorough state check of the configured ServerRole. The numeric value specifies how often the monitoring routines must run. 0 means never run the second-level monitoring routines, 1 means run routines every monitor interval, 2 means run routines every second monitor interval, and so on. The agent uses the BEA supplied WebLogic scripting tool `wlst.sh`, to perform second-level monitoring. Depending upon the ServerRole, `wlst.sh` uses api commands `connect()`, `nmConnect()` and `nmServerStatus()` to perform monitoring routines. **Note:** Exercise caution while setting SecondLevelMonitor to large numbers. For example, if the MonitorInterval is set to 60 seconds and the SecondLevelMonitor is set to 100, then `wlst.sh` is executed every 100 minutes, which may not be as often as intended. For maximum flexibility, no upper limit is defined for SecondLevelMonitor. Example: **1** Default: `0`

Uniquely identifying WebLogic Server instances

A Cluster makes it possible to *virtualize* a WebLogic instance. Through the use of shared storage and Virtual IP address assignment, it is easy to manage a large set of WebLogic Server instances in a single cluster. WebLogic Servers can run on separate cluster nodes or can run concurrently on a single node. In the later case, it is important that the High Availability agent can uniquely identify an instance on a node that is hosting more than one simultaneous WLS Server.

Differentiating WebLogic Server instances is especially important when the agent must kill the processes of a non-responsive or failed instance. Failure to define unique names for each WebLogic Server could result in a clean operation that erroneously kills processes for more than one WebLogic Server instance.

- To uniquely identify an Administrative Server instance, the combination of ServerName and DomainName must be unique for the Administrative Server instance.
- To uniquely identify a Managed Server instance:
  - The combination of ServerName and DomainName must be unique for the Managed Server instance.
  - The value of the AdminUrl attribute must match the value of `management.server` that appears in the long listing of processes for the Managed Server instance.
- To uniquely identify a Node Manager instance, the value of the nmListenAddressPort attribute must match the values of `ListenAddress` and `ListenPort` that appear in the long listing of processes for the Node Manager instance.

Delaying Managed Server startup process

WebLogic Managed Servers initiate a connection to the Administrative Server while attempting to download configuration information.

If the cluster administrator is starting up all the WebLogic Servers within the cluster at the same time, delaying the startup process of Managed Servers until the Administrative Server is fully initialized, is advantageous. You can set the AdminServerMaxWait attribute to orchestrate such a delay.

The online operation uses the AdminServerMaxWait attribute to control a repeating cycle of probe, wait, probe, and wait until the presence of the Administrative Server is detected successfully. Once the server is fully initialized, the online operation then proceeds with the Managed Server startup.

If the Administrative Server is not available before the wait time expires, the online operation generates a cluster log warning message and proceeds with instance startup.

You can control the Managed Server delaying process in two ways:

- If the RequireAdminServer attribste is set to `1` (true), the online operation will not proceed until the Administrative Server is available and ready to accept connections. If the time spent waiting on the availability of the Administrative Server exceeds the value of OnlineTimeout, the online entry point will generate an error message indicating the source of the problem and will terminate.
- If the RequireAdminServer attribute is set to `0` (false) and the AdminServerMaxWait attribute is set to a number greater than zero, the online procedure will wait up to AdminServerMaxWait seconds for the *Administrative* Server to transition to a running state before proceeding with the online procedure. If the time spent waiting on the availability of the Administrative Server exceeds the value of AdminServerMaxWait, the online entry point will simply proceed with the remainder of the online steps and will no longer wait on the availability of an Administrative Server.

The online operation interprets the AdminServerMaxWait attribute value as follows:

Value	Interpretation
`0 - 5`	Wait the specified number of seconds, then immediately start the online procedures. Do not check to see if the Admin Server is ready.
`6 - ($NSR-3)`	Wait the specified number of seconds, then check to see if the Admin Server is ready. **`$NSR`** represents the number of seconds remaining before the OnlineTimeout would be reached.
`> ($NSR-3)`	A value greater than the **`$NSR (minus 3)`** causes the agent to wait up to three seconds before the OnlineTimeout is about to expire, and to insert an info-level message into the cluster log file.

