Note: Before using this information and the product it supports, read the information in “Notices” on page v.

First Edition (November 2008)

This edition applies to WebSphere Virtual Enterprise V6.1.

This document created or updated on November 12, 2008.
Contents

Chapter 1. Virtual enterprise runtime topologies ................................................. 1
  1.1 Topology overview ................................................................. 2
  1.2 Topology components .............................................................. 3
    1.2.1 On demand routers ......................................................... 3
    1.2.2 IBM HTTP Server or other supported Web server ..................... 4
    1.2.3 Cells ....................................................................... 4
    1.2.4 Dynamic clusters .......................................................... 5
    1.2.5 Core groups ................................................................. 5
  1.3 Simple test topology ..................................................................... 6
    1.3.1 Topology overview ............................................................ 7
    1.3.2 Configuring a simple test topology ....................................... 8
  1.4 Standalone topology ..................................................................... 10
    1.4.1 Topology overview ............................................................ 11
    1.4.2 Configuring a standalone topology ....................................... 12
  1.5 Large single cell topology ......................................................... 14
    1.5.1 Topology overview ............................................................ 14
    1.5.2 Configuring a large single cell topology ............................... 15
  1.6 Isolated cells topology .............................................................. 28
    1.6.1 Topology overview ............................................................ 28
    1.6.2 Configuring an isolated cells topology ............................... 29
  1.7 Silo cell topology ....................................................................... 29
    1.7.1 Topology overview ............................................................ 30
    1.7.2 Configuring a Silo cell topology ......................................... 33
  1.8 Unlinked cell topology .............................................................. 37
    1.8.1 Middleware types .............................................................. 37
    1.8.2 Topology overview ............................................................ 38
    1.8.3 Configuring an unlinked cell topology ................................ 39
  1.9 Virtualization .............................................................................. 47

Chapter 2. Service policies ............................................................................ 49
  2.1 How are service policies used? ....................................................... 50
    2.1.1 Work classes ................................................................... 51
    2.1.2 Transaction classes ............................................................ 52
  2.2 Leading practices ........................................................................... 52
    2.2.1 Approach to defining service policies .................................... 52
    2.2.2 Defining service policy performance goals ............................. 53
    2.2.3 Defining work classes ....................................................... 53
    2.2.4 Defining transaction classes .............................................. 54
    2.2.5 Defining maximum CPU usage when using vertical stacking ... 54
  2.3 Creating service policies .............................................................. 55
  2.4 Associating service policies with an application ............................. 58
2.5 References ................................................................. 61

Chapter 3. Health management ............................................. 63
3.1 Using the health management subsystem ............................. 64
3.2 Health policies ............................................................ 64
   3.2.1 Defining a health policy ......................................... 64
   3.2.2 Using Custom health policies ................................. 65
   3.2.3 Leading practices ............................................... 65
   3.2.4 Reaction mode .................................................. 66
   3.2.5 Actions ............................................................ 67
3.3 Monitoring health management tasks ................................ 68
3.4 Health controller ......................................................... 69
   3.4.1 Custom actions .................................................. 70
3.5 ITCAM and Virtual Enterprise ....................................... 71

Chapter 4. Application hosting and chargeback ......................... 75
4.1 Collecting chargeback metrics ........................................ 76
4.2 Leading practices ....................................................... 76

Chapter 5. Product maintenance ........................................... 77
5.1 Approach to maintenance ............................................... 78
5.2 Installing maintenance ................................................ 78
   5.2.1 Finding and downloading fixes and fix packs ............... 79
   5.2.2 Installing the Update Installer ................................ 82
   5.2.3 Applying the maintenance using the Update Installer .... 83
5.3 Using the Centralized Installation Manager ........................ 83
   5.3.1 Installing products ............................................. 84
   5.3.2 Installing a middleware agent ................................ 85
   5.3.3 Installing refresh packs, fix packs, maintenance tools, or Interim fixes .............. 85
5.4 Using maintenance mode ............................................... 90
   5.4.1 Putting a node into maintenance mode ....................... 91
   5.4.2 Future reference ............................................... 91
5.5 Verifying the product maintenance level ............................ 91
   5.5.1 Update Installer wizard ....................................... 92
   5.5.2 Centralized Installation Manager wizard .................... 92
   5.5.3 The updatelog.txt file ......................................... 93
   5.5.4 versionInfo command ......................................... 95
   5.5.5 Administrative console ........................................ 96
   5.5.6 The SystemOut.log file ...................................... 97

Related publications ....................................................... 99
IBM Redbooks ............................................................... 99
Online resources ........................................................... 99
How to get Redbooks ..................................................... 100
Help from IBM ............................................................. 100
Notices

This information was developed for products and services offered in the U.S.A.

