Running webMethods Broker in a High-Availability UNIX Cluster

Version 9.6

April 2014

WEBMETHODS

Document ID: BR-HA-CL-96-20161018

Table of Contents

About this Guide

Installing webMethods Broker in a high-availability cluster requires the efforts of three people: the user's system administrator, the cluster vendor's installation consultant, and a webMethods administrator. This guide is written primarily for the webMethods administrator.

For simplicity, this guide assumes users are installing components in the default locations. For example, it assumes the root directory for webMethods Broker is `/opt/softwareag/`.

Important: If you have a lower fix level installed, some of the features described in this document might not be available to you. For a cumulative list of fixes and features, see the latest fix readme on the Empower website at https://empower.softwareag.com.

Document Conventions

Convention	Description
Bold	Identifies elements on a screen.
Narrowfont	Identifies storage locations for services on webMethods Integration Server, using the convention *folder.subfolder:service* .
UPPERCASE	Identifies keyboard keys. Keys you must press simultaneously are joined with a plus sign (+).
Italic	Identifies variables for which you must supply values specific to your own situation or environment. Identifies new terms the first time they occur in the text.
`Monospace font`	Identifies text you must type or messages displayed by the system.
{ }	Indicates a set of choices from which you must choose one. Type only the information inside the curly braces. Do not type the { } symbols.

Convention	Description
|	Separates two mutually exclusive choices in a syntax line. Type one of these choices. Do not type the | symbol.
[ ]	Indicates one or more options. Type only the information inside the square brackets. Do not type the [ ] symbols.
...	Indicates that you can type multiple options of the same type. Type only the information. Do not type the ellipsis (...).

Online Information

Software AG Documentation Website

You can find documentation on the Software AG Documentation website at http://documentation.softwareag.com. The site requires Empower credentials. If you do not have Empower credentials, you must use the TECHcommunity website.

Software AG Empower Product Support Website

You can find product information on the Software AG Empower Product Support website at https://empower.softwareag.com.

To submit feature/enhancement requests, get information about product availability, and download products, go to Products.

To get information about fixes and to read early warnings, technical papers, and knowledge base articles, go to the Knowledge Center.

Software AG TECHcommunity

You can find documentation and other technical information on the Software AG TECHcommunity website at http://techcommunity.softwareag.com. You can:

- Access product documentation, if you have TECHcommunity credentials. If you do not, you will need to register and specify "Documentation" as an area of interest.
- Access articles, code samples, demos, and tutorials.
- Use the online discussion forums, moderated by Software AG professionals, to ask questions, discuss best practices, and learn how other customers are using Software AG technology.
- Link to external websites that discuss open standards and web technology.

1 Overview

Overview of High-Availability Clustering

High-availability (HA) clustering is a solution that uses clustering software and special purpose hardware to minimize system downtime. HA clusters are groups of computing resources that are implemented to provide high availability of software and hardware computing services. HA clusters operate by having redundant groups of resources (such as CPU, disk storage, network connections, and software applications) that provide service when the primary system resources fail. webMethods Broker can run in an HA cluster environment, under Windows or UNIX.

In a clustered environment, groups of resources (such as CPU, disk storage, network connections, and software applications) are connected to shared storage hardware, and controlling cluster software. When the primary system fails, the cluster software switches control to a secondary group of resources.

Without an HA cluster, a failed resource will remain unavailable until that resource is brought back online. Based on the dependencies among resources, a failed resource can make the entire computing environment unusable. HA clustering remedies this problem by detecting hardware or software failures and immediately starting (or failing over to) the redundant resources on another node without requiring administrative intervention. As part of this failover process, clustering software will start the resources on the redundant node in a predefined order (or resource dependency) to ensure that the entire node will come up properly.

Virtual IP Addresses

For client applications to access services in an HA cluster in a transparent way, a virtual IP address must be supplied to the client applications. This virtual IP address is usually referred to as the "logical host." This logical host identity is a network address (or host name) and is not tied to a single cluster node.

When failover happens, the cluster control software will resolve the virtual IP address to the physical IP address of the active node in the cluster. (A virtual IP address is like any other IP address except it does not have a specific host or node to resolve to. It resolves at run time to a node wherever the IP is physically bound and reachable on the network.) The client application should not be affected in any way other than experiencing a brief outage of the services.