Executing a custom monitor program

You can configure the monitor operation to execute a custom monitor program to perform a user-defined WebLogic Server state check. Based on the UNIX user defined in the User attribute, this MonitorProgram runs in this user-defined shell.

The monitor operation executes MonitorProgram if:

- The MonitorProgram attribute value is set to a valid executable program.

- The first level process check indicates that the WebLogic Server instance is online.
- The SecondLevelMonitor attribute is either set to 0 (false), or SecondLevelMonitor is set to 1 (true) and the second level check indicates that the WebLogic Server instance is online.

This feature allows cluster administrators to define custom programs that can further determine the state of the WebLogic server. For example, if the administrator wants to test the status of a J2EE component running inside the WebLogic server, the administrator can execute a custom program to determine that the underlying application is working properly.

The monitor operation interprets the program exit code as follows:

Exit code	Interpretation
110 or 0	WebLogic Server instance is ONLINE
100 or 1	WebLogic Server instance is OFFLINE
99	WebLogic Server instance is UNKNOWN
Any other value	WebLogic Server instance is UNKNOWN

To ensure that the custom monitor program is always available to the agent application, Symantec recommends storing the file in the directory that the BEA_HOME attribute specifies on the shared storage device.

Clustering WebLogic Servers

This chapter provides an overview of the clustering process, as well as information about configuring service groups on a cluster.

Overview of the clustering process

Assuming that the target implementation has licensed the Veritas Storage Foundation and High Availability products, perform the following steps to cluster an instance of WebLogic Server:

1. Create UNIX user and group accounts.

   Create a UNIX username in the cluster namespace (NIS, NIS+, LDAP or the local password files) for WebLogic Server operations. Ensure that all cluster nodes use the same user with the same user UID and default shell.

   Create a UNIX group in the cluster namespace (NIS, NIS+, LDAP or the local group file) for WebLogic Server operations.

   Caution: Symantec recommends the use of the local configuration files over naming services like NIS, NIS+ or LDAP for the reason that name resolution using a centralized service takes additional time and is subject to network delays. If the local file approach is used, ensure that all nodes are updated with the exact same information to guarantee consistency throughout the cluster. Also make sure the name service resolution configuration (`/etc/nsswitch.conf` on most UNIX systems) gives preference to the local files over centralized naming services.

   ---

2 Create the Supporting Directory Structure.

A well-designed directory structure for your WebLogic Server instances will simplify the cluster configuration and create a storage environment that is more intuitive and easier to manage. Assuming that all WebLogic Server instances will be clustered and installed on shared disk, Symantec recommends a directory structure similar to the following:

Directory	Purpose
`/wls90`	Root directory in which to group all WebLogic Server instances supporting a particular domain.
`/wls90/admin`	Path used to mount the file system dedicated for the WebLogic Administration Server program and configuration files. All WebLogic binaries and configuration files for this Administration Server are stored in this file system.
`/wls90/mng01`	Path used to mount the file system dedicated for WebLogic Managed Server 1 program and configuration files. All WebLogic binaries and configuration files for Managed Server 1 are stored in this file system.
`/wls90/mng02`	Path used to mount the file system dedicated for WebLogic Managed Server 2 program and configuration files. All WebLogic binaries and configuration files for Managed Server 2 are stored in this file system.

Additional notes about the example directory structure:

- The directories and subdirectories above are created on the root file system on each system in the cluster. The mount points need to exist on all systems in the cluster that are configured to run the WebLogic Server instance.
- The sub-directories under `/wls90` are mount points on which file systems will be mounted. These file systems are stored on shared disks. Each WebLogic Server instance is installed on its own dedicated file system; it is not installed in the root file system.
- The example above includes directories for only two WebLogic Managed Servers, but the naming structure supports an unlimited number.

3 Create high level mount points for WebLogic Server operations.

4 Create a disk group and volume.

Consult the *Veritas Volume Manager 4.0 Administrator's Guide* for details on how to provision disk group and volume resources.