IBM may not offer the products, services, or features discussed in this document in other countries. Consult your local IBM representative for information on the products and services currently available in your area. Any reference to an IBM product, program, or service is not intended to state or imply that only that IBM product, program, or service may be used. Any functionally equivalent product, program, or service that does not infringe any IBM intellectual property right may be used instead. However, it is the user's responsibility to evaluate and verify the operation of any non-IBM product, program, or service.

IBM may have patents or pending patent applications covering subject matter described in this document. The furnishing of this document does not give you any license to these patents. You can send license inquiries, in writing, to:
IBM Director of Licensing, IBM Corporation, North Castle Drive, Armonk, NY 10504-1785 U.S.A.

The following paragraph does not apply to the United Kingdom or any other country where such provisions are inconsistent with local law: INTERNATIONAL BUSINESS MACHINES CORPORATION PROVIDES THIS PUBLICATION "AS IS" WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESS OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF NON-INFRINGEMENT, MERCHANTABILITY OR FITNESS FOR A PARTICULAR PURPOSE. Some states do not allow disclaimer of express or implied warranties in certain transactions, therefore, this statement may not apply to you.

This information could include technical inaccuracies or typographical errors. Changes are periodically made to the information herein; these changes will be incorporated in new editions of the publication. IBM may make improvements and/or changes in the product(s) and/or the program(s) described in this publication at any time without notice.

Any references in this information to non-IBM Web sites are provided for convenience only and do not in any manner serve as an endorsement of those Web sites. The materials at those Web sites are not part of the materials for this IBM product and use of those Web sites is at your own risk.

IBM may use or distribute any of the information you supply in any way it believes appropriate without incurring any obligation to you.

Information concerning non-IBM products was obtained from the suppliers of those products, their published announcements or other publicly available sources. IBM has not tested those products and cannot confirm the accuracy of performance, compatibility or any other claims related to non-IBM products. Questions on the capabilities of non-IBM products should be addressed to the suppliers of those products.

This information contains examples of data and reports used in daily business operations. To illustrate them as completely as possible, the examples include the names of individuals, companies, brands, and products. All of these names are fictitious and any similarity to the names and addresses used by an actual business enterprise is entirely coincidental.

COPYRIGHT LICENSE:

This information contains sample application programs in source language, which illustrate programming techniques on various operating platforms. You may copy, modify, and distribute these sample programs in any form without payment to IBM, for the purposes of developing, using, marketing or distributing application programs conforming to the application programming interface for the operating platform for which the sample programs are written. These examples have not been thoroughly tested under all conditions. IBM, therefore, cannot guarantee or imply reliability, serviceability, or function of these programs.
Trademarks

IBM, the IBM logo, and ibm.com are trademarks or registered trademarks of International Business Machines Corporation in the United States, other countries, or both. These and other IBM trademarked terms are marked on their first occurrence in this information with the appropriate symbol (© or ™), indicating US registered or common law trademarks owned by IBM at the time this information was published. Such trademarks may also be registered or common law trademarks in other countries. A current list of IBM trademarks is available on the Web at http://www.ibm.com/legal/copytrade.shtml

The following terms are trademarks of the International Business Machines Corporation in the United States, other countries, or both:

- developerWorks®
- Domino®
- IBM®
- Lotus®
- Redbooks®
- Redbooks (logo)®
- Tivoli®
- WebSphere®
- z/OS®

The following terms are trademarks of other companies:

VMware, the VMware "boxes" logo and design are registered trademarks or trademarks of VMware, Inc. in the United States and/or other jurisdictions.

J2EE, Java, JDBC, JVM, Sun, Sun Java, and all Java-based trademarks are trademarks of Sun Microsystems, Inc. in the United States, other countries, or both.

Windows, and the Windows logo are trademarks of Microsoft Corporation in the United States, other countries, or both.

UNIX is a registered trademark of The Open Group in the United States and other countries.