Types of Cluster Configuration

There are three basic HA cluster configurations:

- **Active/Passive.** A two-node configuration with one node being active and one node in a standby state at any given time. This is the most common configuration, and the one that is recommended for use with Broker applications.

- **Active/Active.** A two-node configuration with both nodes being active at the same time. Each node typically runs different sets (or instances) of services. When one node fails, the services on the failed node will failover to the active node.
- **N-to-1.** A multi-node configuration with one dedicated spare node. After the original failed node is recovered, services are restored from the spare node to the original node and the spare node returns to its standby state.

In addition to the above basic configurations, there are advanced HA configurations that are beyond the scope of this manual. For more information, see your cluster software product documentation for your cluster software.

Running Applications in a High-Availability Cluster

Most applications can run in an HA cluster environment provided that they have:

- Defined start, stop, and monitor procedures
- The ability to store the application's state information and data on a shared disk
- The ability to survive a crash and restart themselves in a known state
- The ability to meet license requirements and host name dependencies

The webMethods Broker application fully meets all of these requirements.

The defined start, stop, and monitor procedures are usually provided as Unix shell scripts that must be incorporated into the cluster control software's infrastructure. Some custom coding will be required to enable the cluster control software to invoke these scripts to control the application.

The cluster control software will determine the health of the resources by periodically probing them using monitor scripts. When the cluster control software determines one of the resources in the cluster has failed, it will shut down the remaining active resources on that cluster node and then start the resources on the spare node.

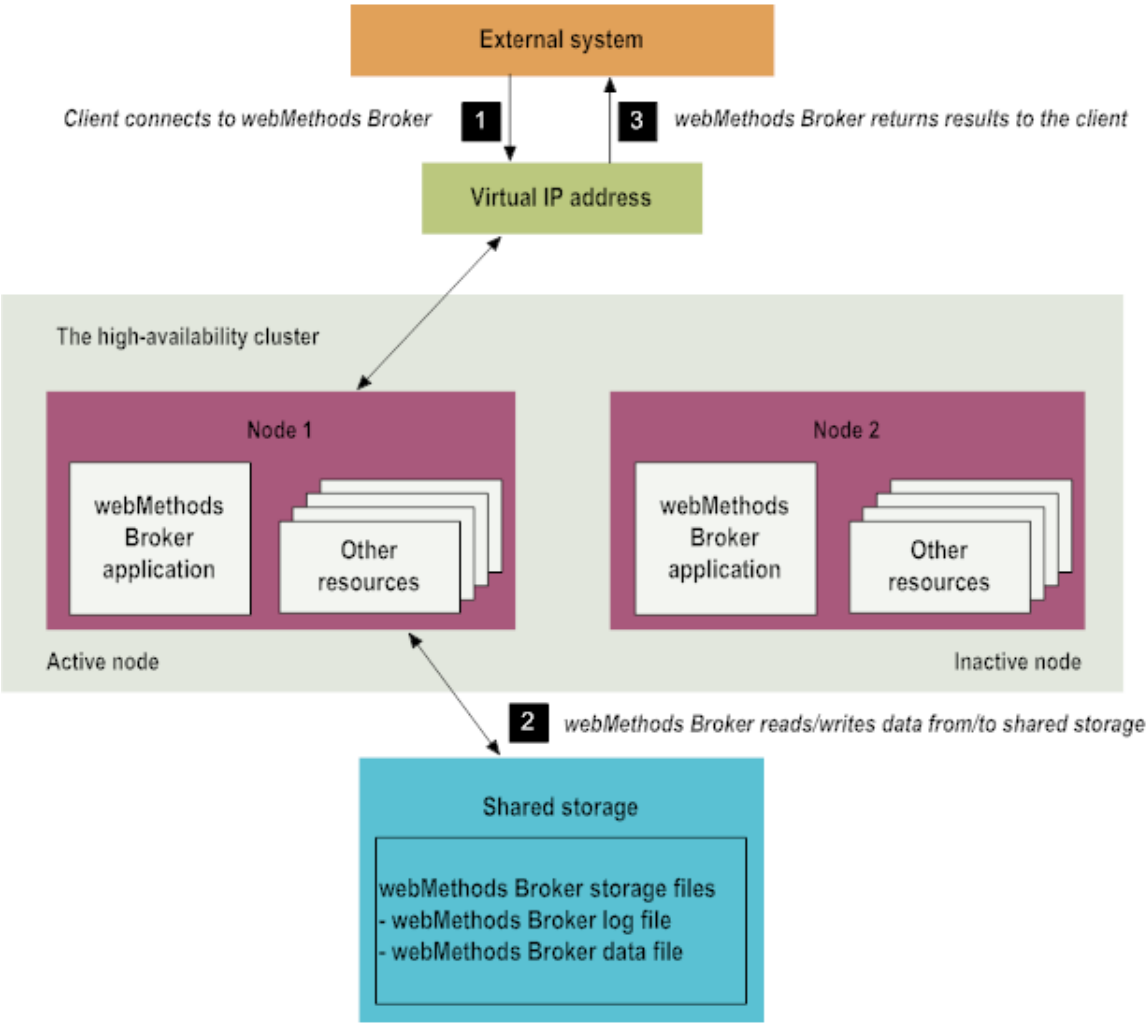
webMethods Broker provides three Korn shell scripts plus one configuration file for starting, stopping, and monitoring a Broker Server. The scripts and the configuration file are available in the *webMethods Broker_directory*/scripts/generic_cluster directory.

