Cluster Server Agent for WebSphere Application Server Installation and Configuration Guide

AIX, Linux, Solaris

8.0.2

Veritas InfoScale™ Availability Agents

Last updated: 2023-06-30

Legal Notice

Veritas Technologies LLC
2625 Augustine Drive
Santa Clara, CA 95054

http://www.veritas.com

Technical Support

Technical Support maintains support centers globally. All support services will be delivered in accordance with your support agreement and the then-current enterprise technical support policies. For information about our support offerings and how to contact Technical Support, visit our website:

https://www.veritas.com/support

You can manage your Veritas account information at the following URL:

https://my.veritas.com

If you have questions regarding an existing support agreement, please email the support agreement administration team for your region as follows:

Worldwide (except Japan)	CustomerCare@veritas.com
Japan	CustomerCare_Japan@veritas.com

Documentation

Make sure that you have the current version of the documentation. Each document displays the date of the last update on page 2. The latest documentation is available on the Veritas website:

https://sort.veritas.com/documents

Documentation feedback

Your feedback is important to us. Suggest improvements or report errors or omissions to the documentation. Include the document title, document version, chapter title, and section title of the text on which you are reporting. Send feedback to:

infoscaledocs@veritas.com

You can also see documentation information or ask a question on the Veritas community site:

http://www.veritas.com/community/

Veritas Services and Operations Readiness Tools (SORT)

Veritas Services and Operations Readiness Tools (SORT) is a website that provides information and tools to automate and simplify certain time-consuming administrative tasks. Depending on the product, SORT helps you prepare for installations and upgrades, identify risks in your datacenters, and improve operational efficiency. To see what services and tools SORT provides for your product, see the data sheet:

https://sort.veritas.com/data/support/SORT_Data_Sheet.pdf

Contents

Chapter 1

Introducing the agent for WebSphere Application Server

This chapter includes the following topics:

- About the Cluster Server agent for WebSphere Application Server
- Supported software
- How the agent supports intelligent resource monitoring
- WebSphere Application Server agent functions

About the Cluster Server agent for WebSphere Application Server

Cluster Server (VCS) agents monitor specific resources within an enterprise application. They determine the status of resources and start or stop them according to external events.

The Cluster Server agent for WebSphere Application Server provides high availability for WebSphere Application Server in a clustered environment.

The agent supports the following types of WebSphere Application Server instances:

- Deployment Manager
- Node Agent
- Application Server
- Cluster

For the latest updates or software issues for this agent, see the *Cluster Server Agent Pack Release Notes*.

Supported software

For information on the software versions that the Cluster Server agent for WebSphere Application Server supports, see the Veritas Services and Operations Readiness Tools (SORT) site: https://sort.veritas.com/agents.

How the agent supports intelligent resource monitoring

With Intelligent Monitoring Framework (IMF), VCS supports intelligent resource monitoring in addition to the poll-based monitoring. Poll-based monitoring polls the resources periodically whereas intelligent monitoring performs asynchronous monitoring.

When an IMF-enabled agent starts up, the agent initializes the Asynchronous Monitoring Framework (AMF) kernel driver. After the resource is in a steady state, the agent registers with the AMF kernel driver, the details of the resource that are required to monitor the resource.

For example, the agent for WebSphere Application Server registers the PIDs of the WebSphere Application Server processes with the AMF kernel driver.

The imf_getnotification function of the agent waits for any resource state changes. When the AMF kernel driver module notifies the imf_getnotification function about a resource state change, the agent framework runs the monitor agent function to ascertain the state of that resource. The agent notifies the state change to VCS, which then takes appropriate action.

For more information, see the *Cluster Server Administrator’s Guide*.

WebSphere Application Server agent functions

The agent consists of resource type declarations and agent executables. The agent executables are organized into online, offline, monitor, and clean functions.

Online

The online function is responsible for starting a WebSphere Application Server. The online function performs the following tasks:

- Verifies that the WebSphere Application Server instance is not already online.

- Determines the version of the WebSphere Application Server software.
- Starts the WebSphere Application Server instance by executing the appropriate start script, which is supplied by the WebSphere installation program. The script executed depends upon the type of server being started.

Server Type	Start Command
Deployment Manager	`WAS_HOME/binDir/startManager.sh`
Node Agent	`WAS_HOME/binDir/startNode.sh`
Application Server	`WAS_HOME/binDir/startServer.sh`

Cluster	`ServerProfile/bin/wsadmin.sh -lang jython -c "clusterName='<cluster_name>';`
	`clusterObj = AdminControl.completeObjectName('type=Cluster,`
	`name='+clusterName+',*');`
	`AdminControl.invoke( clusterObj, 'start');"`

Offline

The offline function is responsible for stopping a WebSphere Application Server instance. The offline function performs the following tasks:

- Verifies that the WebSphere Application Server instance is not already offline.
- Determines the version of the WebSphere Application Server software.
- Stops the WebSphere Application Server instance by executing the appropriate stop script, which is supplied by the WebSphere installation program. The script executed depends upon the type of server being stopped.

Server Type	Stop Command
Deployment Manager	`WAS_HOME/binDir/stopManager.sh`
Node Agent	`WAS_HOME/binDir/stopNode.sh`
Application Server	`WAS_HOME/binDir/stopServer.sh`

Cluster	`ServerProfile/bin/wsadmin.sh -lang jython -c "clusterName='<cluster_name>';`
	`clusterObj = AdminControl.completeObjectName('type=Cluster,`
	`name='+clusterName+',*');`
	`AdminControl.invoke( clusterObj, 'stop');"`

Monitor

The monitor function is responsible for monitoring the state of WebSphere Application Servers on all nodes in the cluster.

The monitor function performs the following tasks:

- First-level monitoring quickly checks for the existence of the system process (the Java Virtual Machine) that represents the WebSphere Application Server instance. It determines the process existence by scanning the system process table and searching for strings in the process command line that uniquely identify the JVM process associated with the WebSphere Application Server instance. These search strings include the values specified in resource attributes WAS_HOME, WAS_NODE, and ServerName.

  For the Cluster server type, the agent queries the Deployment Manager to get the current cluster state. The agent uses the wsadmin scripts to obtain the cluster state.

  The agent also supports Intelligent Monitoring Framework (IMF) in the first-level check. IMF enables intelligent resource monitoring. See “How the agent supports intelligent resource monitoring” on page 8. You can use the MonitorFreq key of the IMF attribute to specify the frequency at which the agent invokes the monitor function. See “MonitorFreq” on page 48.

- If second-level monitoring is enabled (if SecondLevelMonitor > 0), the monitor function performs a deeper, more thorough state check of the WebSphere Application Server. Second-level monitoring uses the IBM-supplied utility program serverStatus.sh. The output from this program is parsed to confirm the server is running.

  When enabled, the integer value specified in attribute SecondLevelMonitor determines how frequently the program is executed. For example, if SecondLevelMonitor is set to 1, the monitor function executes serverStatus.sh during each monitor interval. If SecondLevelMonitor is set to 3, the monitor function executes serverStatus.sh every third monitor interval. This mechanism lets you control the system load generated by monitoring.

  The serverStatus.sh script spawns a Java program that establishes a connection to the WebSphere Application Server. Spawning a JVM every monitor interval places additional load on the system. If performance is more important than a second-level state check, then consider disabling second-level monitoring and only performing the first-level process check.

Note: The attribute used to configure the second-level check and its frequency depends on the software versions of VCS and WebSphere agent you have installed: For VCS 5.1 SP1 or later with Websphere agent version 5.1.9.0, use the LevelTwoMonitorFreq attribute. For VCS 5.1 or earlier with WebSphere agent 5.1.8.0 or earlier, use the SecondLevelMonitor attribute.

- The monitor function executes a custom monitor program specified in the attribute MonitorProgram. This program does not execute if either the first- or second-level monitor reports that the resource is offline. You can omit second-level monitoring, and attempt running a custom monitor check immediately after first-level monitoring.
  This feature allows VCS dministrator to define custom programs that determine the state of the WebSphere Application Server. For example, the administrator may want to test the status of a J2EE component running inside the server and ensure that the underlying application is functioning properly.
  See “WebSphere Application Server agent attributes” on page 32.

Clean

The clean function removes any WebSphere Application Server instance processes remaining after a fault event or after an unsuccessful attempt to online or offline the resource.

The clean function performs the following tasks:

- Kills the process that starts the WebSphere Application Server instance. It is unlikely that this process exists, but it needs to be removed if for some reason it still exists during clean.
- Kills the process that stops the WebSphere Application Server instance. It is unlikely this process exists, but it needs to be removed if for some reason it still exists during clean.
- Kills the JVM process for the WebSphere Application Server instance. This process is identified by searching the system process table using the values specified in attributes WAS_HOME, WAS_NODE, and ServerName.
- Stops all cluster members immediately for the Cluster server type. The agent uses the `stopImmediate` option of the cluster to stop the cluster members.

Note: For information about the additional functions of the agent for WebSphere Application Server when IMF is enabled: See "Agent functions for the IMF functionality" on page 47.

Chapter 2

Installing, upgrading, and removing the agent for WebSphere Application Server

This chapter includes the following topics:

Before you install the Cluster Server agent for WebSphere Application Server

You must install the Cluster Server agent for WebSphere Application Server on all the systems that will host WebSphere Application Server service groups.

Before you install the agent for WebSphere Application Server, ensure that the following prerequisites are met.

- Install and configure Cluster Server.
  For more information on installing and configuring Cluster Server, refer to the Cluster Server installation and configuration guides.
- Remove any previous version of this agent.
  To remove the agent,
  See "Uninstalling the agent in a VCS environment" on page 21.
- Install the latest version of ACC Library.
  To install or update the ACC Library package, locate the library and related documentation in the Agent Pack tarball.
  See "About the ACC library" on page 15.

Prerequisites for enabling i18n support

Perform the following steps to enable i18n support to the agent:

- Install ACCLib version 5.1.2.0 or later.
  See "Installing the ACC library" on page 15.
- For VCS 5.0 and earlier releases, copy the latest ag_i18n_inc.pm module from the following location on the agent pack disc.

Note: Review the readme.txt for instructions to copy this module.

VCS 5.0	cd1/*platform*/*arch_dist*/vcs/application/i18n_support/5.0
VCS 4.1	cd1/*platform*/*arch_dist*/vcs/application/i18n_support/4.1
VCS 4.0	cd1/*platform*/*arch_dist*/vcs/application/i18n_support/4.0

where *arch_dist* takes the following values:
'sol_sparc' for Solaris SPARC
'generic' for Linux

Note: *arch_dist* is not applicable to AIX.

About the ACC library

The operations of a Cluster Server agent depend on a set of Perl modules known as the ACC library. The library must be installed on each system in the cluster that runs the agent. The ACC library contains common, reusable functions that perform tasks, such as process identification, logging, and system calls.

Instructions to install or remove the ACC library on a single system in the cluster are given in the following sections. The instructions assume that the ACCLib tar file has already been extracted.

Note: The LogDbg attribute should be used to enable debug logs for the ACCLib-based agents when the ACCLib version is 6.2.0.0 or later and VCS version is 6.2 or later.

Installing the ACC library

Install the ACC library on each system in the cluster that runs an agent that depends on the ACC library.

To install the ACC library

1. Log in as a superuser.
2. Download ACC Library.

   You can download either the complete Agent Pack tar file or the individual ACCLib tar file from the Veritas Services and Operations Readiness Tools (SORT) site (https://sort.veritas.com/agents).
3. If you downloaded the complete Agent Pack tar file, navigate to the directory containing the package for the platform running in your environment.

AIX	*cd1*/aix/vcs/application/acc_library/*version*_library/pkgs
Linux	*cd1*/linux/generic/vcs/application/acc_library/*version*_library/rpms
Solaris	*cd1*/solaris/*dist_arch*/vcs/application/acc_library/*version*_library/pkgs where *dist_arch* is sol_sparc.

4. If you downloaded the individual ACCLib tar file, navigate to the pkgs directory (for AIX and Solaris), or rpms directory (for Linux).

5. Install the package. Enter **Yes**, if asked to confirm overwriting of files in the existing package.

   AIX
   ```
   # installp -ac -d VRTSacclib.bff VRTSacclib
   ```

   Linux
   ```
   # rpm -i \
   VRTSacclib-VersionNumber-GA_GENERIC.noarch.rpm
   ```

   Solaris
   ```
   # pkgadd -d VRTSacclib.pkg
   ```

   See "Installing the ACC library IPS package on Oracle Solaris 11 systems" on page 16.