Other company, product, or service names may be trademarks or service marks of others.
Preface

WebSphere® Virtual Enterprise extends WebSphere Application Server Network Deployment, providing an enhanced quality of service in dynamic operations and extended manageability. WebSphere Virtual Enterprise provides application server virtualization, resource management, and a host of advanced operational facilities, such as performance visualization, health monitoring, and application editions. This combination of capabilities is sometimes referred to collectively as dynamic operations.

This IBM® Redpaper publication discusses leading practices for WebSphere Virtual Enterprise Version 6.1. It provides complementary documentation to the WebSphere Virtual Enterprise Information Center. The paper starts by discussing some of the more common topologies for setting up a Virtual Enterprise environment. It provides information about advantages, disadvantages, and guidance on how to implement the topology. It also touches on high focus areas in WebSphere Virtual Enterprise, including service policies, health management, application hosting and chargeback, and product maintenance.

Prior to V6.1.0.3, WebSphere Virtual Enterprise was known as WebSphere Extended Deployment Operations Optimization. WebSphere Virtual Enterprise is now available as part of WebSphere Extended Deployment, or as a separate package.

The team that wrote this paper

This paper was produced by a team of specialists from around the world working at the International Technical Support Organization, Raleigh Center.

Akiko Mitake joined IBM in 2003. She has been a member of the WebSphere Level1 Support team for six years and is the WebSphere Extended Deployment technical lead on this team in Japan. Her other areas of expertise include WebSphere Application Server and WebSphere Process Server.

Fabio Santos Bento da Silva has been with IBM Brazil for two years. He is currently a WebSphere IT specialist with IBM Global Account. He has 12 years of experience in Java™ and he is an IBM Certified WebSphere and Portal Server Administrator.

Thanks to the following people for their contributions to this project:

Carla Sadtler
International Technical Support Organization, Raleigh Center

Derrick Foley
IBM Raleigh

Adrian P. Vrouwenvelder
IBM Raleigh

Ann Black-Ziegelbein
IBM Raleigh

Carl Braam
IBM Raleigh

Lan Vuong
IBM Raleigh
Become a published author

Join us for a two- to six-week residency program! Help write a book dealing with specific products or solutions, while getting hands-on experience with leading-edge technologies. You will have the opportunity to team with IBM technical professionals, Business Partners, and Clients.

Your efforts will help increase product acceptance and customer satisfaction. As a bonus, you will develop a network of contacts in IBM development labs, and increase your productivity and marketability.

Find out more about the residency program, browse the residency index, and apply online at:
ibm.com/redbooks/residencies.html

Comments welcome

Your comments are important to us!

We want our papers to be as helpful as possible. Send us your comments about this paper or other IBM Redbooks® in one of the following ways:

► Use the online Contact us review Redbooks form found at:
ibm.com/redbooks

► Send your comments in an e-mail to:
redbooks@us.ibm.com

► Mail your comments to:
IBM Corporation, International Technical Support Organization
Dept. HYTD Mail Station P099
2455 South Road
Poughkeepsie, NY 12601-5400
Virtual enterprise runtime topologies

This chapter discusses common topologies for WebSphere Virtual Enterprise. It starts by discussing the topology components and general considerations for the prime components. It then introduces common topologies, both for a runtime environment with a single Network Deployment cell, and for more wide-spread environments with multiple cells.
1.1 Topology overview

This chapter discusses several recommended topologies. The following list is an overview of the topologies discussed:

The recommended topologies for single cell environments are as follows:

- Simple test topology
  Section 1.3, “Simple test topology” on page 6 is appropriate for non-production environments. There is no security, fault tolerance, or scalability of a production topology.

- Standalone topology
  Section 1.4, “Standalone topology” on page 10 is recommended for small production environments. It consists of only one cell and one core group.

- Large single cell topology
  Section 1.5, “Large single cell topology” on page 14 is recommend for situations where you have a large existing Network Deployment environment and want to implement Virtual Enterprise. This topology consists of multiple machines, many application servers, multiple core groups, and requires a core group bridge between the core groups.

The recommended topologies for multiple cell environments are as follows:

- Isolated cells topology
  Section 1.6, “Isolated cells topology” on page 28 is a large topology with simple communication and workload that completes within one cell and one core group. No communication occurs between the cells, or between the core groups.