Running the webMethods Broker Application in a High-Availability Cluster

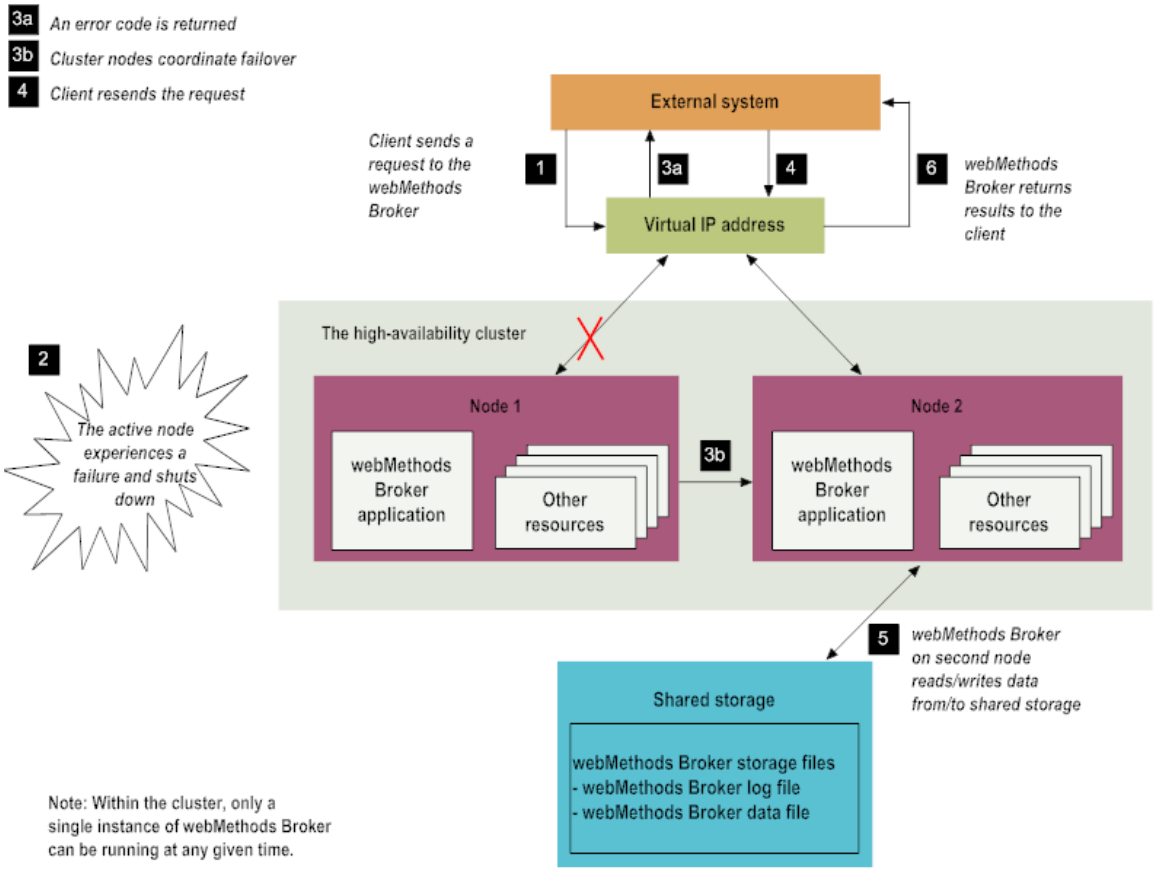
webMethods Broker runs as a service in a cluster. Within a cluster, there can be only a single instance of a Broker Server running at any given time. The spare Broker Server is stopped.