5 Create the file system.

6 Create a Virtual IP Address.

Provision a Virtual IP address in the network namespace (i.e. NIS, NIS+ or LDAP). Ensure the IP address and host name pair are defined for all nodes in the cluster. If the IP and host name pair are defined in the local host map, make sure all cluster nodes have the same host map record.

7 Create service group and resources on a cluster.

Create a service group on a cluster and define resources for the NIC, IP, DiskGroup, and Mount resources. Consult the cluster documentation for detailed information on nic, ip, diskgroup and mount resource types.

Online these newly created resources on one node in the cluster.

8 Install & Configure WebLogic Server.

Install the WebLogic software on the newly created and mounted file system. Once installed, change the file and group ownership to reflect the WebLogic Server UNIX user and group accounts created earlier.

Modify the server configuration to use the Virtual IP address and port. Refer the BEA WebLogic Server documentation for instructions to bind a WebLogic Server instance to its dedicated virtual IP address and port number. Configuring the WebLogic Server to bind is essential to ensure that it always listens on the same virtual IP address and port number regardless of the system in the cluster on which it is running.

9 Finalize & Test the Configuration.

Create the WebLogic Server resource.

Online the newly created resource.

Test instance startup, shutdown and switchover as required, confirming overall availability requirements.

If you want to refer to a sample configuration service group, go to “Sample Configurations” on page 41.

Uninstalling the Veritas Agent for WebLogic Server

Follow the steps below to remove the Veritas High Availability agent for WebLogic Server from the cluster. You must perform these steps while the cluster is active.

To uninstall the agent

1. Log in as `root`.
2. Set the cluster configuration mode to read/write by typing the following command from any system in the cluster:
   ```
   # haconf -makerw
   ```
3. Remove all WebLogic resources from the cluster. Use the following command to verify that all resources have been removed.
   ```
   # hares -list Type=WebLogic9
   ```
4. Remove the agent from the cluster configuration by typing the following command from any system in the cluster.
   ```
   #hatype -delete WebLogic9
   ```
   Removing the agent's type file from the cluster removes the include statement for the agent from the `main.cf` file, but the agent's type file is not removed from the cluster configuration directory. You can remove the agent's type file later, from the cluster configuration directory.
5. Save these changes. Then set the cluster configuration mode to read-only by typing the following command from any system in the cluster:
   ```
   # haconf -dump -makero
   ```

6 Use the platform's native software management program to remove the WebLogic agent software from each node in the cluster. Execute the following command to uninstall the agent:

Platform	**Command**
HP-UX	`# swremove VRTSwls9`
Linux	`# rpm -e VRTSwls9`
Solaris	`# pkgrm VRTSwls9`

Troubleshooting the Veritas Agent for WebLogic Server

This chapter covers tips and pointers on using the agent for WebLogic with Veritas high availability products. To resolve issues effectively, follow the steps in the order presented below. You may come across unique issues, but make sure that you follow these steps in the presented order to avoid unnecessary issues.

These troubleshooting tips and pointers are applicable whether working with VCS or with VAD clustering technologies. Wherever applicable, VCS and VAD are referred to as cluster.

Testing WebLogic Servers outside a cluster

If you face problems while working with a resource, you must disable the resource within the cluster framework. A disabled resource is not under the control of the cluster framework, and so you can test the WebLogic server independent of the cluster framework. Refer to the cluster documentation for information about disabling a resource.

You can then restart the WebLogic Server outside the cluster framework.

Note: Use the same parameters that the resource attributes define within the cluster framework while restarting the resource outside the framework.