Note: The LogDbg attribute should be used to enable debug logs for the ACCLib-based agents when the ACCLib version is 6.2.0.0 or later and VCS version is 6.2 or later.

Installing the ACC library IPS package on Oracle Solaris 11 systems

Install the ACC library IPS package on an Oracle Solaris 11 system.

To install the ACC library IPS package on Oracle Solaris 11 systems

1. Copy the `VRTSacclib.p5p` package from the `pkgs` directory to the system in the `/tmp/install` directory.

2. Disable the publishers that are not reachable as package install may fail, if any, of the already added repositories are unreachable.

   ```
   # pkg set-publisher --disable <publisher name>
   ```

3. Add a file-based repository in the system.

   ```
   # pkg set-publisher -g /tmp/install/VRTSacclib.p5p Veritas
   ```

4. Install the package.

   ```
   # pkg install --accept VRTSacclib
   ```

5. Remove the publisher from the system.

   ```
   # pkg unset-publisher Veritas
   ```

6. Enable the publishers that were disabled earlier.

   ```
   # pkg set-publisher --enable <publisher name>
   ```

Installing the ACC library package on Solaris brand non-global zones

With Oracle Solaris 11, you must install the agent package inside non-global zones. The native non-global zones are called Solaris brand zones.

To install the ACC library package on Solaris brand non-global zones

1 Ensure that the SMF services, `svc:/application/pkg/system-repository:default` and `svc:/application/pkg/zones-proxyd:default`, are online on the global zone.

```
# svcs svc:/application/pkg/system-repository:default
```

```
# svcs svc:/application/pkg/zones-proxyd:default
```

2 Log on to the non-global zone as a superuser.

3 Ensure that the SMF service `svc:/application/pkg/zones-proxy-client:default` is online inside non-global zone:

```
# svcs svc:/application/pkg/zones-proxy-client:default
```

4 Copy the `VRTSacclib.p5p` package from the `pkgs` directory to the non-global zone (for example, at the `/tmp/install` directory).

5 Disable the publishers that are not reachable, as package install may fail, if any of the already added repositories are unreachable.

```
# pkg set-publisher --disable <publisher name>
```

6 Add a file-based repository in the non-global zone.

```
# pkg set-publisher -g/tmp/install/VRTSacclib.p5p Veritas
```

7 Install the package.

```
# pkg install --accept VRTSacclib
```

8 Remove the publisher on the non-global zone.

```
# pkg unset-publisher Veritas
```

9 Clear the state of the SMF service, as setting the file-based repository causes the SMF service `svc:/application/pkg/system-repository:default` to go into the maintenance state.

```
# svcadm clear svc:/application/pkg/system-repository:default
```

10 Enable the publishers that were disabled earlier.

```
# pkg set-publisher --enable <publisher>
```

Note: Perform steps 2 through 10 on each non-global zone.

Installing the agent in a VCS environment

Install the agent for WebSphere Application Server on each node in the cluster.

To install the agent in a VCS environment

1. Download the agent from the Veritas Services and Operations Readiness Tools (SORT) site: https://sort.veritas.com/agents.

   You can download either the complete Agent Pack tar file or an individual agent tar file.

2. Uncompress the file to a temporary location, say /tmp.

3. If you downloaded the complete Agent Pack tar file, navigate to the directory containing the package for the platform running in your environment.

AIX	`cd1/aix/vcs/application/websphere_agent/`
	`vcs_version``/`*`version`*`_agent/pkgs`

Linux	`cd1/linux/generic/vcs/application/websphere_agent/`
	`vcs_version``/`*`version`*`_agent/rpms`

Solaris	`cd1/solaris/`*`dist_arch`*`/vcs/application/websphere_agent/`
	`vcs_version``/`*`version`*`_agent/pkgs`
	where, *dist_arch* is sol_sparc

   If you downloaded the individual agent tar file, navigate to the pkgs directory (for AIX and Solaris), or rpms directory (for Linux).

4. Log in as a superuser.

5. Install the package.

   AIX

   ```
   # installp -ac -d
   VRTSvcswas.rte.bff VRTSvcswas.rte
   ```

   Linux

   ```
   # rpm -ihv \
   VRTSvcswas-AgentVersion-GA_GENERIC.noarch.rpm
   ```

   Solaris

   ```
   # pkgadd -d . VRTSvcswas
   ```

After installing the agent package, you must import the agent type configuration file.

See "Importing the agent types files in a VCS environment" on page 29.

Installing the agent IPS package on Oracle Solaris 11 systems

To install the agent IPS package on an Oracle Solaris 11 system

1. Copy the `VRTSvcswas.p5p` package from the `pkgs` directory to the system in the `/tmp/install` directory.
2. Disable the publishers that are not reachable as package install may fail, if any of the already added repositories are unreachable.

   ```
   # pkg set-publisher --disable <publisher name>
   ```

   where the publisher name is obtained using the `pkg publisher` command.
3. Add a file-based repository in the system.

   ```
   # pkg set-publisher -g /tmp/install/VRTSvcswas.p5p Veritas
   ```
4. Install the package.

   ```
   # pkg install --accept VRTSvcswas
   ```
5. Remove the publisher from the system.

   ```
   # pkg unset-publisher Veritas
   ```
6. Enable the publishers that were disabled earlier.

   ```
   # pkg set-publisher --enable <publisher name>
   ```

Installing agent packages on Solaris brand non-global zones

To install the agent package on Solaris brand non-global zones

1. Ensure that the SMF services, `svc:/application/pkg/system-repository:default` and `svc:/application/pkg/zones-proxyd:default`, are online on the global zone.

   ```
   # svcs svc:/application/pkg/system-repository:default
   ```

   ```
   # svcs svc:/application/pkg/zones-proxyd:default
   ```
2. Log on to the non-global zone as a superuser.

3. Ensure that the SMF service `svc:/application/pkg/zones-proxy-client:default` is online inside non-global zone:

   ```
   # svcs svc:/application/pkg/zones-proxy-client:default
   ```

4. Copy the `VRTSvcswas.p5p` package from the `pkgs` directory to the non-global zone (for example, at the `/tmp/install` directory).

5. Disable the publishers that are not reachable, as package install may fail, if any of the already added repositories are unreachable.

   ```
   # pkg set-publisher --disable <publisher name>
   ```

6. Add a file-based repository in the non-global zone.

   ```
   # pkg set-publisher -g/tmp/install/VRTSvcswas.p5p Veritas
   ```

7. Install the package.

   ```
   # pkg install --accept VRTSvcswas
   ```

8. Remove the publisher on the non-global zone.

   ```
   # pkg unset-publisher Veritas
   ```

9. Clear the state of the SMF service, as setting the file-based repository causes the SMF service `svc:/application/pkg/system-repository:default` to go into the maintenance state.

   ```
   # svcadm clear svc:/application/pkg/system-repository:default
   ```

10. Enable the publishers that were disabled earlier.

    ```
    # pkg set-publisher --enable <publisher>
    ```

Note: Perform steps 2 through 10 on each non-global zone.

Installing the agent in a Solaris 10 brand zone

To install the WebSphere Application Server agent in a brand zone on Solaris 10:

- Ensure that the ACClibrary package, `VRTSacclib`, is installed in the non-global zone.

  To install VRTSacclib in the non-global zone, run the following command from the global zone:

  ```
  # pkgadd -R /zones/zone1/root -d VRTSacclib.pkg
  ```

- To install the agent package in the non-global zone, run the following command from the global zone:

  ```
  # pkgadd -R zone-root/root -d . VRTSvcswas
  ```

For example: `# pkgadd -R /zones/zone1/root -d . VRTSvcswas`

Note: You can ignore the following messages that might appear:

```
## Executing postinstall script.

ln: cannot create
/opt/VRTSagents/ha/bin/WebSphere/imf_getnotification: File exists

ln: cannot create /opt/VRTSagents/ha/bin/WebSphere/imf_register: File
exists

or ## Executing postinstall script.

ln: cannot create
/opt/VRTSagents/ha/bin/WebSphere/imf_getnotification: No such file
or directory

ln: cannot create /opt/VRTSagents/ha/bin/WebSphere/imf_register: No
such file or directory
```

Uninstalling the agent in a VCS environment

You must uninstall the agent for WebSphere Application Server from a cluster while the cluster is active.

To uninstall the agent in a VCS environment

1. Log in as a superuser.
2. Set the cluster configuration mode to read/write by running the following command from any node in the cluster:

   ```
   # haconf -makerw
   ```

3. Remove all WebSphere Application Server resources from the cluster. Run the following command to verify that all resources have been removed:

   ```
   # hares -list Type=WebSphere
   ```

4 Remove the agent type from the cluster configuration by running the following command from any node in the cluster:

```
# hatype -delete WebSphere
```

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 running the following command from any node in the cluster:

```
# haconf -dump -makero
```

6 Use the platform's native software management program to remove the agent for WebSphere Application Server from each node in the cluster.

Run the following command to uninstall the agent:

AIX	`# installp -u VRTSvcswas.rte`
Linux	`# rpm -e VRTSvcswas`

Solaris	`# pkgrm VRTSvcswas`
	Note: To uninstall the agent IPS package on a Solaris 11 system, run the following command:
	`# pkg uninstall VRTSvcswas`

Removing the ACC library

Perform the following steps to remove the ACC library.

To remove the ACC library

1. Ensure that all agents that use ACC library are removed.
2. Run the following command to remove the ACC library package:

AIX

```
# installp -u VRTSacclib
```

Linux

```
# rpm -e VRTSacclib
```

Solaris

```
# pkgrm VRTSacclib
```

Note: To uninstall the ACCLib IPS package on a Solaris 11 system, run the following command:

```
# pkg uninstall VRTSacclib
```

Upgrading the agent in a VCS environment

Perform the following steps to upgrade the agent with minimal disruption, in a VCS environment.

Updating WebSphere5 agent to the WebSphere agent

To update the WebSphere5 agent to the WebSphere agent

You can upgrade the agent by one of the following ways:

- Updating the agent using update script
  See "Updating the agent using the update script" on page 23.
- Updating the agent by changing the main.cf file
  See "Updating the agent by changing the main.cf file " on page 24.

Updating the agent using the update script

Perform the following steps to update the agent using the update script.

To update the agent using update script

1. Install the WebSphere Application Server agent package (VRTSvcswas) on all nodes.

   See "Installing the agent in a VCS environment" on page 18.

2. Navigate to the agent source directory:

VCS4.x	cd /opt/VRTSvcs/bin/WebSphere
VCS5.x	cd /opt/VRTSagents/ha/bin/WebSphere

3. Run the update script available in the agent directory.

   ```
   # ./update.pl -o WebSphere5 -n WebSphere
   ```

-o	--oldType	Present resource type
-n	--newType	New resource type

   If the script runs successfully, it indicates that the old agent is updated with the new agent type. Hence, go to step 4 otherwise go to step 5.

4. Use the platform's native software management program to remove the old agent package (VRTSvcswas5) for WebSphere Application Server from each node in the cluster.

   Run the following command to uninstall the agent:

AIX	# installp -u VRTSvcswas5.rte
HP-UX	# swremove VRTSvcswas5
Linux	# rpm -e VRTSvcswas5
Solaris	# pkgrm VRTSvcswas5

5. If the update is not successful then restore the main.cf from the backup mentioned in log file and try to update the agent using the steps mentioned in following section.

   Updating the agent by changing the main.cf file

Updating the agent by changing the main.cf file

Perform the following steps to update the agent by updating the main.cf file.

To update the agent changing the main.cf file

1. Install the WebSphere Application Server agent package (VRTSvcswas) on all nodes.

   See "Installing the agent in a VCS environment" on page 18.

2. Persistently freeze the service groups that host the application.

   ```
   # hagrp -freeze GroupName -persistent
   ```

3. Import the WebSphere Application Server agent type file.

   See "Importing the agent types files in a VCS environment" on page 29.