When a client makes a request to a Broker, the Broker handles the request much the same as in an unclustered environment. Although, in a clustered environment the Broker writes the client information to a shared disk instead of a private data store.

The diagram below illustrates the flow of documents through a typical clustered environment.



If a Broker fails, subsequent requests for the session are redirected to a spare Broker in the cluster that is currently active and running, as shown in the diagram below.



Setting Up webMethods Broker to Run in a High-Availability Cluster Environment

There are three task categories when setting up webMethods Broker to run in an HA cluster environment:

- System and network administration tasks performed by the user's system administrators.
- Cluster hardware and software (for example, Veritas, HP ServiceGuard, IBM HACMP or Oracle Solaris Cluster) installation performed by cluster installation consultants.
- webMethods Broker and HA script installation performed by a webMethods administrator.

The table below summarizes the steps for configuring webMethods Broker to run in an HA cluster environment. The table columns, *User's SysAd*, *Cluster Vendor* and *webMethods Administrator*, indicate that the responsible party for the tasks are the user's system administrator, cluster vendor's installation consultant and the webMethods administrator, respectively.

Important: You must perform steps 7, 8, and 9 for each node.

Step No.	Task	Comments	User's SysAd	Cluster Vendor	webMethods Administrator
1	Review this book	Read this book to gain a better understanding of the installation and configuration process.			X
2	Install HA cluster environment			X	
3	Configure HA cluster environment including the shared disk storage			X	
4	Administer the HA cluster environment so it is ready for software installation		X		
5	Configure the external network connection to the HA cluster and create the virtual host (virtual IP address) for the HA cluster		X		
6	Test the basic HA installation to ensure it		X	X	

Step No.	Task	Comments	User's SysAd	Cluster Vendor	webMethods Administrator
	functions properly				
7a	Install and configure webMethods Broker on the cluster node	See "Install and Configure the webMethods Broker Software" on page 16 for instructions.		X	X
7b	Test that webMethods Broker runs on the cluster node	See "Verify That webMethods Broker Is Running" on page 17 for instructions.			X
8	Update the webMethods Broker Monitor configuration file	See "Configure webMethods Broker Monitor" on page 17 for instructions.			X
9a	Configure the webMethods Broker HA scripts.	See "Configure the High-Availability Cluster Scripts" on page 18 for instructions.			X
9b	Update the HA script configuration file with appropriate system parameters from the HA cluster.	See "Edit the wment-defs.sh File" on page 19 for instructions.			X
9c	Test the HA scripts to ensure	See "Verify the webMethods Scripts Work Correctly" on			X

Step No.	Task	Comments	User's SysAd	Cluster Vendor	webMethods Administrator
	they function properly.	page 21 for instructions.			
10	Incorporate each node's HA scripts into the cluster control software.			X	X
11	Test the entire HA installation with the webMethods Broker application running to ensure it functions (fails over) properly.	You can verify this two ways: ■ A manual failover using the cluster's vendor-specific commands ■ An automatic failover by provoking/ simulating a webMethods Broker crash	X	X	X

2 Installation and Configuration

Introduction

As part of the overall process to configure webMethods Broker to run in a high-availability (HA) cluster environment, webMethods Broker must be installed on all the nodes to which the cluster software can failover the webMethods Broker. For a summary of this process, see "Setting Up webMethods Broker to Run in a High-Availability Cluster Environment" on page 11.

This chapter provides instructions for the steps the webMethods administrator must take for each node in the cluster. Finish one node before beginning work on the next.

Install and Configure the webMethods Broker Software

The instructions in this section document the procedure to install and configure webMethods Broker on a cluster node. When installing webMethods Broker on a cluster node, you must work with the cluster vendor's installation consultant.