- Silo cell topology
  Section 1.7, “Silo cell topology” on page 29 usually consists of two identical cells with the same applications and routes requests from IBM HTTP Servers, like multiple remote data centers.

- Unlinked cell topology
  Section 1.8, “Unlinked cell topology” on page 37 is recommended for customers who want to exploit WebSphere Virtual Enterprise features on existing WebSphere Application Server 5.1 or 6.0 systems, or non-WebSphere systems.

For each topology there are advantages and disadvantages that must be considered with your particular needs in mind. For example, it is important that you analyze how many application servers your environment will support and to predict the growth of the environment. You must also be aware of any service level agreements (SLAs) that exist. An SLA is a contract that defines an understanding between a service provider and their customer. It sets expectations for performance, and defines the procedure and reports needed to track compliance. Terms set in an SLA can affect the performance requirements, and by extension, affect the way you plan your topology and define your service levels.
1.2 Topology components

In order to design a topology for your enterprise, it is necessary to understand the components of a topology design and the impact of the various layers on the overall performance and complexity of the environment. This section provides a quick introduction to the components you will find in the recommended topologies:

- On demand routers
- IBM HTTP Server or other supported Web server
- Cells
- Dynamic clusters
- Core groups

1.2.1 On demand routers

The on demand router (ODR) is an intelligent HTTP and Session Initiation Protocol (SIP) proxy server in WebSphere Virtual Enterprise. The ODR is the point of entry into a WebSphere Virtual Enterprise environment and is a gateway through which HTTP requests and SIP messages flow to back-end application servers. It includes the following primary functions:

- Request routing
- Intelligent routing based on sense and response mechanisms from back-end servers
- Classification of incoming requests based on rules defined by the business owner

You can create ODRs using the WebSphere administrative console or using a wsadmin script.

Topology considerations for the ODR

When planning a Virtual Enterprise topology, many of the planning considerations are focused on the ODR. It is important to ensure that the ODRs are scalable and highly available.

The ODR introduces an additional and critical processing layer to the server network topology. Because it is central to the functioning of the Virtual Enterprise environment, the ODR becomes a single point of failure (SPOF) and, by implication, any performance issues with the ODR may affect the entire Virtual Enterprise environment. Therefore, the administrators and architects who plan the topology must take care to ensure the high availability of ODRs. Factors that affect ODR performance include the number of supported clients, message size, secure sockets layer (SSL) implementation, and type of hardware.

The decision as to the number of ODRs to place in an environment depends on the enterprise and infrastructure. It is generally recommended that you have at least two ODRs to provide high availability. We also suggest that an ODR balance workload between the servers within the same cell and core group as the ODR.

Various factors will come into play when determining whether to use additional ODRs. For example, the number of clients served, the number of applications, the types and size of sessions, and security factors are all factors to take into consideration. As the number of clients increase, there is more overhead inherent in keeping track of all clients. We therefore suggest that you have a close estimation of clients accessing the environment, and evaluate the performance levels of the current set of ODRs. The decision to add additional ODRs should be considered if the client base increases over time, or even exponentially for a short time, due to a business activity such as a promotional offer.
1.2.2 IBM HTTP Server or other supported Web server

In the topologies discussed in this paper, a Web server is placed in front of the ODR as a trusted proxy server. You can choose any Web server that is supported by WebSphere Application Server 6.1, like Apache HTTP Server, Internet Information Services, Lotus® Domino® Enterprise Server, or Sun™ Java™ System Web server. The examples in this paper use IBM HTTP Server as the Web server, so where you see “IBM HTTP Server” in the following discussions, replace that with the name of the Web server you are using.

For a list of supported software for WebSphere Application Server V6.1, including Web servers, see the following Web page:

http://www-01.ibm.com/support/docview.wss?rs=180&uid=swg27007642

Web server plug-in

The Web server must have the WebSphere Application Server Plug-in installed. By default, the plug-in configuration file provides properties for establishing communication between the Web server and the application server. In a WebSphere Virtual Enterprise environment, however, the plug-in file must provide the properties for establishing communication between the Web server and the ODR. This is accomplished by having the ODR generate the plug-in configuration file. Each ODR can generate a plug-in configuration file that causes a Web server plug-in to route as follows:

- All ODRs
- All ODRs in the same cell as itself
- All ODRs on the same node as itself
- Only to itself

You can configure ODR instances to dynamically generate the plug-in configuration file. This will ensure the plug-in is re-generated when certain administrative actions, such as installing a new application or configuring a new virtual host, are taken.