To restart a Node Manager outside the cluster framework

1 Log in as `root` in to the host on which the WebLogic Node Manager application is to run.

2. Use the values defined in the agent attributes to initiate the Node Manager start program.
   For example, assume that the following values are assigned:

Attribute	Value
User	**weblogic**
BEA_HOME	**/bea/wls90/admin**
nmListenAddressPort	**wls90admsol:5556**
nmType	**ssl**
ServerRole	**NodeManager**

3. Log in to the Node Manager using the user name specified in the User attribute:
   ```
   su - weblogic
   ```
4. Go to the directory specified in the BEA_HOME attribute:
   ```
   cd /bea/wls90/admin
   ```
5. Start the WebLogic Scripting Tool:
   ```
   /bea/wls90/admin/weblogic90/common/bin/wlst.sh
   ```
6. Start the Node Manager:
   ```
   startNodeManager(verbose='true',NodeManagerHome='/bea/wls90/adm
   in/weblogic90/common/nodemanager',
   ListenPort='5556',ListenAddress='wls90admsol')
   ```

If the Node Manager starts succeefully, following message is displayed:
```
Successfully launched the Node Manager.
```

7. Enter this command:
   ```
   exit ()
   ```

If the Node Manager works properly outside the cluster framework, you can then attempt to implement the Node Manager within the cluster framework.

To restart a Managed or Administrative Server outside the cluster framework

1. Log in as `root` in to the host on which the WebLogic application is to run.
2. Use the values defined in the agent attributes to initiate the WebLogic Server start program.

For example, for an Administrative Server, assume that the following values are assigned:

Attribute	Value
ServerName	**AdminServer**
ServerRole	**Administrative**
BEA_HOME	**/bea/wls90/admin**
DomainName	**WLS90Domain**
nmListenAddressPort	**wls90admsol:5556**
nmType	**ssl**
User	**weblogic**

3. Log in to the Administrative Server using the user name specified in the User attribute:
   ```
   su - weblogic
   ```
4. Go to the directory specified in the BEA_HOME attribute:
   ```
   cd /bea/wls90/admin
   ```
5. Start the WebLogic Scripting Tool:
   ```
   /bea/wls90/admin/weblogic90/common/bin/wlst.sh
   ```
6. Connect to the Node Manager:
   ```
   nmConnect('weblogic', 'asdf1234', 'wls90adminsol', '5556',
   'WLS90Domain',
   '/bea/wls90/admin/user_projects/domains/WLS90Domain','ssl')
   ```
7. Start the Administrative Server:
   ```
   nmStart (‘AdminServer’)
   ```

If the server starts successfully, the following message is displayed:

```
Starting server AdminServer
Server AdminServer started successfully
```

If the WebLogic Server works properly outside the cluster framework, you can then attempt to implement the server within the cluster framework.

Using correct software and operating system versions

To ensure that no issues arise due to incorrect software and operating system versions, refer to “Supported software” on page 8 for the correct versions of operating system and software to be installed on the resource systems.

Meeting prerequisites

Before installing the WebLogic agent, double check that you meet the prerequiste requirements. For example, you must install the ACC library on VCS before installing the WebLogic agent. Refer to “Prerequisites for installing the agent for WebLogic” on page 13 for the list of prerequisites.

Configuring WebLogic resources

Before using a WebLogic resource, ensure that you configure the resource properly. Refer to “Agent attributes” on page 20 for the list of resource types with which you can configure all WebLogic resources.

Reviewing log files

If you are facing problems while using the WebLogic agent or a WebLogic Server instance, refer to the following sections to access the relevant files for information about the issue.

Using WebLogic agent log files

In case of problems while using the WebLogic agent, you can access the agent log files for more information. The agent saves output of every entry point process in the temporary folder of the resource system. If the temporary folder is `/tmp`, the log files are saved using the following naming format:

```
/tmp/.VRTS<AgentName>/<ResourceName>_<EntryPointName>.out
```

For example:

```
/tmp/.VRTSWebLogic9/WLS90Mng01_nodemanager_online.out
/tmp/.VRTSWebLogic9/WLS90Mng01_nodemanager_offline.out
/tmp/.VRTSWebLogic9/WLS90Mng01_nodemanager_clean.out
/tmp/.VRTSWebLogic9/WLS90Mng01_nodemanager_monitor.out
```

If a resource, *WLS90Mng01_nodemanager* is unable to bring a WebLogic Node Manager online, you can access the `/tmp/.VRTSWebLogic9/WLS90Mng01_nodemanager_online.out` for more information so that you can diagnose the problem.