4. Dump the current VCS configuration.

   ```
   # haconf -dump
   ```

5. Stop the cluster services forcibly.

   ```
   # hastop -all -force
   ```

6. Ensure that the agent operations are stopped on all the nodes.

   ```
   # ps -ef | grep WebSphere5
   ```

   ```
   # ps -ef | grep WebSphere
   ```

7. Edit the `main.cf` file by performing the following steps:

   - Delete the following entry:

     ```
     include "WebSphere5Types.cf"
     ```

   - Replace WebSphere5 with WebSphere throughout the main.cf file.
   - Save the file.

   Note: The `WebSphere5Types.cf` file varies according to platform and VCS version.

8. Start the cluster services.

   ```
   # hastart
   ```

9. Start the agent on all nodes, if not started.

   ```
   # haagent -start WebSphere -sys SystemName
   ```

10 Unfreeze the service groups once all the resources come to an online steady state.

```
# hagrp -unfreeze GroupName -persistent
```

11 Use the platform's native software management program to remove the old agent package for WebSphere Application Server from each node in the cluster.

Run the following command to uninstall the agent:

AIX	# installp -u VRTSvcswas5.rte
HP-UX	# swremove VRTSvcswas5
Linux	# rpm -e VRTSvcswas5
Solaris	# pkgrm VRTSvcswas5

Updating the WebSphere agent to latest release

Perform the following steps to upgrade the agent with minimal disruption, in a VCS environment

1 Persistently freeze the service groups that host the application.

```
# hagrp -freeze group -persistent
```

2 If the agent is running, stop the agent.

```
# haagent -stop WebSphere -force -sys hostname
```

3 Ensure that the agent operations are stopped on all the nodes.

```
# ps -ef |grep WebSphere
```

4 Upgrade the agent.

- Linux `# rpm -Uvh VRTSvcswas-agentVersion-GENERIC.noarch.rpm`
- For AIX and Solaris:
  See “To uninstall the agent in a VCS environment” on page 21.
  See “To install the agent in a VCS environment” on page 18.
  - Copy the new WebSphereTypes.cf file from the agent's conf directory, to the VCS conf directory /etc/VRTSvcs/conf/config.

VCS 4.x	■ AIX ■ Solaris	`/etc/VRTSvcs/conf/sample_WebSphere/WebSphere.cf`

VCS 5.x	■ AIX	`/etc/VRTSagents/ha/conf/WebSphere/WebSphereTypes.cf`
VCS 5.0	Solaris SPARC	`/etc/VRTSagents/ha/conf/WebSphere/WebSphereTypes50.cf`
VCS 5.1	Solaris SPARC	`/etc/VRTSagents/ha/conf/WebSphere/WebSphereTypes51.cf`

- Check for the changes in the resource values required, if any, due to the new agent types file.

5 Start the agent.

```
# haagent -start WebSphere -sys System
```

6 Unfreeze the service groups once all the resources come to an online steady state.

```
# hagrp -unfreeze GroupName -persistent
```

Chapter 3

Configuring the agent for WebSphere Application Server

This chapter includes the following topics:

Before configuring the agent for WebSphere Application Server for the ServerType cluster

Before you configure the Cluster Server agent for WebSphere Application Server for the ServerType cluster, you must:

- Increase the value of the MonitorTimeout attribute to 120, if the second level monitoring is enabled for the resource. The monitoring time vary proportionally to the number of cluster members when the second level monitoring is enabled.
- Ensure that the cluster member name is unique.

- The WAS Home should be same across all the host operating systems, where the WebSphere Application Server cluster is set up.
- Set up passwordless SSH connection across all the host operating systems, where the WebSphere Application Server cluster is set up.

About configuring the Cluster Server agent for WebSphere Application Server

After installing the Cluster Server agent for WebSphere Application Server, you must import the agent type configuration file. After importing this file, review the attributes table that describes the resource type and its attributes, and then create and configure WebSphere Application Server resources.

To view the sample agent type definition and service groups configuration:

See "About sample configurations for the agents for WebSphere Application Server" on page 71.

Importing the agent types files in a VCS environment

To use the agent for WebSphere Application Server, you must import the agent types file into the cluster. You can import the agent types file using the VCS graphical user interface or using the command line interface.

To import the agent types file using the VCS Java GUI

1. Start the Cluster Manager (Java Console) 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:

VCS 4.x	■ AIX ■ Linux ■ Solaris	`/etc/VRTSvcs/conf/sample_WebSphere/WebSphereTypes.cf`
VCS 5.x or later	■ AIX ■ Linux	`/etc/VRTSagents/ha/conf/WebSphere/WebSphereTypes.cf`
VCS 5.0	Solaris SPARC	`/etc/VRTSagents/ha/conf/WebSphere/WebSphereTypes50.cf`

VCS 5.1 or later	Solaris SPARC	`/etc/VRTSagents/ha/conf/WebSphere/` `WebSphereTypes51.cf`

4. Click **Import**.
5. Save the VCS configuration.

The WebSphere Application Server agent type is now imported to the VCS engine.

You can now create WebSphere Application Server resources. For additional information about using the VCS GUI, refer to the *Cluster Server Administrator's Guide*.

To import the agent types file using the CLI

1. Log on to any one of the systems in the cluster as the superuser.
2. Create a temporary directory.

```
# mkdir ./temp
```

```
# cd ./temp
```

3. Copy the sample file `Types.cf`.

VCS 4.x	AIX Linux Solaris	`/etc/VRTSvcs/conf/sample_WebSphere/` `WebSphereTypes.cf`
VCS 5.x or later	AIX Linux	`/etc/VRTSagents/ha/conf/WebSphere/` `WebSphereTypes.cf`
VCS 5.0	Solaris SPARC	`/etc/VRTSagents/ha/conf/WebSphere/` `WebSphereTypes50.cf`
VCS 5.1 or later	Solaris SPARC	`/etc/VRTSagents/ha/conf/WebSphere/` `WebSphereTypes51.cf`

The following example assumes VCS 5.0 is installed on Solaris:

```
# cp /etc/VRTSagents/ha/conf/WebSphere/WebSphereTypes.cf .
```

4. Create a dummy `main.cf` file.

   ```
   # echo 'include "WebSphereTypes.cf"' > main.cf
   ```

5. Create the WebSphere resource type as follows:

   ```
   # hacf -verify .

   # haconf -makerw

   # sh main.cmd

   # haconf -dump
   ```

The WebSphere Application Server agent type is now imported to the VCS engine. You can now create WebSphere Application Server resources. For additional information about using the VCS CLI, refer to the *Cluster Server Administrator's Guide*.

Importing the agent types file using Cluster Manager (Java Console)

To import the agent types file using Cluster Manager (Java Console)

1. Start the Cluster Manager (Java Console) 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:

VCS 4.x	■ AIX ■ Linux ■ Solaris	`/etc/VRTSvcs/conf/sample_WebSphere/WebSphereTypes.cf`
VCS 5.x or later	■ AIX ■ Linux	`/etc/VRTSagents/ha/conf/WebSphere/WebSphereTypes.cf`
VCS 5.0	Solaris SPARC	`/etc/VRTSagents/ha/conf/WebSphere/WebSphereTypes50.cf`
VCS 5.1 or later	Solaris SPARC	`/etc/VRTSagents/ha/conf/WebSphere/WebSphereTypes51.cf`

4. Click **Import**.
5. Save the VCS configuration.

   The WebSphere Application Server agent type is now imported to the VCS engine.

   You can now create WebSphere Application Server resources.

Importing the agent types file using the CLI

To import the agent types file using the command line interface (CLI):

1. If VCS is running, run the `/etc/VRTSagents/ha/conf/WebSphere/WebSphereTypes.cmd` file from the command line.
2. If VCS is not running, perform the following steps:

   1. Copy the agent types file from the `/etc/VRTSagents/ha/conf/<AgentTypes_file>` directory to the `/etc/VRTSvcs/conf/config` directory.

   Where, <AgentTypes_file> is chosen according to the following table:

VCS 4.x	■ AIX ■ Linux ■ Solaris	`/etc/VRTSvcs/conf/sample_WebSphere/WebSphereTypes.cf`
VCS 5.x or later	■ AIX ■ Linux	`/etc/VRTSagents/ha/conf/WebSphere/WebSphereTypes.cf`
VCS 5.0	Solaris SPARC	`/etc/VRTSagents/ha/conf/WebSphere/WebSphereTypes50.cf`
VCS 5.1 or later	Solaris SPARC	`/etc/VRTSagents/ha/conf/WebSphere/WebSphereTypes51.cf`

   2. Include the agent types file in the main.cf file.

   3. Start HAD.

WebSphere Application Server agent attributes

Table 3-1 shows the required attributes for the agent for WebSphere Application Server.

Table 3-1 Required attributes

Required attribute	Description
ServerName	The server name assigned to the WebSphere Server during its installation. In Network Deployment configurations, the default ServerName for Deployment Managers is dmgr and the default ServerName for the Node Agents is nodeagent, but these names are not mandatory. For the Cluster server type, specify the cluster name as the value for the ServerName attribute. See “Uniquely identifying WebSphere Application Server instances” on page 40. Type and dimension: string-scalar Default: No default value Example: server1
ServerProfile	Server profile name of the WebSphere Server instance or complete path to the WebSphere Application Server profile. If a profile is installed in a non-default location, provide the complete path to the WebSphere Application Server Profile. If a profile is installed in a default location, provide only the profile name. For the Cluster server type, specify the ServerProfile attribute as Deployment Manager that manages the entire WAS cluster. This attribute is applicable to WebSphere version 6.0 and later, and must be null if the WebSphere major version number is 5. You must specify this attribute if the resource manages a WebSphere Application Server version 6.0 and later. Type and dimension: string-scalar Default: No default value Example 1: Dmgr01 Example 2: `/WAS/AppSrv/profiles/AppSrv01/`

Table 3-1 Required attributes *(continued)*

Required attribute	Description
ServerType	The type of WebSphere Application Server that the cluster will manage. Valid names are as follows: ■ DeploymentManager: Resource is a Deployment Manager. ■ NodeAgent: Resource is a Node Agent. ■ ApplicationServer: Resource is an Application Server, which may be a stand-alone server or may be part of a Network Deployment and is a member of a WebSphere Cell. ■ Cluster: Resource is a WebSphere Application Server Cluster, which is a collection of servers that are managed together. The agent uses this value to determine how to manage the WebSphere Application Server within a cluster. Refer to the WebSphere documentation for a full explanation of the purposes and use of each WebSphere Application Server type. Type and dimension: string-scalar Default: No default value Example: DeploymentManager
User	The UNIX username used to run the programs that start, stop, and monitor the WebSphere resource, which include the program specified in the MonitorProgram attribute. IBM recommends using the root account, but you may use any account. If User is not set to root, the username must be synchronized across the systems within the cluster. In other words, the user name must resolve to the same UID and have the same default shell on each system in the cluster. Type and dimension: string-scalar Default: No default value Example: root

Table 3-1 Required attributes *(continued)*

Required attribute	Description
WAS_HOME	The absolute path to the WebSphere Application Server or WebSphere Application Server Network Deployment root installation directory. This attribute is used to locate programs executed by the agent. It is also where the *`binDir`*`/setupCmdLine.sh` file resides. The value is also used to uniquely identify the ServerType processes. Using WAS_HOME to uniquely identify an Application Server's process IDs requires that WAS_HOME be unique compared to WAS_HOME for all other WAS instances in the cluster. See "Uniquely identifying WebSphere Application Server instances" on page 40. **Note:** Both WAS_HOME and WAS_ND_HOME are defined as WAS_HOME in the standard environment file `setupCmdLine.sh`, which is supplied with WebSphere. Type and dimension: string-scalar Default: No default value Example: `/ibm/was/v51/cell1/node2`
WAS_NODE	The WebSphere node name to which the server instance belongs. The node name is an administrative identifier that is internal to the WebSphere environment and is assigned when the node is installed. WebSphere requires that a node name must be unique within a WebSphere cell. For the Cluster server type, specify the Deployment Manager node name for the WAS_NODE attribute. See "Uniquely identifying WebSphere Application Server instances" on page 40. Type and dimension: string-scalar Default: No default value Example: was51c1n2

Table 3-2 lists the optional attributes for the agent for WebSphere Application Server.