1.2.3 Cells

When you create a deployment manager profile, a cell is created. A cell provides a way to group nodes of your Network Deployment environment. You define the nodes that make up a cell, according to the specific criteria that make sense in your organizational environment. The deployment manager provides a central administration point for the nodes in the cell. It communicates administration changes to the servers in the nodes through node agents.

Topologies that incorporate more than one cell isolate hardware and software failures. This isolation is helpful in a variety of situations:

- Rolling out a new application or a revision of an existing application
  You can bring the new application or revision online in one cell, and test it in a live situation while other cells handle client requests with the production version of the application.
- Deploying a new version of the WebSphere Application Server software
  You can bring the new version into production, and test it in a live situation without interrupting service.
- Applying fixes or patches to the WebSphere Application Server software
  You can take each cell offline to upgrade it, without interrupting the application.

In some cases multi-cell topologies allow for more robust failover and high availability than larger single cell topology solutions.
1.2.4 Dynamic clusters

A dynamic cluster is a cluster of application servers that run the same applications. Workload is dynamically balanced across the servers in the cluster through the use of weights and performance information that is collected from the cluster members.

**Leading practice:** It is common to have one dynamic cluster per application. Consider that if applications share a dynamic cluster they share application servers. If health management decides to recycle an application server because one application is failing, all other applications on that server will be impacted.

For more information about dynamic clusters, consult the following resources:
- IBM Redbooks publication *Optimizing Operations with WebSphere Extended Deployment V6.1*, SG24-7422

1.2.5 Core groups

A core group is a high availability domain that consists of a set of processes in the same cell that can directly establish high availability relationships. A cell must contain at least one core group, although multiple core groups are supported. Each core group contains a core group coordinator to manage its high availability relationships, and a set of high availability policies that are used to manage the highly available components within that core group. Every deployment manager, node agent, and application server is a member of a core group. When a process is created it is automatically added to a core group. You can move processes from one core group to another. By default, the cell contains a single core group, called DefaultCoreGroup.

**Core group rules**
- Every process is a member of exactly one core group.
- All members of a cluster must be members of the same core group.
- Each core group must contain at least one node agent or the deployment manager.

**Leading practice:** A single core group is usually sufficient. However, some topologies or special circumstances require multiple core groups. When you create a dynamic cluster of application servers, the recommendation is to have a maximum number of 50 WebSphere Application Server-related processes (application servers, node agents, and ODRs) per core group. Consider this limit when you set the number of instances and stacking number for your dynamic cluster.

This limitation does not apply to dynamic clusters of non-WebSphere Application Server processes, such as assisted life cycle servers. For more information about life cycle support, see Section 1.8, “Unlinked cell topology” on page 37.
Core group scaling considerations
The amount of system resources, such as CPU and memory, that the high availability manager consumes does not increase linearly as the size of a core group increases. When the number of JVMs grows, the amount of resources that a large core group consumes might become significant. For more information about core group scaling considerations, see the IBM Information Center article Core group scaling considerations, available from the following Web page:


Core group bridge service
The core group bridge service uses a set of bridge interfaces to enable members of one core group to communicate with members of another core group. This communication is done using access point groups to connect the core groups. A core group access point defines a set of bridge interfaces that resolve to IP addresses and ports.

Leading practice: Cross-cell core group bridging is not recommended because it is resource intensive. When you need to configure a core group bridge, use intra-cell core group bridging.

You might configure multiple core groups for a cell in the following scenarios:

- One or more firewalls exist within a cell. A core group cannot contain members from multiple firewall protection domains.
- A large number of processes in the cell and the core group protocols, such as the View Synchrony Protocol, consume correspondingly large amounts of resources, such as processor resources.
- Core group protocols, such as the Failure Detection Protocol, need tuning or configuring to use values that work best with smaller numbers of core group members.

1.3 Simple test topology

This topology is appropriate for non-production environments. While it contains the functionality of a Virtual Enterprise environment, it does not have the security, fault tolerance, or scalability capabilities of a production topology. With this topology, you can test the basic workload management behavior for an application.