Caution: These files are overwritten each time you execute the corresponding entry point process. In case of information that you want to save, make a copy of the files at another location.

Using cluster log files

In case of problems while using the WebLogic agent, you can also access the cluster engine log file for more information about a particular resource.

- The VCS engine log file is `/var/VRTSvcs/log/engine_A.log`.
- The VAD engine log file is `/var/VRTSvad/log/engine_A.log`.

Using WebLogic Server log files

If the WebLogic Server is facing problems, access the log files of the WebLogic Server to further investigate the problem. The log files are located at:

- For Node Managers:

  ```
  <BEA_HOME>/weblogic90/common/nodemanager/nodemanager.log
  ```

- For Administrative Servers:

  ```
  <BEA_HOME>/user_projects/domains/<DomainName>/servers/<ServerNa
  me>/<ServerName>.log
  <BEA_HOME>/user_projects/domains/<DomainName>/servers/<ServerNa
  me>/<ServerName>.out
  ```

- For Managed Servers:

  ```
  <BEA_HOME>/user_projects/domains/<DomainName>/servers/<ServerNa
  me>/<ServerName>.log
  <BEA_HOME>/user_projects/domains/<DomainName>/servers/<ServerNa
  me>/<ServerName>.out
  <BEA_HOME>/user_projects/domains/<DomainName>/servers/<ServerNa
  me>/access.log
  ```

Using trace level logging

The ResLogLevel attribute controls the level of logging that is written in a cluster log file for each WebLogic resource. You can set this attribute to **TRACE**, which enables very detailed and verbose logging.

If you set ResLogLevel to **TRACE**, a very high volume of messages is produced. Symantec recommends that you must localize the ResLogLevel attribute for particular resource.

To localize ResLogLevel attribute for a resource

1. Identify the resource for which you want to enable detailed logging.
2. Localize the ResLogLevel attribute for the identified resource:

   ```
   hares -local Resource_Name ResLogLevel
   ```

3. Set the ResLogLevel attribute to **TRACE** for the identified resource:

   ```
   hares -modify Resource_Name ResLogLevel TRACE -sys SysA
   ```

4. Test the identified resource. The operation reproduces the problem that you are attempting to diagnose.
5. Set the ResLogLevel attribute back to **INFO** for the identified resource:
   ```
   hares -modify Resource_Name ResLogLevel INFO -sys SysA
   ```
6. Review the contents of the cluster engine output log file to diagnose the problem.

You may also contact Symantec support for more help.

Sample Configurations

This appendix describes a typical service group configured to monitor the state of WebLogic Servers in a cluster. The sample configuration graphically depicts the resource types, resources, and resource dependencies within the service group. Review these dependencies carefully before configuring the agent. For more information about these resource types, see the *Veritas Cluster Server Bundled Agents Reference Guide*.

Sample service group configuration

A WebLogic Server resource usually consists of:

Disk Group: Veritas Volume Manager disk group contains information required by the DiskGroup agent to import and export the shared disk object used in support of a clustered WebLogic instance. While the use of shared disk is not required to cluster an instance of WebLogic, Symantec recommends the use of a shared volume to eliminate the requirement to synchronize local copies of the WLS binaries and configuration files on each node in a multi-node cluster.