Table 3-2 Optional attributes

Optional Attribute	Definition
ResLogLevel	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 functions. Type and dimension: string-scalar Default: INFO Example: TRACE **Note:** The use of the ResLogLevel attribute is deprecated from VCS version 6.2 onwards. You must use the LogDbg attribute instead of the ResLogLevel attribute to enable debug logs for the ACCLib-based agents, when the ACCLib version is 6.2.0.0 or later. The agent captures the first failure data of the unexpected events and automatically logs debug messages in their respective agent log files.
LogDbg	For ACCLib-based agents, you must use the LogDbg resource type attribute to enable the debug logs when the ACCLib version is 6.2.0.0 or later and the VCS version is 6.2 or later. Set the LogDbg attribute to DBG_5 to enable debug logs for ACCLIB based agent. By default, setting the LogDbg attribute to DBG_5 enables debug logs for all WebSphere resources in the cluster. If debug logs must be enabled for a specific WebSphere resource, override the LogDbg attribute. Type and dimension: string-keylist Default: No default value For more information on how to use the LogDbg attribute, refer to the *Cluster Server Administrator's Guide*.

Table 3-2 Optional attributes *(continued)*

Optional Attribute	Definition
MonitorProgram	The full pathname and command-line arguments for an externally-provided custom monitor program. The program is executed within the security context of the UNIX account specified in the User attribute. The program must be completely self-contained and independent, and it must return one of the following exit codes: ■ 110 or 0: The WebSphere Application Server is ONLINE. ■ 100 or 1: The WebSphere Application Server is OFFLINE. ■ All other: The WebSphere Application Server state is UNKNOWN. Veritas recommends storing the external monitor program on the shared storage device, in the directory specified by the WAS_HOME attribute, to ensure the file is always available on the online system. Type and dimension: string-scalar Default: No default value Example: `/usr/WAS51/server1/bin/mymonitor.sh`
SecondLevelMonitor	Specifies if second-level monitor is enabled and how frequently it is performed. Second-level monitor is a deeper, more thorough state check of the WebSphere resource, performed by executing the IBM-supplied utility program `serverStatus.sh`. The output from this program is parsed to confirm the server status is running. The integer value specified by this attribute determines how frequently the second-level monitor program is executed. For example, if SecondLevelMonitor is set to 1, the monitor function will execute `serverStatus.sh` during each monitor interval. A value of 3 executes the program every third monitor interval. If SecondLevelMonitor is set to 0, the monitor function will never perform the second-level monitor. **Note:** The SecondLevelMonitor attribute is applicable to VCS versions earlier than VCS 5.1 SP1 with WebSphere agent versions earlier than 5.1.9.0. From VCS version 5.1 SP1 with WebSphere agent version 5.1.9.0 onwards, the SecondLevelMonitor attribute of the WebSphere agent is deprecated. Instead, a resource type level attribute LevelTwoMonitorFreq should be used to specify the frequency of in-depth monitoring. Type and dimension: integer-scalar Default: 0 Example: 1

Table 3-2 Optional attributes *(continued)*

Optional Attribute	Definition
LevelTwoMonitorFreq	The frequency at which the agent for this resource type must perform second-level or detailed monitoring. You can also override the value of this attribute at the resource level. The value indicates the number of monitor cycles after which the agent will monitor WebSphere in detail. For example, the value 5 indicates that the agent will monitor WebSphere in detail after every five online monitor intervals. **Note:** This attribute is applicable to VCS version 5.1 SP1 with WebSphere agent version 5.1.9.0 or later. If the VCS version is earlier than VCS 5.1 SP1 and the WebSphere agent version is earlier than 5.1.9.0, the SecondLevelMonitor attribute should be used. If you upgraded the VCS version to VCS 5.1 SP1 and the WebSphere agent version to 5.1.9.0 (or later), and if you had enabled detail monitoring in the previous version, then do the following: ■ Set the value of the LevelTwoMonitorFreq attribute to the same value as that of the SecondLevelMonitor attribute. Type and dimension: integer-scalar Default: 0
StartOptions	The command-line options that are passed to the WebSphere start script when it is executed within the online function. Multiple options should be separated by a space. Refer to the WebSphere product documentation for a list and description of supported start options. Type and dimension: string-scalar Default: No default value Example: "-replacelog -trace"

Table 3-2 Optional attributes *(continued)*

Optional Attribute	Definition

StopOptions	The command-line options that are passed to the WebSphere stop script when it is executed within the offline function. Multiple options should be separated by a space. Refer to the WebSphere product documentation for a list and description of supported stop options.
	Type and dimension: string-scalar
	Note: The old AdminUserName and AdminPassword attributes are deprecated. If required, you can specify them as part of StopOptions. Instead, Veritas recommends that you follow the security measures as directed by IBM in this support article:
	https://www.ibm.com/support/knowledgecenter/en/SSFHUY_1.0.6/deploy/configure_the_soap.client.properties_file.html
	Specifically, edit the `/WebSphere/AppServer/profiles/AppSrv01/properties/soap.client.props` file, and change the values of the following properties:
	`com.ibm.SOAP.securityEnabled=true`
	`com.ibm.SOAP.loginUserid=wasadmin`
	`com.ibm.SOAP.loginPassword=wasadmin`
	where, `com.ibm.SOAP.loginUserid` is the name of the Web Application Server administrator user and `com.ibm.SOAP.loginPassword` is the corresponding password.
	Default: No default value
	Example: "-replacelog -trace"

DMHostName	The hostname of the Deployment Manager, which manages the WebSphere Application Server cluster. Type and dimension: string-scalar Default: No default value Example: “vhost1.xyz.com"
DMPort	The SOAP Connector Port of the Deployment Manager, which manages the WebSphere Application Server cluster. Type and dimension: string-scalar Default: No default value Example: 8881

Table 3-2 Optional attributes *(continued)*

Optional Attribute	Definition
MinClusterMember	The minimum number of cluster members that should be running for considering the WebSphere Application Server cluster as online. Type and dimension: integer-scalar Default: 1 Example: 2

Note: For information about the additional attributes of the agent for WebSphere Application Server when IMF is enabled: See "Attributes that enable IMF" on page 48.

Uniquely identifying WebSphere Application Server instances

You can virtualize a WebSphere Application Server instance using a cluster. Using shared disk and virtual IP addresses, you can manage a large set of WebSphere Application Server instances in a single cluster.

Set the WAS_HOME, WAS_NODE, and ServerName attributes such that the combined values are unique for each WebSphere Application Server instance.

WebSphere Application Servers can run on separate cluster nodes or can run concurrently on a single node. If WebSphere Application Servers run concurrently on a single node, you must ensure that the agent can uniquely identify each WebSphere Application Server on a host system that is running more than one WebSphere Application Server.

For unique identification, the agent's monitor and clean functions use the values specified by attributes WAS_HOME, WAS_NODE, and ServerName to uniquely identify each running WebSphere Server JVM process.

Differentiating WebSphere Application 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 WebSphere Application Server could result in a clean operation that kills processes for more than one WebSphere Application Server instance.

Important considerations while configuring the agent

While configuring the agent, make the following settings:

- The time required to fully start a WebSphere instance depends on the number, size, and complexity of Java applications started within the server. Be sure to compare the value of the OnlineTimeout attribute with the actual time required to fully initialize the WebSphere Application Server. Large WebSphere Application Server deployments may require a larger OnlineTimeout. Properly tuning this attribute ensures that the cluster does not time out the online entry point while a WebSphere Application Server is initializing.
- Allow sufficient time for the WebSphere Application Server to shut down completely before probing the resource to determine if the request to stop was successful. Depending upon the environment, you may need to adjust the OfflineTimeout attribute for this resource to allow the instance ample time to shut down. Properly tuning this attribute ensures that the cluster does not time out the offline entry point while a WebSphere Application Server is completing a graceful shut down.

After a WebSphere Application Server is placed under cluster control, do not attempt to start or stop the instance without using a cluster interface. Only use the Web Console, Java Console, or command-line interface, to start or stop a managed WebSphere instance.

Service group configuration options

The WebSphere deployment type and strategy determines the number of service groups in a cluster required and the number of WebSphere Application Servers managed within each service group. Although not comprehensive, the following examples depict common scenarios to consider.

Figure 3-1 depicts a service group that manages a Deployment Manager Server.

Other service groups manage WebSphere Servers of the type Node agent and Application Server.

Figure 3-1 Service group that manages a Deployment Manager Server

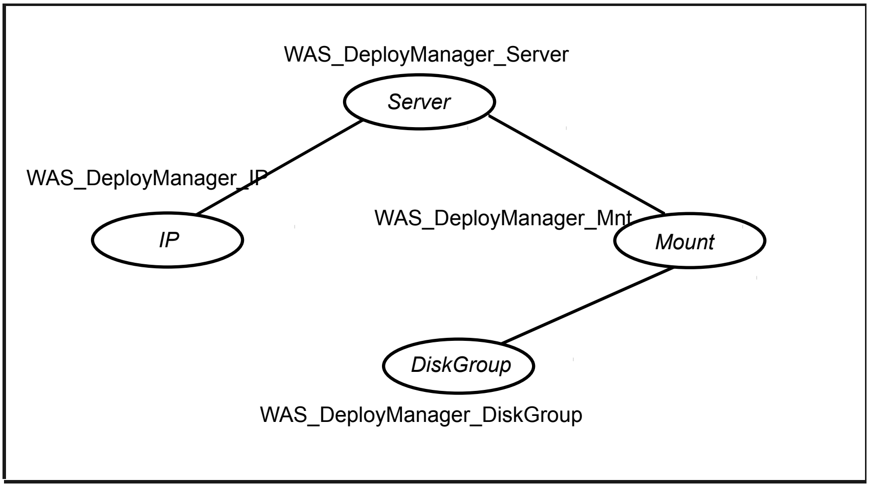


Figure 3-2 depicts a service group that manages a Node Agent Server.

In this configuration, the cluster does not control the Application Servers managed by this Node Agent instance. Thus, the Node Agent Server may fully manage and monitor its managed Application Servers without conflict with the cluster.

Figure 3-2 WebSphere Application Server NodeAgent on all systems

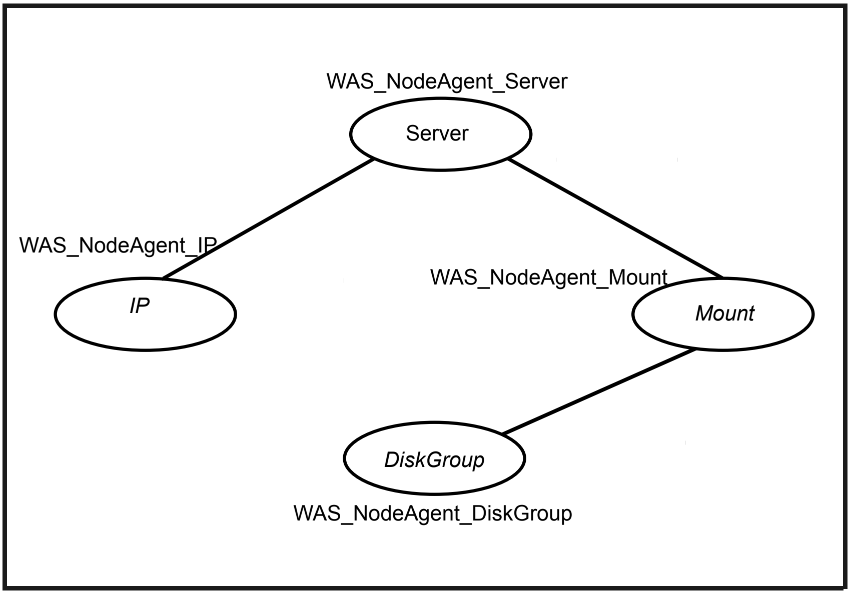


Figure 3-3 depicts a service group that controls a Node Agent Server and its two managed Application Servers.

In this configuration, the cluster controls the Application Servers that are managed by this Node Agent instance. Thus, the Node Agent Server should be configured to not monitor and restart its failed Application Servers, as this would conflict with cluster actions in response to the failure.