This topology provides you with workload balancing among servers in a dynamic cluster through the IBM HTTP Server and the ODR. You can maximize utilization using your defined business goals, and maximize the use of your computing resources using autonomic managers including the Autonomic Request Flow Manager (ARFM), Application Placement Controller (APC) and the Dynamic WorkLoad Manager (DWLM).
1.3.1 Topology overview

Figure 1-1 shows the simple test topology.

This topology can be configured by placing the following components on one or more physical machines.

- The IBM HTTP Server is placed as an entry point of this system. The Web server plug-in is configured so that requests destined for an application server are forwarded to the ODR. The plug-in file is generated by the ODR.
- The ODR is placed between the IBM HTTP Server and the application servers. With the help of the autonomic managers, the ODR categorizes and prioritizes the incoming work before routing it to the appropriate application servers in the dynamic clusters.
- Each dynamic cluster consists of multiple application servers on multiple nodes. Workload for an application is spread across the servers. If one server fails, incoming work for the application is simply routed to other servers in the dynamic cluster.

Because there is only one ODR and one IBM HTTP Server, there is no failover capability if either of these should fail. This topology can be extended with additional ODRs and IBM HTTP Servers, as shown in Section 1.4, “Standalone topology” on page 10.
If you choose to run multiple components in this topology on one node, be aware of the following concerns:

- Potential memory usage on one node
  Ensure enough memory is available to start all necessary servers, including deployment manager, node agents, ODRs and application servers.
- Potential for CPU contention between the WebSphere Virtual Enterprise processes
  Ensure that enough CPU resources are available to WebSphere Virtual Enterprise processes.
- Potential port conflicts if you are configuring multiple processes, especially when configuring multiple ODRs on one node
- WebSphere Virtual Enterprise cannot directly manage the utilization of the deployment manager, node agent and ODR processes.

If the utilization of a node increases, WebSphere Virtual Enterprise can only help by throttling traffic to the application servers.

### 1.3.2 Configuring a simple test topology

The following steps illustrate how to set up this topology.

1. Create one deployment manager profile and a custom profile for each node.

   In an existing Network Deployment environment, you can use the Centralized Installation Manager (CIM) to assist in installing WebSphere Virtual Enterprise across the existing nodes. For more information about CIM, see Chapter 5, “Product maintenance” on page 77 and the following resources:
   - IBM Information Center article *Centralized installation manager overview*, available from the following Web page:
     
   - Redbooks publication *Optimizing Operations with WebSphere Extended Deployment V6.1*, SG24-7422

2. Create the ODR. You can create the ODR either from the administrative console or by using a wsadmin script. The procedure is described in the following resources:
   - IBM Information Center article *Creating ODRs*, available from the following Web page:
     
   - Redbooks publication *Optimizing Operations with WebSphere Extended Deployment V6.1*, SG24-7422

3. Create the dynamic clusters from the administrative console. The number of total JVMs need to be less than 50, including the deployment manager, node agents, and ODRs. The following resources go into detail on how to create dynamic clusters:
   - IBM Information Center article *Creating dynamic clusters*, available from the following Web page:
     
   - Redbooks publication *Optimizing Operations with WebSphere Extended Deployment V6.1*, SG24-7422
4. Install the IBM HTTP Server and the plug-in, then define the existing IBM HTTP Server to the WebSphere administrative console. See the IBM Information Center article *Web server definition*, available from the following Web page:


5. Set up the IBM HTTP Server as a trusted proxy to the ODR. By configuring the secure proxy server, you can inform the ODR that the IBM HTTP Server is a trusted secure proxy so that the ODR can receive requests. A trusted security proxy passes information such as the virtual host name or user identity to the ODR in private HTTP headers.

These changes should prevent you from having to modify any virtual host settings to allow IBM HTTP Server to route to the ODRs, even on nonstandard ports. The procedure is described in the IBM Information Center article *Configuring a Web server as a trusted proxy server*, available from the following Web page:


**Note:** In topologies with multiple ODRs, you need to repeat this step for each ODR.

6. Configure the ODR to generate and propagate the plug-in configuration file automatically. The steps required to do this can be found in the following resources:

- IBM Information Center article *Configuring an on demand router to dynamically update the Web server plug-in configuration*, available from the following Web page:


- Redbooks publication *Optimizing Operations with WebSphere Extended Deployment V6.1*, SG24-7422

It is recommended that you set the scope of the ODR plug-in generation to “Cell” unless you need to generate the plug-in file for an IBM HTTP Server associated with particular node, or to particular ODR. To set the scope from the administrative console, select **Servers → On Demand Routers → odr_name → On Demand Router settings**. Select **Cell** for the Generate Plugin Configuration (Figure 1-2).