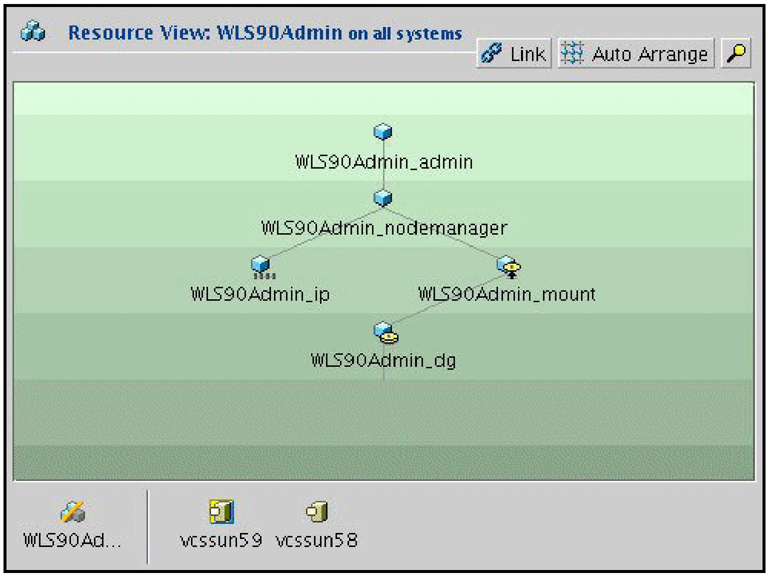
Mount: This resource mounts, monitors, and unmounts the file system that is dedicated to the WebLogic Server installation and configuration files. Use the resource type Mount to create this resource.

Network Interface: This resource monitors the network interface card through which the WebLogic Server communicates with other services.

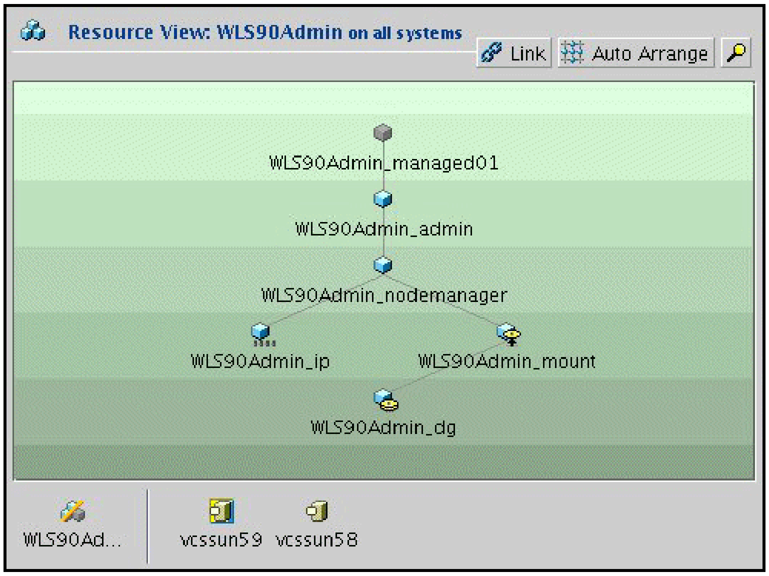
Virtual IP: This resource configures the virtual IP address dedicated to the WebLogic Server. External services, programs, and clients use this address to communicate with this WebLogic Server instance.

WebLogic Server: This resource starts, stops, and monitors the WebLogic Server instance. Use the WebLogic resource type to create this resource.

The following figure shows an example of how a single service group looks with an Administrative Server only.



The following figure is an example of how a service group looks with Administrative and Managed Servers.

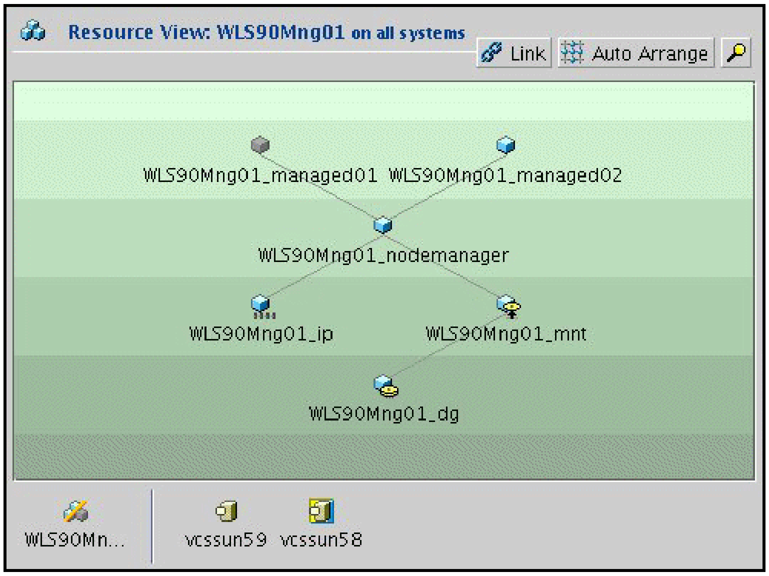


Service Group dependencies

Cluster administrators use Service Group dependencies to create links between unrelated Service Group objects within a cluster. In this version of WebLogic server, you no longer require Service Group dependencies.