Figure 3-3 Service group that controls a Node Agent Server and its two managed Application Servers

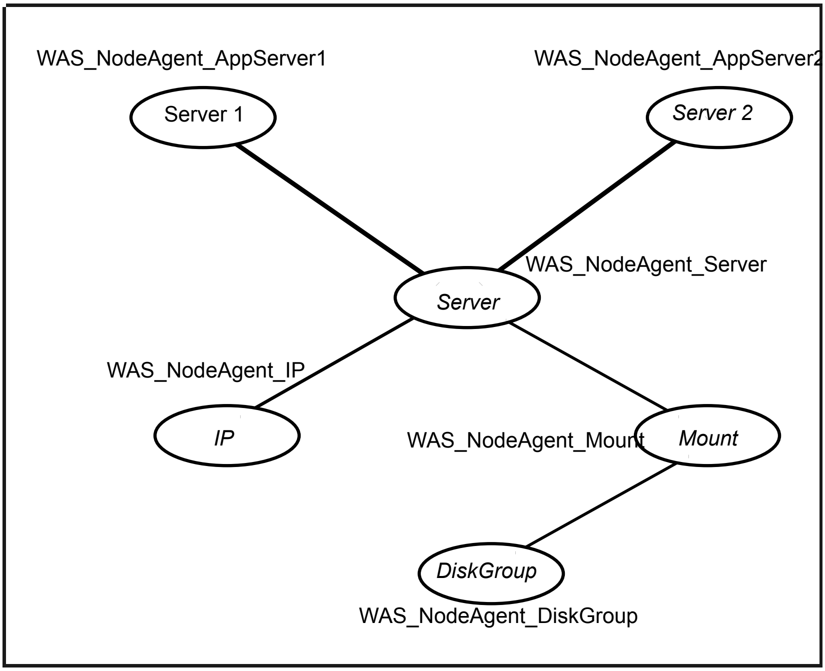


Figure 3-4 depicts service group dependencies for the WebSphere Application Server Cluster service group.

Figure 3-4 Service group dependency

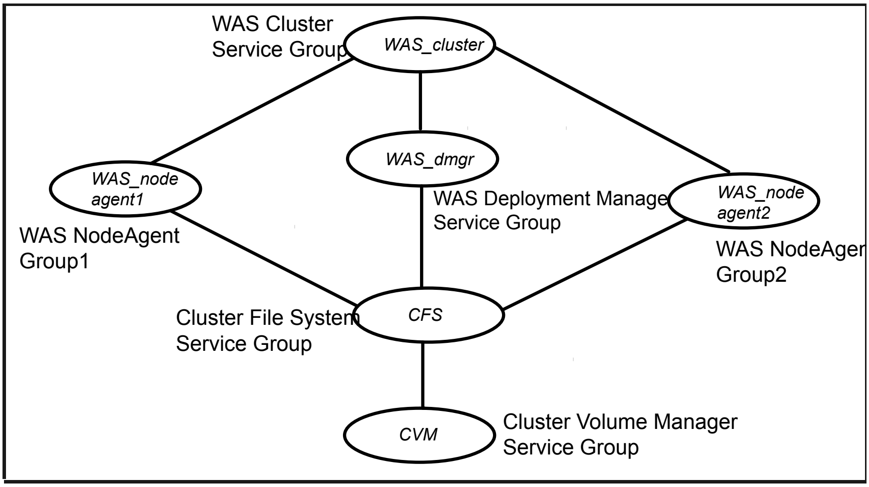


Figure 3-5 depicts the resource dependencies for the WebSphere Application Server Deployment Manager resource.

Figure 3-5 Resource dependency for Deployment Manager

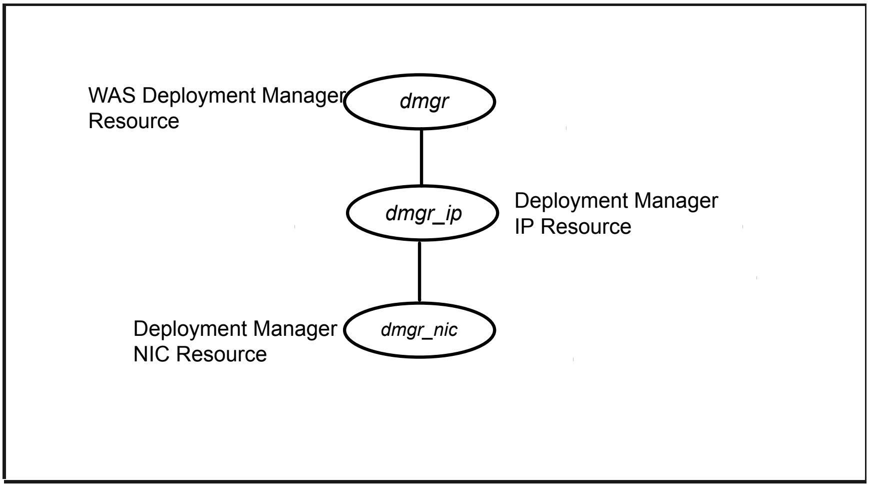


Figure 3-6 depicts the resource dependency for the CFS service group.

Figure 3-6 Resource dependency for CFS

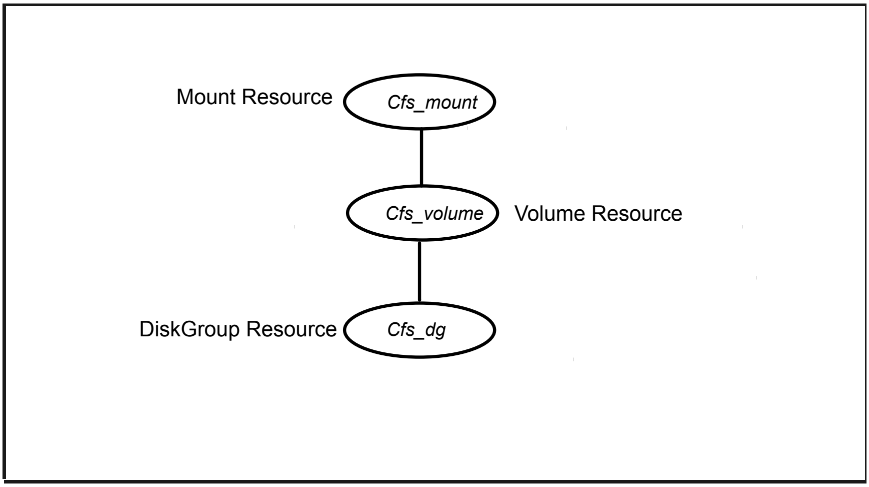
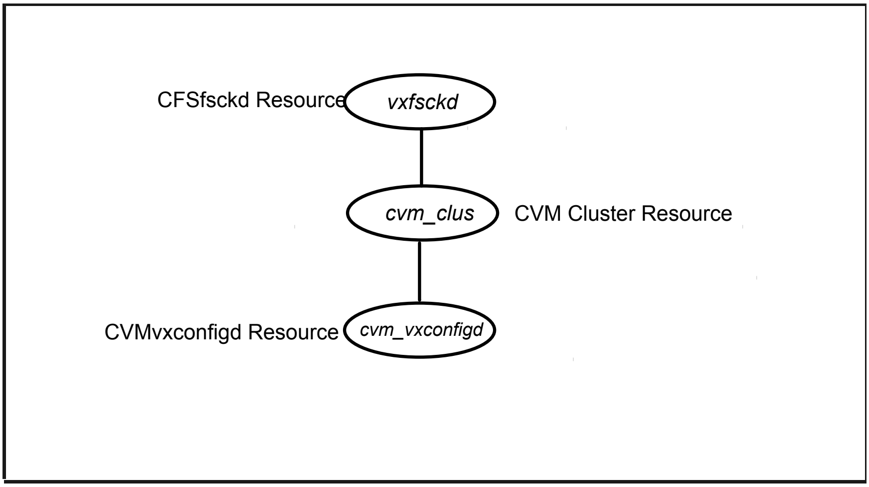


Figure 3-7 depicts the resource dependency for the CVM service group.

Figure 3-7 Resource dependency for CVM



Enabling the agent for WebSphere Application Server to support IMF

This chapter includes the following topics:

About Intelligent Monitoring Framework

With the IMF feature, VCS supports intelligent resource monitoring in addition to the poll-based monitoring. Poll-based monitoring polls the resources periodically whereas intelligent monitoring performs asynchronous monitoring. You can enable or disable the intelligent resource monitoring functionality of the WebSphere Application Server agent.

VCS process and mount-based agents use the AMF kernel driver that provides asynchronous event notifications to the agents that are enabled for IMF.

You can enable the WebSphere Application Server agent for IMF, provided the following software versions are installed:

- Cluster Server (VCS) 5.1 SP1 or later

- Cluster Server agent for WebSphere Application Server version 5.1.0.0 or later

Refer to the *Cluster Server Administrator’s Guide* for more information about IMF notification module functions and administering the AMF kernel driver.

Benefits of IMF

IMF offers the following benefits:

- Performance
  Enhances performance by reducing the monitoring of each resource at a default of 60 seconds for online resources, and 300 seconds for offline resources. IMF enables the agent to monitor a large number of resources with a minimal effect on performance.
- Faster detection
  Asynchronous notifications would detect a change in the resource state as soon as it happens. Immediate notification enables the agent to take action at the time of the event.

Agent functions for the IMF functionality

If the WebSphere Application Server agent is enabled for IMF support, the agent supports the following functions, in addition to the functions mentioned in the WebSphere Application Server agent functions topic.

imf_init

This function initializes the WebSphere Application Server agent to interface with the AMF kernel driver, which is the IMF notification module for the agent for WebSphere Application Server. This function runs when the agent starts up.

imf_getnotification

This function gets notifications about resource state changes. This function runs after the agent initializes with the AMF kernel module. This function continuously waits for notification and takes action on the resource upon notification.

imf_register

This function registers or unregisters resource entities with the AMF kernel module. This function runs for each resource after the resource goes into a steady state—online or offline.

Attributes that enable IMF

If the agent for WebSphere Application Server is enabled for IMF support, the agent uses type-level attributes in addition to the agent-specific attributes.

See “WebSphere Application Server agent attributes” on page 32.

IMF

This resource type-level attribute determines whether the WebSphere Application Server agent must perform intelligent resource monitoring. You can also override the value of this attribute at the resource level.

This attribute includes the following keys:

Mode

Define this attribute to enable or disable intelligent resource monitoring.

The valid values are as follows:

- 0—Does not perform intelligent resource monitoring
- 1—Performs intelligent resource monitoring for offline resources and performs poll-based monitoring for online resources
- 2—Performs intelligent resource monitoring for online resources and performs poll-based monitoring for offline resources
- 3—Performs intelligent resource monitoring for both online and for offline resources.

Note: The agent supports intelligent resource monitoring for online resources only. Hence, Mode should be set to either 0 or 2.

Type and dimension: integer-association

Default: 0 for VCS 5.1 SP1, 3 for VCS 6.0 and later.

MonitorFreq

This key value specifies the frequency at which the agent invokes the monitor agent function. The value of this key is an integer.

Default: 1

You can set this key to a non-zero value for cases where the agent requires to perform both poll-based and intelligent resource monitoring.

If the value is 0, the agent does not perform poll-based process check monitoring.

After the resource registers with the AMF kernel driver, the agent calls the monitor agent function as follows:

- After every (MonitorFreq x MonitorInterval) number of seconds for online resources
- After every (MonitorFreq x OfflineMonitorInterval) number of seconds for offline resources

RegisterRetryLimit

If you enable intelligent resource monitoring, the agent invokes the imf_register agent function to register the resource with the AMF kernel driver.

The value of the RegisterRetryLimit key determines the number of times the agent must retry registration for a resource. If the agent cannot register the resource within the limit that is specified, then intelligent monitoring is disabled until the resource state changes or the value of the Mode key changes.

Default: 3.

IMFRegList

An ordered list of attributes whose values are registered with the IMF notification module.

Type and dimension: string-vector

Default: No default value

Note: The attribute values can be overriden at the resource level.

Before you enable the agent to support IMF

Before you enable the WebSphere Application Server agent to support IMF, ensure that the AMF kernel module is loaded and AMF is configured. For details, refer to the ‘Administering the AMF kernel driver' section of the *Cluster Server Administrator's Guide*. For details about the commands you can configure AMF using the `amfconfig -h` command.

Enabling the agent to support IMF

In order to enable the WebSphere Application Server agent to support IMF, you must make the following configuration changes to the attributes of the agent:

- AgentFile: Set the AgentFile attribute to **Script51Agent** or **Script60Agent** as appropriate for your agent version
- IMF Mode: Set the IMF Mode attribute to **2**
- IMFRegList: Update the IMFRegList attribute

The following sections provide more information about the commands you can use to make these configuration changes, depending on whether VCS is in a running state or not.