![Figure 1-2 Setting the plug-in configuration scope](image)

The ODR generates a plug-in configuration file and places it in the WAS_HOME/profiles/profile_name/etc/ directory of its own machine. You can see that an XML file called plugin-cfg.xml is generated when you save the configuration change and restart the ODR.

This file is updated every time any change that effects the routing from IBM HTTP Server to ODR is made (For example, application installation or uninstallation, or when a start/stop of another ODR occurs).

In this topology, we assume that the IBM HTTP Server and the ODR are in the same machine. If these are in different machines, you need to follow Step 7 of Section 1.4.2, “Configuring a standalone topology” on page 12.
7. Change the path for the plug-in configuration file in the IBM HTTP Server in httpd.conf file to point to the directory where the configuration files are placed during generation, as in the following example:

    WAS_HOME/profiles/profile_name/etc/plugin-cfg-cell_name-node_name-odr_name.xml

The httpd.conf file is located under IHS_HOME/conf directory.

8. Deploy an application to the dynamic clusters. From the administrative console, select Applications > → Install new middleware application → Java 2 Platform, Enterprise Edition.

On the Map modules to servers panel, check the modules and select the target dynamic cluster, then click Apply.

9. Define service policies, transaction policies, and work classes with realistic response time for your business goal. The steps to define service policies, transaction policies, and work classes are described in the following resources:

   - IBM Information Center article Defining a service policy, available from the following Web page:
   
   - Redbooks publication Optimizing Operations with WebSphere Extended Deployment V6.1, SG24-7422

10. Implement health policies in order to monitor your application server environment and execute an appropriate action when the predefined health condition is detected as broken. For more information, refer to the following resources:

   - IBM Information Center article Configuring health management, available from the following Web page:

   - Redbooks publication Optimizing Operations with WebSphere Extended Deployment V6.1, SG24-7422

1.4 Standalone topology

In production environments, security and fault tolerance must be addressed. In this topology, there are multiple deployment managers, ODRs, and IBM HTTP Servers to address the issue of fault tolerance. The placement of these components contributes to the security of the system.

The standalone topology consists of one cell and one core group. We recommend this topology for small environment (less than 50 JVMs).
1.4.1 Topology overview

Figure 1-3 on page 11 shows the simple test topology.

In this topology, a load balancer distributes workload between the two IBM HTTP Servers, and each IBM HTTP Server distributes workload to the two ODRs. The deployment manager must also be configured to be highly available in order to eliminate a single point of failure for administrative functions in the cell.

- The load balancer is placed as an entry point into your environment. It can be placed either outside or inside the Protocol Firewall. You can use WebSphere Application Server Network Deployment Edge Component's Load Balancer to realize the load balancing towards the IBM HTTP Servers.

  The load balancer component must be highly available because it is an entry point to the system. For information about WebSphere Application Server Network Deployment Edge Component's Load Balancer, consult the following resources:
  - WebSphere Application Server Edge Component Information Center
    
  - WebSphere Application Server V6 Scalability and Performance Handbook, SG24-6392

- IBM HTTP Servers are placed in the DMZ to receive requests from the load balancer. These requests are then routed to the ODRs with session affinity.

  When a Web server is defined to the deployment manager as part of the cell, complex communication is required between IBM HTTP Servers and management features in the scope. This can create security and port management issues when the communication must go across a firewall. For this reason, it is recommended that you place the IBM HTTP Servers outside the Cell scope.
IBM HTTP Servers use the plug-in configuration file generated by the ODR. You can avoid propagation issues by copying the generated plug-in from the local drive to a drive accessible to both the IBM HTTP Server and the ODR. The HTTP Server configuration file can be modified to point to the drive as the location of the plugin-cfg.xml file.

- ODRs should not be placed (neither supported nor recommended) in a DMZ because the ODR requires complex communication between autonomic managers, application servers, and deployment managers. ODRs distribute workload between target dynamic cluster members using a round-robin scheme by default. The workload is automatically updated and changed based on the workload data calculated by the autonomic managers. The ODR generates the plug-in configuration file every time a configuration change happens that affects the file, such as adding, deleting, modifying applications, or when the number of ODRs changes.