The Managed Server online operation can automatically perform an Administrative Server probe. So even though Managed Server instances depend on the domain Administrative Server instance, you can have a Service Group with Managed Servers only. For more information, refer to “Delaying Managed Server startup process” on page 26.

For example, the following figure shows how a single Service Group looks with Managed Servers only.



Sample configuration in a VCS environment

To provide a complete example, the following `main.cf` excerpt from a Solaris cluster defines a service group to support one WebLogic Server instance.

```
group wls90Admin
(
SystemList = { systemA = 1, systemB = 2 }
)
 DiskGroup wls90Admin_dg (
        DiskGroup = wls90admin
        )
 Mount wls90Admin_mnt (
    MountPoint = "/wls90/admin"
    BlockDevice = "/dev/vx/dsk/wls90admin/wlsadmin"
    FSType = vxfs
    FsckOpt = "-y"
    )
NIC wls90Admin_nic (
    Device = hme0
    NetworkType = ether
    )
IP wls90Admin_ip (
     Device = hme0
     Address = "192.126.5.166"
     NetMask = "255.255.255.0"
     )
WebLogic wls90Admin_app (
     User = wls90admin
     EnvFile = "/wls90/admin/config/wls90/setWLSenv.sh"
     ScriptDir = "/wls90/admin/config/wls90"
     StartScript = "startWebLogic.sh"
     StopScript = "stopWebLogic.sh"
     Host = wls90admhost
     ServerRole = Administrative
     ServerName = AdminServer
     SecondLevelMonitor = 1
     )
wls90Admin_app requires wls90Admin_ip
wls90Admin_app requires wls90Admin_mnt
wls90Admin_ip  requires wls90Admin_nic
wls90Admin_mnt requires wls90Admin_dg
```

Sample configuration in a VAD environment

To view a sample VAD configuration file (`main.xml`) with an Administrative Server instance, a Node Manager instance, and a Managed Server instance, go to the `/etc/VRTSagents/ha/conf/WebLogic9/` directory.

Index

A

ACC library 14
- package 13
- supported versions 8

Agent
- removing 33

agent
- clustering 29
- configuring agent 17
- installation prerequistes 13
- installing 13
- introduction 7
- sample configuration 26
- sample service group configuration 41
- supported operating systems 8
- supported software 8
- upgrading 14
- what's new 7

agent attributes
- AdminServerMaxWait 24
- AdminUrl 24
- BEA_HOME 20
- DomainName 21
- ListenAddressPort 21
- MonitorProgram 24
- nmListenAddressPort 21
- nmType 21
- RequireAdminServer 25
- ResLogLevel 22
- SecondLevelMonitor 25
- ServerName 22
- ServerRole 22
- type definition 19
- User 23
- WLSPassword 23
- WLSUser 22

agent operations 8

agent uninstallation 33
- uninstalling on HP-UX 34
- uninstalling on Linux 34
- uninstalling on Solaris 34

AIX
- installing 16

Appendix B 41

H

HP-UX
- supported versions 8
- uninstalling 34

hp-ux
- installing 15

I

installing agent 15
- hp-ux 15
- linux 15
- solaris 15

L

Linux
- supported versions 8
- uninstalling 34

linux
- installing 15

R

release notes
- previous fixes 7

S

Solaris
- supported versions 8
- uninstalling 34

solaris
- installing 15

supported software 7

V

VAD

- supported versions 8
- Veritas Application Director 7

VCS
- clustering overview 29
- supported versions 8
- Veritas Cluster Server 7