Note: If you have upgraded VCS from an earlier version to version 5.1 SP1 or later, and you already have version 5.1.00 or later of the agent installed, ensure that you run the following commands to create appropriate symbolic links:

```
# cd /opt/VRTSagents/ha/bin/<resourceType>
```

```
# ln -s /opt/VRTSamf/imf/imf_getnotification imf_getnotification
```

```
# ln -s /opt/VRTSagents/ha/bin/<resourceType>/monitor imf_register
```

If VCS is in a running state

To enable the resource for IMF when VCS is in a running state

1 Make the VCS configuration writable.

```
# haconf -makerw
```

2 Run the following command to update the AgentFile attribute.

```
# hatype -modify <resourceType> AgentFile\
/opt/VRTSvcs/bin/Script51Agent
```

3. For VCS version 6.0 or later, run the following commands to add the IMF attributes:

   ```
   # haattr -add -static  <resourceType> IMF -integer -assoc Mode 0 \
   MonitorFreq 1 RegisterRetryLimit 3
   ```

   ```
   # haattr -add -static <resourceType> IMFRegList -string -vector
   ```

   Note: Run these commands only once after you first enable IMF support for the agent.

   ---

4. Run the following command to update the IMF attribute.

   ```
   # hatype -modify <resourceType> IMF Mode num MonitorFreq num
   RegisterRetryLimit num
   ```

   For example, to enable intelligent monitoring of online resources, with the MonitorFreq key set to 5, and the RegisterRetryLimit key is set to 3, run the following command:

   ```
   # hatype -modify <resourceType> IMF Mode 2 MonitorFreq 5 \
   RegisterRetryLimit 3
   ```

   Note: The valid values for the Mode key of the IMF attribute are 0 (disabled) and 2 (online monitoring).

   ---

5. Run the following command to update the IMFRegList attribute:

   ```
   # hatype -modify WebSphere IMFRegList WAS_NODE WAS_HOME ServerName
   ```

6. Save the VCS configuration.

   ```
   # haconf -dump -makero
   ```

7. If the agent is running, restart the agent.

   For information on the commands you can use to restart the agent, see Restarting the agent.

Restarting the agent

To restart the agent:

1. Run the following command to stop the agent forcefully:

   ```
   # haagent -stop <resourceType> -force -sys <systemName>
   ```

   Note: Stopping the agent forcefully eliminates the need to take the resource offline.

   ---

2. Run the following command to start the agent:

   ```
   # haagent -start <resourceType> -sys <systemName>
   ```

If VCS is not in a running state

To change the agent type definition file when VCS is not in a running state

1. Update the AgentFile attribute.

   ```
   static str AgentFile = "/opt/VRTSvcs/bin/Script51Agent"
   ```

2. Update the IMF attribute.

   The valid values for the Mode key of the IMF attribute are 0 (disabled) and 2 (online monitoring).

   ```
   static int IMF{} = { Mode=num, MonitorFreq=num,
   RegisterRetryLimit=num }
   ```

   For example, to update the IMF attribute such that the Mode key is set to 2, the MonitorFreq key is set to 5, and the RegisterRetryLimit key is set to 3:

   ```
   static int IMF{} = { Mode=2, MonitorFreq=5, RegisterRetryLimit=3
   }
   ```

3. Update the IMFRegList attribute.

   ```
   static str IMFRegList[] = { WAS_NODE, WAS_HOME, ServerName }
   ```

Disabling intelligent resource monitoring

To disable intelligent resource monitoring

1. Make the VCS configuration writable.

   ```
   # haconf -makerw
   ```

2. To disable intelligent resource monitoring for all the resources of a certain type, run the following command:

   ```
   # hatype -modify WebSphere IMF -update Mode 0
   ```

3. To disable intelligent resource monitoring for a specific resource, run the following command:

   ```
   # hares -override resource_name IMF
   ```

   ```
   # hares -modify resource_name IMF -update Mode 0
   ```

4. Save the VCS configuration.

   ```
   # haconf -dump -makero
   ```

Chapter 5

Configuring the service groups for WebSphere Application Server using the CLI

This chapter includes the following topics:

Before configuring the service groups for WebSphere Application Server

Before you configure the WebSphere Application Server service group, you must:

- Verify that the Cluster Server components are installed and configured on all nodes in the cluster where you will configure the service group.
  For more information on installing the components, refer to the *[InfoScale Availability] Installation Guide*.
- Verify that the Cluster Server agent for WebSphere Application Server is installed on all nodes in the cluster.

Configuring service groups for WebSphere Application Server

While various methods and procedures can be used to install and cluster a WebSphere Application Server, Veritas recommends the following general process:

Allocating shared disk resource for the WebSphere node

A WebSphere node is a logical group of WebSphere Application Servers that are located on the same physical machine. This machine is also called a host. Multiple WebSphere nodes can exist on a single node.

Veritas recommends installing each WebSphere node to be clustered on a separate, dedicated shared disk resource (e.g. LUN). Work with the appropriate administrative group in your organization to obtain a shared disk resource for the WebSphere node.

Creating a Veritas disk group, volume, and file system

Create a Veritas disk group, volume, and file system on the shared disk resource allocated for the WebSphere node.

Although not recommended, WebSphere Application Servers can be clustered without using Veritas Volume Manager or Veritas File System. But the tight integration between the cluster, Volume Manager, and File System ensures a more comprehensive and resilient high availability solution for your WebSphere Application Server.

Obtaining dedicated virtual IP addresses and host names

Obtain dedicated virtual IP addresses and host names required to support the WebSphere node IP network configuration.

Several configurations are possible. For example, a Node agent, which is an administrative process that manages all servers running on a WebSphere node, can share one IP address and host name with all of its managed servers. Alternatively, the Node agent and each of its managed servers could be assigned its own IP address and host name.

No matter which configuration you deploy, these network addresses and host names will be used exclusively by this WebSphere node, regardless of which system in the cluster is running it.

Obtaining a dedicated user account if needed

If the WebSphere Application Server will not run using the root account, obtain a dedicated UNIX account for the WebSphere Application Server. Refer to the description of attribute User for important instructions and requirements to create the account.

Creating service group and supporting resources

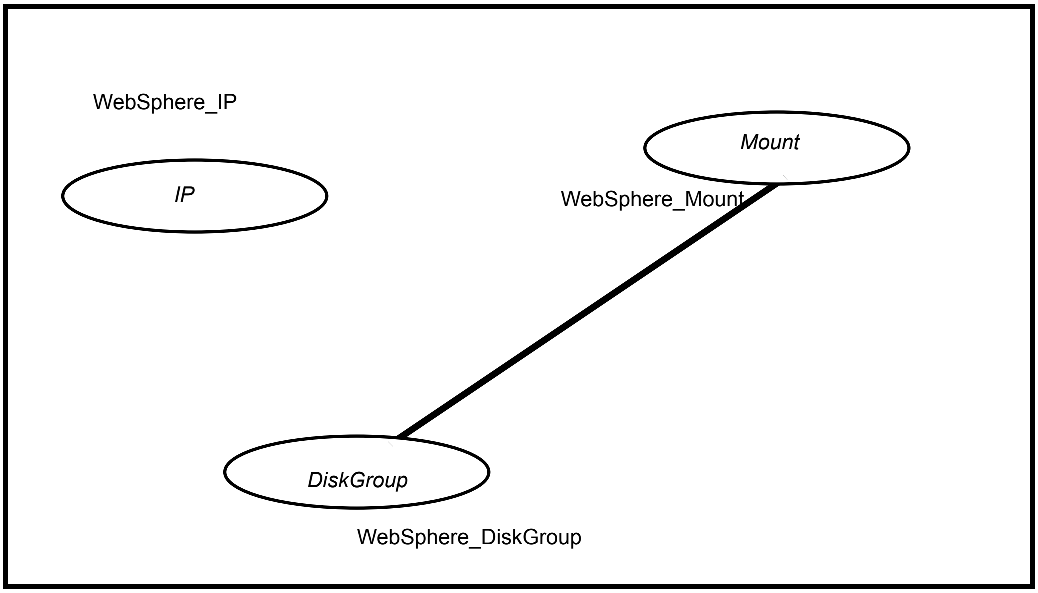
First create a Service Group on a cluster to contain the resources supporting the WebSphere node.

Then create the appropriate cluster resources and links to place the previously created shared disk and networking objects under cluster control.

Test the service group configuration by placing it online. Your service group should appear similar to the following figure.

Figure 5-1 shows a typical service group.

Figure 5-1 Typical service group



Installing the WebSphere software

With the disk and network resources now available and online in the cluster, you are ready to install the WebSphere software.

Follow the instructions in the WebSphere product documentation and install the WebSphere Application Server software. Be sure to instruct the installation program to install the software on the shared disk file system previously established for this WebSphere node.

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

Directory	Purpose
/WAS	Top level directory under which all WebSphere nodes are installed.
/WAS/cell1	Subdirectory under which all WebSphere nodes assigned to cell1 are installed.
/WAS/cell1/depmgr	Subdirectory is the mount point for the shared disk resource dedicated to the Deployment Manager instance supporting cell1.
/WAS/cell1/node1	Subdirectory is the mount point for the shared disk resource dedicated to the WebSphere node named *node1*, which belongs to cell1. The WebSphere software supporting this node agent and its managed Application Servers is installed in this directory.
/WAS/cell1/node2	Subdirectory is the mount point for the shared disk resource dedicated to the WebSphere node named node2, which belongs to cell1. The WebSphere software supporting this node agent and its managed Application Servers is installed in this directory.
/WAS/cell1/node3	Subdirectory is the mount point for the shared disk resource dedicated to the WebSphere node named node3, which belongs to cell1. The WebSphere software supporting this node agent and its managed Application Servers is installed in this directory.

Continue with the same naming pattern for all remaining cells and WebSphere Application Servers.

During the installation, be sure to set the node's Host Name to the dedicated virtual IP host name previously allocated to this node.

Finally, be sure to configure the server's port numbers to avoid conflicts with the port numbers of other WebSphere Application Servers that may be running simultaneously on the same system. Configuring the port numbers is especially important in a cluster environment where WebSphere nodes can be easily moved around the systems in the cluster in almost any combination.

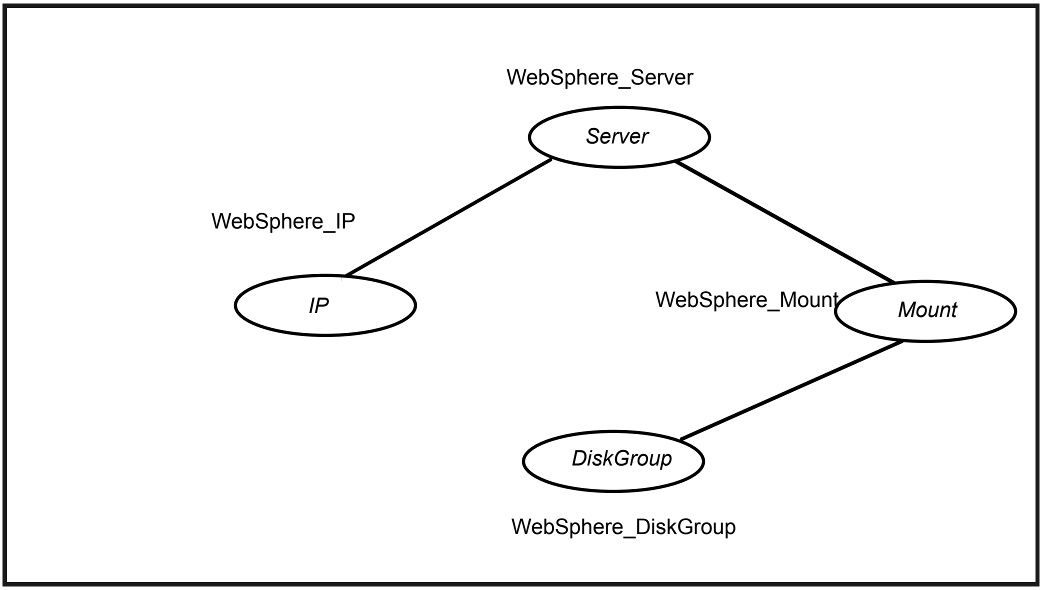
Placing the WebSphere Application Server under cluster control

After the WebSphere Application Server installation is complete, create a cluster resource using the agent for WebSphere to place the server under cluster control.