Important: The webMethods Broker installation must be identical on all nodes. For example, the port numbers that Broker Server and Broker Monitor listen on must be the same on all nodes. Also, all instances of Broker Server must point to the same webMethods Broker storage files on the shared storage.

Installing on the First Node

When installing webMethods Broker on the first cluster node, work with the cluster vendor's installation consultant to prepare the node to respond to the virtual IP address and have access to the storage files on the shared storage.

To install webMethods Broker on the first node, follow the instructions in *Installing Software AG Products*. Be sure to create a new webMethods Broker Server configuration (that is, the storage files) and place it on the shared storage.

Installing on the Second Node

When installing webMethods Broker on the second cluster node, work with the cluster vendor's installation consultant.

To install webMethods Broker on the second node, do the following:

1. Make the first node inactive.
2. Install webMethods Broker, using the instructions in *Installing Software AG Products*, without creating a new Broker Server configuration.

3. Make the second node active so that it will respond to the virtual IP address and have access to the storage files on the shared storage.
4. Use the `server-config` utility with the `add` subcommand to point the webMethods Broker executable to the storage files you already created for the first node on the shared storage. For information about the `server-config` utility, see *Administering webMethods Broker*.

Verify That webMethods Broker Is Running

Use the Broker user interface on My webMethods to verify that webMethods Broker is properly installed and working.

To verify that webMethods Broker is running, do the following:

1. Use the following command to check the status of the running Broker Server:

```
broker_status
```

2. Use the following command to verify that the Broker Server can respond to documents:

```
broker_ping
```

3. For complete information of these commands, see the *Administering webMethods Broker* guide.

Configure webMethods Broker Monitor

You must configure webMethods Broker Monitor to *not* automatically restart Broker.

To configure webMethods Broker Monitor, do the following:

1. Open the awbrokermon.cfg configuration file in an editor.
2. Add this line to the file:

```
monitor-start-servers=no
```

3. Save the configuration file.
4. Restart webMethods Broker Monitor so the change to the configuration file takes effect.

Configure Broker Server

You must configure Broker Server to *not* automatically restart.

To configure Broker Server, do the following:

1. Stop the Broker Server.

2. Open the awbroker.cfg configuration file in an editor.
3. Add this line to the file:

```
auto-restart=0
```

4. Save the configuration file.
5. Restart Broker Server so the change to the configuration file takes effect.

Configure the webMethods Broker High-Availability Cluster Scripts

By default,webMethods Broker HA cluster scripts are installed on the *webMethods Broker_directory*/scripts/generic_cluster directory. This file contains the following scripts:

- **wment-defs.sh** This file contains configuration parameters representing the webMethods Broker setup; the startBroker.sh, stopBroker.sh, and brokerStatus.sh scripts refer to this file. The parameters of this file are described in "Edit the wment-defs.sh File" on page 19.
- **startBroker.sh** This script starts a Broker Server. Broker Server details are specified in the wment-defs.sh parameter file.
- **stopBroker.sh** This script stops a Broker Server. Broker Server details are specified in the wment-defs.sh parameter file.
- **brokerStatus.sh** This script checks the health of a Broker Server. Broker Server details are specified in the wment-defs.sh parameter file.

  You can configure this script to run in two different modes, based on the requirements of the clustering software. You can either run the script once until Broker is pinged successfully, or run the script indefinitely until Broker is not available.

You may rename these scripts. For example, in a Veritas cluster you should rename them to online, offline, and monitor, respectively.

Important: Always use the startBroker.sh and stopBroker.sh scripts to start and stop the Broker Server. If you use the S45broker65 script you may receive unexpected results.

Configure the High-Availability Cluster Scripts

To configure the webMethods Broker HA cluster scripts, do the following:

1. Go to the *webMethods Broker_directory* /scripts/generic_cluster directory.
2. Copy the file sample-wment-defs.sh to wment-defs.sh.

Edit the wment-defs.sh File

Review and optionally edit the parameters in the wment-defs.sh file to reflect the node's setup. This file's parameters are described in the table below.