Your service group should now appear similar to the following figure.

Figure 5-2 shows a typical service group.

Figure 5-2 Typical service group



Warning: After a WebSphere Application Server is placed under cluster control, do not attempt to start or stop the instance without using a cluster interface. Only use the Web Console, Java Console, or command-line interface to start or stop a managed WebSphere instance.

Creating service groups for WebSphere Application Server under Solaris non-global zones

To configure zones on each cluster node

1. Set up the non-global zone configuration.

   `hazonesetup` *`servicegroup_name zoneres_name zone_name password systems`*

   For example:

   `hazonesetup -g` *`servicegroup_name`* `-r` *`zoneres_name`* `-z` *`zone_name`* `-p` *`password`* `-s` *`systems`*

2. Verify the non-global zone configuration.

   `hazoneverify` *`servicegroup_name`*

3. Whenever you make a change that affects the zone configuration, run the `hazonesetup` command to reconfigure the zones in VCS.
4. Make sure that the zone configuration files are consistent on all nodes at all times. The file is located at `/etc/zones/zone_name.xml`.
5. Make sure that the application is identical on all nodes. If you update the application configuration on one node, apply the same updates to all nodes.
6. Configure the service groups for WebSphere Application Server.

Chapter 6

Troubleshooting the agent for WebSphere Application Server

This chapter includes the following topics:

Using the correct software and operating system versions

Ensure that you use correct software and operating system versions.

For information on the software versions that the agent for WebSphere Application Server supports, see the Veritas Services and Operations Readiness Tools (SORT) site: https://sort.veritas.com/agents.

Meeting prerequisites

Before installing the agent for WebSphere Application Server, ensure that the following prerequisites are met.

For example, you must install the ACC library on VCS before installing the agent for WebSphere Application Server.

See "Before you install the Cluster Server agent for WebSphere Application Server" on page 13.

Configuring WebSphere Application Server resources

Before using WebSphere Application Server resources, ensure that you configure the resources properly. For a list of attributes used to configure all WebSphere Application Server resources, refer to the agent attributes.

Starting the WebSphere Application Server instance 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 WebSphere Application Server instance independent of the cluster framework. Refer to the cluster documentation for information about disabling a resource.

You can then restart the WebSphere Application Server instance outside the cluster framework.

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

A sample procedure to start a WebSphere instance outside the cluster framework, is described as follows.

To start a WebSphere Deployment Manager outside the cluster framework

1 Using the user name specified in the User attribute, log into the host on which the WebSphere Deployment Manager application is to run.

2 Use the values specified in the agent attributes to start the WebSphere Deployment Manager.

For example, assume that the WebSphere Deployment Manager environment is set as follows:

Attribute	Value
ServerType	DeploymentManager
ServerName	dmgr
WAS_NODE	was60c1dmsol
WAS_HOME	/ibm/was/v60/cell1/depmgr
ServerProfile	Dmgr01Specify this attribute for WebSphere version 6.0 and later

3 Go to specified directory.

5.x	/ibm/was/v60/cell1/depmgr/bin
6.x	/ibm/was/v60/cell1/depmgr/profiles/Dmgr0/bin

4 Using the startManager.sh script, start the Deployment Manager.

5.x	`/ibm/was/v60/cell1/depmgr/bin/startManager.sh`
6.x	`/ibm/was/v60/cell1/depmgr/profiles/Dmgr01/bin/startManager.sh`

5 Ensure that the Deployment Manager Server starts successfully.

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

To start a WebSphere node agent outside the cluster framework

1. Using the user name specified in the User attribute, log into the host on which the WebSphere Node agent application is to run.
2. Use the values specified in the agent attributes to start the WebSphere Node agent.

   For example, assume that the WebSphere Node agent environment is set as follows:

Attribute	Value
ServerType	NodeAgent
ServerName	nodeagent
WAS_NODE	was60c1n1sol
WAS_HOME	/ibm/was/v60/cell1/node1
ServerProfile	Default

3. Go to specified directory.

WebSphere version	Directory
5.x	/ibm/was/v60/cell1/node1/bin
6.x	/ibm/was/v60/cell1/node1/profiles/default/bin

4. Using the startNode.sh script, start the Node Agent:

5.x	`/ibm/was/v60/cell1/node1/bin/startNode.sh`
6.x	`/ibm/was/v60/cell1/node1/profiles/default/bin/startNode.sh`

5. Ensure that the Node Agent starts successfully.

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

Reviewing error log files

If you face problems while using WebSphere Application Server or the agent for WebSphere Application Server, use the log files described in this section to investigate the problems.

Reviewing cluster log files

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

The VCS engine log file is at `/var/VRTSvcs/log/engine_A.log`.

Reviewing agent log files

In case of problems while using the agent for WebSphere, 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/.`*`Resource_Name`*`.`*`Entry_Point`*`.`*`Process_ID`*

For example, for a resource WAS50DeployMgr_dmgr:

`/tmp/.WAS50DeployMgr_dmgr.online.`*`Process_ID`*

`/tmp/.WAS50DeployMgr_dmgr.offline.`*`Process_ID`*

`/tmp/.WAS50DeployMgr_dmgr.clean.`*`Process_ID`*

`/tmp/.WAS50DeployMgr_dmgr.monitor.`*`Process_ID`*

If a resource, WAS50DeployMgr_dmgr is unable to bring a WebSphere Node Manager online, you can access the /tmp/.WAS50DeployMgr_dmgr.online.*Process_ID* for more information so that you can diagnose the problem.

Note: These files are overwritten each time you execute the corresponding agent function process. In case you want to save the information, make a copy of the files at another location.

Using trace level logging

The ResLogLevel attribute controls the level of logging that is written in a cluster log file for each WebSphere Application Server 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 are produced. Veritas recommends that you localize the ResLogLevel attribute for a particular resource.

The LogDbg attribute should be used to enable the debug logs for the ACCLib-based agents when the ACCLIB version is 6.2.0.0 or later and the VCS version is 6.2 or later.

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. Note the time before you begin to operate the identified resource.
5. Test the identified resource. The function reproduces the problem that you are attempting to diagnose.
6. Note the time when the problem is reproduced.
7. Set the ResLogLevel attribute back to INFO for the identified resource:

   ```
   # hares -modify Resource_Name ResLogLevel INFO -sys SysA
   ```

8. Save the configuration changes.

   ```
   # haconf -dump
   ```

9. Review the contents of the log file.

   Use the time noted in the previous steps to diagnose the problem.

   You can also contact Veritas support for more help.

To enable debug logs for all resources of type WebSphere

Enable the debug log.

```
# hatype -modify WebSphere LogDbg DBG_5
```

To override the LogDbg attribute at resource level

Override the LogDbg attribute at the resource level and enable the debug logs for the specific resource.

```
# hares -override WebSphere LogDbg
# hares -modify WebSphere LogDbg DBG_5
```

Defining additional environment variables for a WebSphere Application Server instance

The WebSphere Application Server uses the `setupCmdLine.sh` file, which defines the required environment needed for WebSphere Application Server. The file is sourced in all the start, stop, and monitoring scripts used by the WebSphere Application Server agent.

```
bash-3.00# cat
/opt/IBM/WebSphere/AppServer/profiles/AppSrv01/bin/startServer.sh

#!/bin/sh

binDir=`dirname ${0}`

. ${binDir}/setupCmdLine.sh

${WAS_HOME}/bin/startServer.sh "$@"
```

By design, the WebSphere Application Server agent does not source the user's proflile. If you need to export any additional environment variables, create an environment file with the required variables and source it in following scripts:

`startManager.sh, startNode.sh, startServer.sh, stopManager.sh, stopNode.sh, stopServer.sh, serverStatus.sh`

For example, append a line in `startServer.sh` in the following manner:

```
# Call User Environement here.

. <Path to env file>/setUserEnv.sh
```

Troubleshooting the configuration for IMF

If you face problems with the IMF configuration or functionality, consider the following:

- Ensure that the following attributes are configured with appropriate values.
  - AgentFile

  - IMF
  - IMFRegList
    If IMFRegList is not configured correctly, the WebSphere Application Server resources that have been registered for IMF get unregistered every time the monitor function is run.

- If you have configured the required attributes to enable the WebSphere Application Server agent for IMF, but the agent is still not IMF-enabled, restart the agent. The imf_init function runs only when the agent starts up, so when you restart the agent, imf_init runs and initializes the WebSphere Application Server agent to interface with the AMF kernel driver.
- You can run the following command to check the value of the MonitorMethod attribute and to verify that a resource is registered for IMF.

  `# hares -value` *`resource`* `MonitorMethod` *`system`*

  The MonitorMethod attribute specifies the monitoring method that the agent uses to monitor the resource:

  - Traditional—Poll-based resource monitoring
  - IMF—Intelligent resource monitoring

- You can use the `amfstat` to see a list of registered PIDs for a WebSphere resource.
  Following is a sample output of the `ps -ef` command.

```
# ps -ef| egrep 'server1|nodeagent'

root     30429     1  0 Nov29 ?        00:00:29 /WAS/AppServer/
java/bin/java -Declipse.security -Dwas.status.socket=21973 -
Dosgi.install.area=/WAS/AppServer -Dosgi.configuration.area=/
WAS/AppServer/profiles/AppSrv01/configuration -Dosgi.framework.
extensions=com.ibm.cds,com.ibm.ws.eclipse.adaptors -Xshareclasses
:name=webspherev70_%g,groupAccess,nonFatal -Xscmx50M -Xboot
classpath/p:/WAS/AppServer/java/jre/lib/ext/ibmorb.jar:/WAS/
AppServer/java/jre/lib/ext/ibmext.jar -classpath /WAS/AppServer/
profiles/AppSrv01/properties:/WAS/AppServer/properties:/WAS/
AppServer/lib/startup.jar:/WAS/AppServer/lib/bootstrap.jar:/WAS/
AppServer/lib/jsf-nls.jar:/WAS/AppServer/lib/lmproxy.jar:/WAS/
AppServer/lib/urlprotocols.jar:/WAS/AppServer/deploytool/itp/
batchboot.jar:/WAS/AppServer/deploytool/itp/batch2.jar:/WAS/
AppServer/java/lib/tools.jar -Dibm.websphere.internalClass
AccessMode=allow -Xms50m -Xmx256m -Xcompressedrefs -Dws.ext.dirs
=/WAS/AppServer/java/lib:/WAS/AppServer/profiles/AppSrv01/classes
:/WAS/AppServer/classes:/WAS/AppServer/lib:/WAS/AppServer/
```

```
installedChannels:/WAS/AppServer/lib/ext:/WAS/AppServer/web/
help:/WAS/AppServer/deploytool/itp/plugins/com.ibm.etools.ejbdeploy
/runtime -Dderby.system.home=/WAS/AppServer/derby -Dcom.ibm.itp.
location=/WAS/AppServer/bin -Djava.util.logging.configureByServer=
true -Duser.install.root=/WAS/AppServer/profiles/AppSrv01 -Djavax.
management.builder.initial=com.ibm.ws.management.PlatformMBean
ServerBuilder -Dwas.install.root=/WAS/AppServer -Dpython.cachedir=
/WAS/AppServer/profiles/AppSrv01/temp/cachedir -Djava.util.logging.
manager=com.ibm.ws.bootstrap.WsLogManager -Dserver.root=/WAS/App
Server/profiles/AppSrv01 -Djava.awt.headless=true -Djava.security.
auth.login.config=/WAS/AppServer/profiles/AppSrv01/properties/ws
jaas.conf -Djava.security.policy=/WAS/AppServer/profiles/AppSrv01
/properties/server.policy com.ibm.wsspi.bootstrap.WSPreLauncher -
nosplash -application com.ibm.ws.bootstrap.WSLauncher com.ibm.ws.
runtime.WsServer /WAS/AppServer/profiles/AppSrv01/config Dmgr01
Cell AppSrv01Node nodeagent

root     31996 30429  0 Nov29 ?        00:00:41 /WAS/AppServer/java/
bin/java -Declipse.security -Dwas.status.socket=59734 -Dosgi.install.
area=/WAS/AppServer -Dosgi.configuration.area=/WAS/AppServer/profiles
/AppSrv01/configuration -Djava.awt.headless=true -Dosgi.framework.
extensions=com.ibm.cds,com.ibm.ws.eclipse.adaptors -Xshareclasses:
name=webspherev70_%g,groupAccess,nonFatal -Xscmx50M -Xbootclasspath
/p:/WAS/AppServer/java/jre/lib/ext/ibmorb.jar:/WAS/AppServer/java
/jre/lib/ext/ibmext.jar -classpath /WAS/AppServer/profiles/AppSrv01/
properties:/WAS/AppServer/properties:/WAS/AppServer/lib/startup.jar:/
WAS/AppServer/lib/bootstrap.jar:/WAS/AppServer/lib/jsf-nls.jar:/WAS/
AppServer/lib/lmproxy.jar:/WAS/AppServer/lib/urlprotocols.jar:/WAS/
AppServer/deploytool/itp/batchboot.jar:/WAS/AppServer/deploytool/itp
/batch2.jar:/WAS/AppServer/java/lib/tools.jar -Dibm.websphere.
internalClassAccessMode=allow -Xms50m -Xmx256m -Xcompressedrefs
-Dws.ext.dirs=/WAS/AppServer/java/lib:/WAS/AppServer/profiles/
AppSrv01/classes:/WAS/AppServer/classes:/WAS/AppServer/lib:/WAS/
AppServer/installedChannels:/WAS/AppServer/lib/ext:/WAS/AppServer/
web/help:/WAS/AppServer/deploytool/itp/plugins/com.ibm.etools.
ejbdeploy/runtime -Dderby.system.home=/WAS/AppServer/derby -Dcom.
ibm.itp.location=/WAS/AppServer/bin -Djava.util.logging.configure
ByServer=true -Duser.install.root=/WAS/AppServer/profiles/AppSrv01
-Djavax.management.builder.initial=com.ibm.ws.management.Platform
MBeanServerBuilder -Dwas.install.root=/WAS/AppServer -Dpython.
cachedir=/WAS/AppServer/profiles/AppSrv01/temp/cachedir -Djava.
util.logging.manager=com.ibm.ws.bootstrap.WsLogManager -Dserver.
root=/WAS/AppServer/profiles/AppSrv01 -Dcom.ibm.security.jgss.
```

```
debug=off -Dcom.ibm.security.krb5.Krb5Debug=off -Djava.security.
auth.login.config=/WAS/AppServer/profiles/AppSrv01/properties/
wsjaas.conf -Djava.security.policy=/WAS/AppServer/profiles/
AppSrv01/properties/server.policy com.ibm.wsspi.bootstrap.WS
PreLauncher -nosplash -application com.ibm.ws.bootstrap.WSLauncher
com.ibm.ws.runtime.WsServer /WAS/AppServer/profiles/AppSrv01/config
Dmgr01Cell AppSrv01Node server1root     32215 23059  0 18:07 pts/1
00:00:00 grep WAS
```

The `amfstat` command shows the Queue Manager PIDs monitored by the WebSphere Application Server agent.

```
# amfstat
AMF Status Report

Registered Reapers (2):
=======================
 RID    PID    EVENT    REAPER
 11   28232    2    0    WebSphere

Process ONLINE Monitors (2):
============================
 RID    R_RID    PID    GROUP
 12    11    31996    was_server
 13    11    30429    was_nodeagent
```

- Run the following command to set the ResLogLevel attribute to TRACE. When you set ResLogLevel to TRACE, the agent logs messages in the WebSphere_A.log file.

  `# hares -modify ResourceName ResLogLevel TRACE`

  For more information about the ResLogLevel attribute, See Table 3-1 on page 33.
- Run the following command to view the content of the AMF in-memory trace buffer.

  `# amfconfig -p dbglog`

Known issues

This release of the agent for WebSphere Application Server has the following known issues:

Problem

An error message might appear when you run the `hares -offline` command to take a resource offline.

Description

When a resource is taken offline, it is unregistered from the AMF module. However, the imf_register function attempts to unregister the resource again.

Workaround

It is safe to ignore this error message.

Sample Configurations

This appendix includes the following topics:

- About sample configurations for the agents for WebSphere Application Server
- Sample agent type definition for WebSphere Application Server
- Sample configuration in a VCS environment

About sample configurations for the agents for WebSphere Application Server

The sample configuration graphically depicts the resource types, resources, and resource dependencies within the service group. Review these dependencies carefully before configuring the agents for WebSphere Application Server. For more information about these resource types, refer to the *Cluster Server Bundled Agents Reference Guide*.

Sample agent type definition for WebSphere Application Server

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 WebSphereTypes.cf file in the `/etc/VRTSvcs/conf/config` cluster configuration directory.

An excerpt from this file follows.

```
type WebSphere (
   static str ArgList [] = { ResLogLevel, State, IState, ServerName,
      WAS_NODE, WAS_HOME ,User, ServerProfile, ServerType,
      StartOptions, StopOptions, MonitorProgram, SecondLevelMonitor}
   str  ResLogLevel = INFO
   str  ServerName
   str  WAS_NODE
   str  WAS_HOME
   str  User
   str  ServerProfile
   str  ServerType
   str  StartOptions
   str  StopOptions
   str  MonitorProgram
   int  SecondLevelMonitor
   )
```

VCS 5.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 WebSphereTypes.cf file in the `/etc/VRTSagents/ha/conf/config` cluster configuration directory.

An excerpt from this file follows.

```
type WebSphere (
   static boolean AEPTimeout = 1
   static str AgentFile = "/opt/VRTSvcs/bin/Script50Agent"
   static str AgentDirectory = "/opt/VRTSagents/ha/bin/WebSphere"
   static str ArgList [] = { ResLogLevel, State, IState, ServerName,
                             WAS_NODE, WAS_HOME ,User, ServerProfile,
                             ServerType, StartOptions, StopOptions,
                             MonitorProgram, SecondLevelMonitor,
                             AdminUserName, AdminPassword }
   str  ResLogLevel = "INFO"
   str  ServerName
   str  WAS_NODE
   str  WAS_HOME
   str  User
   str  ServerProfile
   str  ServerType
   str  StartOptions
   str  StopOptions
   str  MonitorProgram
   int  SecondLevelMonitor = 0
   str  AdminUserName
   str  AdminPassword
)
```

Sample configuration in a VCS environment

The following is an excerpt from a VCS configuration file (main.cf) that defines a Network Deployment of WebSphere Application Servers and two independent Application Servers (Application Servers that are not part of a Network Deployment).

This configuration demonstrates that you can combine Network Deployment WebSphere Cells with independent WebSphere Application Servers. In the example, there is one WebSphere Cell consisting of one Deployment Manager named `dmgr`. The WebSphere Cell contains two Node Managers, both named `nodeagent`.

Review the information to configure a service group that manages one independent, stand-alone Application Server.

See “Service group configuration options” on page 41.

```
WebSphere WAS_DELOYPMGR (
  Critical = 0
  ResLogLevel = TRACE
  ServerName = dmgr
  WAS_NODE = system1Node01
  WAS_HOME = "/oraem/WAS/AppServer"
  User = root
  ServerProfile = "/oraem/WAS/AppServer/profiles/Dmgr01_Prof"
  ServerType = DeploymentManager
  AdminUserName = adminUser
  AdminPassword = abcdefg
  )

WebSphere WAS_NODEAGENT (
  Critical = 0
  ResLogLevel = TRACE
  ServerName = nodeagent
  WAS_NODE = system1Node02
  WAS_HOME = "/oraem/WAS/AppServer"
  User = root
  ServerProfile = "/oraem/WAS/AppServer/profiles/AppSrv02"
  ServerType = NodeAgent
  )
```

```
WebSphere WAS_RES (
  Critical = 0
  ResLogLevel = TRACE
  ServerName = server1
  WAS_NODE = system1Node01
  WAS_HOME = "/oraem/WAS/AppServer"
  User = root
  ServerProfile = "/oraem/WAS/AppServer/profiles/AppSrv01"
  ServerType = ApplicationServer
  )
```

The following is a sample excerpt of the `main.cf` file for VCS 6.2.

```
WebSphere was_res (
  ResLogLevel = TRACE
  ServerName = server1
  WAS_NODE @system01 = System01Node01
  WAS_NODE @system02 = System02Node01
  WAS_HOME = "/opt/IBM/WebSphere/AppServer"
  User = root
  ServerProfile = AppSrv01
  ServerType = ApplicationServer
  LogDbg = { DBG_5 }
  )
```

The following is a sample resource configuration for WebSphere Application Server Cluster for VCS 6.2.

```
include "CFSTypes.cf"
include "CVMTypes.cf"
include "WebSphereTypes.cf"

cluster wascfsclus (
 UserNames = { admin = cmnFmhMjnInnLvnHmk }
 ClusterAddress = "10.11.12.13"
 Administrators = { admin }
 HacliUserLevel = COMMANDROOT
 )

system SystemA (
 )

system SystemB (
 )
```

```
group app1nodegrp (
 SystemList = { SystemA = 0 }
 AutoStartList = { SystemA }
 )

WebSphere app1noderes (
 ResLogLevel = TRACE
 ServerName = nodeagent
 WAS_NODE = appsrv1-node1
 WAS_HOME = "/was_configuration/appsrv1"
 User = root
 ServerProfile = appsrv1
 ServerType = NodeAgent
 )

requires group cvmmount1 online local firm

group app2nodegrp (
 SystemList = { SystemB = 0 }
 AutoStartList = { SystemB }
 )

WebSphere app2noderes (
 ResLogLevel = TRACE
 ServerName = nodeagent
 WAS_NODE = appsrv2-node1
 WAS_HOME = "/was_configuration/appsrv2"
 User = root
 ServerProfile = appsrv2
 ServerType = NodeAgent
 )

requires group cvmmount2 online local firm

group cvm (
 SystemList = { SystemA = 0, SystemB = 1 }
 AutoFailOver = 0
 Parallel = 1
 AutoStartList = { SystemA, SystemB }
 )

CFSfsckd vxfsckd (
 )

CVMCluster cvm_clus (
 CVMClustName = wascluster
 CVMNodeId = { SystemA = 0, SystemB = 1 }
 CVMTransport = gab
 CVMTimeout = 200
 )

CVMVxconfigd cvm_vxconfigd (
 Critical = 0
 CVMVxconfigdArgs = { syslog }
 )

cvm_clus requires cvm_vxconfigd
vxfsckd requires cvm_clus

group cvmmount (
 SystemList = { SystemA = 0, SystemB = 1 }
 Parallel = 1
 AutoStartList = { SystemA, SystemB }
 )

CFSMount cvmmount (
 MountPoint = "/was_configuration"
 BlockDevice = "/dev/vx/dsk/wasdg/wasvol"
 )

CVMVolDg cvmdg (
 CVMDiskGroup = wasdg
 CVMVolume = { wasvol }
 CVMActivation = sw
 )

requires group cvm online local firm
cvmmount requires cvmdg

group wasclusgrp (
 SystemList = { SystemA = 0, SystemB = 1 }
 Parallel = 1
 AutoStartList = { SystemA, SystemB }
 )
```

```
WebSphere wasclusres (
 ResLogLevel = TRACE
 ServerName = wascluster
 WAS_NODE = appsrv1-node1
 WAS_HOME = "/was_configuration/appsrv1"
 User = root
 ServerProfile = "/was_configuration/appsrv1/profiles/dmgr"
 ServerType = Cluster
 DMHostName = "SystemB.veritas.com"
 DMPort = 8881
 LevelTwoMonitorFreq = 1
 MonitorTimeout = 120
 )

requires group wasdmgr online global soft
requires group app1nodegrp online global firm
requires group app2nodegrp online global firm

group wasdmgr (
 SystemList = { SystemA = 0, SystemB = 1 }
 AutoStartList = { SystemB }
 )

IP dmgr_ip (
  Device = eth0
  Address = "10.209.78.78"
  NetMask = "255.255.252.0"
  )

 NIC dmgr_nic (
    Enabled = 0
    Device = eth0
    )

 WebSphere wasdmgrres (
  ResLogLevel = TRACE
  ServerName = dmgr
  WAS_NODE = dmgr-node1
  WAS_HOME = "/was_configuration/appsrv1"
  User = root
  ServerProfile = "/was_configuration/appsrv1/profiles/dmgr"
  ServerType = DeploymentManager
  )
```

```
requires group cvmmount online global soft
dmgr requires dmgr_ip
dmgr_ip requires dmgr_nic
```