Parameter	Description
ADDITIONAL_PATH	Any subdirectory containing system utilities and commands that the startBroker.sh, stopBroker.sh, and brokerStatus.sh scripts may need. You may add to the list of subdirectories that come pre-populated in the wment-defs.sh file.
LOG_FILE	Log file in which the startBroker.sh, stopBroker.sh, and brokerStatus.sh scripts will write log entries. The default value is ./wmbkr_cluster.log.
WMENT_LOGICAL_HOST	Logical host name of the cluster. This is the name that the startBroker.sh, stopBroker.sh, and brokerStatus.sh scripts will use while communicating with the webMethods Broker subsystem on the node.
WMENT_AWBROKER_SCRIPT	Absolute path to the directory where the startBroker.sh and stopBroker.sh scripts can find the Broker startup script, aw_broker82.
WMENT_HOME	Absolute path to where webMethods Broker is installed (for example, /opt/softwareag/Broker).
WMENT_CONFIG	Absolute path to the webMethods Broker monitor directory where the awbrokermon.cfg file resides (for example, /opt/softwareag/Broker/bin).
BROKERMON_PORTNO	The port number (for example, 6850) that the startBroker.sh script uses to communicate with the Broker Monitor.
BROKER_PORTNO	The port number (for example, 6849) that the stopBroker.sh and brokerStatus.sh scripts use to communicate with the Broker Server.

Parameter	Description
BROKER_NAME	webMethods Broker name that can be used in the probe operation. Leave this parameter empty if the default webMethods Broker is the target.
PROBE_INDEFINITELY	Indicates whether the probe should return after a successful probe, or it should return only after the probe fails. The default value is YES. If the cluster software has a probe retry capability, set this parameter to NO.
PROBE_MAX_RETRY	Number of attempts the brokerStatus.sh script makes before declaring Broker Server could not be started. The default value is 5. If the cluster software has a probe retry capability, disable the webMethods Broker retry capability by setting PROBE_MAX_RETRY to 0.
PROBE_INTERVAL_SECS	Number of seconds the brokerStatus.sh script sleeps between probes. The default value is 60. If the cluster software has a probe retry capability, disable the webMethods Broker retry capability by setting PROBE_INTERVAL_SECS to 0.
START_PROBE_MAX_RETRY	Number of attempts the startBroker.sh script makes before declaring that Broker Server could not be started. The default value is 5.
START_PROBE_INTERVAL_SECS	Number of seconds the startBroker.sh script sleeps between probes to check the start status of the Broker Server. The default value is 60.
SSL_CERTFILE	Absolute path to where the SSL certificates file resides.
SSL_PASSWORD	Password required to access the SSL certificate file.
SSL_TRUSTFILE	Absolute path to where the trust store file resides.
SSL_NOENCRYPT	Indicates if the probe communication with webMethods Broker is to be encrypted.

Parameter	Description
	■ If you want the communication to be encrypted, set the parameter to a null value. ■ If you want the communication to be unencrypted, set the parameter to a non-null value.

Verify the webMethods Scripts Work Correctly

Verify that the webMethods scripts work correctly by manually executing them on the node.

The node you are testing must be active and have the dependent resources available (for example, the IP address is bound and the shared disk is mounted).

Run the scripts in the order shown below. If webMethods Broker is currently running on the node, stop it using the stopBroker.sh script before starting the verification.

startBroker.sh

Run this script on the active cluster node:

```
./startBroker.sh
```

Then use the command `broker_ping` on the virtual IP address; the ping should return the status of webMethods Broker. If the ping returns an error, check the log file in the scripts directory for more information.

brokerStatus.sh

Run this script on the active cluster node where webMethods Broker is currently running:

```
./brokerStatus.sh
```

Check the log file in the scripts directory for a probe success or failure message.

Note: Depending on how the `PROBE_INDEFINITELY` parameter is set, the brokerStatus.sh script may or may not return after a successful probe.

stopBroker.sh

Run this script on the cluster node where webMethods Broker is currently running:

./stopBroker.sh

Then use the command `broker_ping` on the virtual IP address; the ping should return a "Broker not running" error. If the ping indicates success or returns a different error, check the log file in the scripts directory for more information.

Debugging the Scripts

To debug the scripts when they fail to work correctly, do the following:

Add the following command in the beginning of each script:

set -x

This command will cause the script to display commands as they appear on the screen, which you can later analyze to identify the problem.