- Deployment managers should be configured to be highly available. Under this configuration, when a primary, active deployment manager is stopped or fails, the standby deployment manager takes over. Each deployment manager should be installed and configured to run on a different physical or logical machine, and should be configured on a shared file system which supports a fast lock recovery. The IBM Storage Area Network File System (SAN FS) is recommended, and the Network File System Version 4 (NFS) is also an option.

**Advantages of this topology**

This topology is configured using one cell and one core group, and is the most simple topology to exploit Virtual Enterprise features in a production environment. Because all workload and administrative communication is completed within one core group, it reduces the management traffic compared to larger systems with multiple cells or multiple core groups.

**Disadvantages of this topology**

The number of JVMs in one core group should be less than 50, including deployment managers, node agents, and ODRs. This limits the growth potential and could make the justification of cost over benefits difficult.

### 1.4.2 Configuring a standalone topology

Here are the steps required to configure this topology.

1. Configure the high availability deployment manager (HA Dmgr) in order to avoid the SPOF of deployment manager features. The HA dmgr process needs to be configured on a shared file system, SAN FS or NFS, which supports fast lock recover. Consult the following resources for more information about configuring a HA dmgr:
   - IBM Information Center article *Configuring a high availability deployment manager environment*, available at the following Web page:
     
   - Redbooks publication *Optimizing Operations with WebSphere Extended Deployment V6.1*, SG24-7422

**Note:** This feature is supported only on distributed platforms and is not available on z/OS®
2. Create multiple ODRs on multiple nodes without a cluster configuration.
   See Step 2 of Section 1.3.2, “Configuring a simple test topology” on page 8.

3. Create the dynamic clusters.
   See Step 3 of Section 1.3.2, “Configuring a simple test topology” on page 8.

4. Install the Web server plug-in on the IBM HTTP Server machines but do not define the
   IBM HTTP Servers to the WebSphere configuration.

5. Define each IBM HTTP Server as a trusted proxy on each ODR.
   See Step 5 of Section 1.3.2, “Configuring a simple test topology” on page 8.

6. Configure the ODRs to generate the plugin-cfg.xml file automatically.
   See Step 6 of Section 1.3.2, “Configuring a simple test topology” on page 8.

   **Leading practice:** In a system with multiple ODRs, it is recommended that you
   generate the plug-in file on each ODR with the scope set to Cell. When you have the
   plug-in file generated on only one ODR, the plug-in file is not updated when that ODR is
   failed or stopped.

   The scope is set in the Generate Plugin Configuration property of the ODR settings. To
   find this setting in the administrative console, navigate to Servers → On Demand
   Routers → ODR_name → On Demand Router settings.

   In Figure 1-3 on page 11, the plug-in file is generated on both ODR1 and ODR2 at the
   cell scope.

7. Create a script to copy the plug-in files to a location that the IBM HTTP Servers can
   access. Because the physical machines of the IBM HTTP Servers and ODRs are different,
   use a shared file system for a location of copied plug-in file. You need to specify the script
   for both ODR1 and ODR2, by selecting Servers → On Demand Routers →
   ODR_name → On Demand Router settings (Figure 1-4). The script is executed every
   time the plug-in file is updated.

   ![Proxy Plugin Configuration Policy](image)

   Figure 1-4 Specifying a script to copy the plug-in files to the Web server

   A sample script is provided in Redbooks publication Optimizing Operations with
   WebSphere Extended Deployment V6.1, SG24-7422

8. Change the path of the plug-in configuration file in the httpd.conf file to point to the plug-in
   file on the shared file system. The httpd.conf file is located under the IHS_HOME/conf
   directory.

9. Deploy an application and create service policies, transaction classes, work classes, and
   health policies appropriately.
   See Steps 8–10 of Section 1.3.2, “Configuring a simple test topology” on page 8.
1.5 Large single cell topology

A large system consists of multiple machines with many application servers. When the number of JVMs in one core group grows, the amount of resources that the core group consumes can become significant. When this becomes the case, you need to distribute the processes within the cell into multiple core groups of appropriate sizes.

This section introduces a topology of a single Virtual Enterprise cell with multiple core groups. This topology requires a core group bridge between the core groups. The management traffic on this bridge can become considerable, especially in the event a problem occurs with one of the Virtual Enterprise members. Because of this, we recommend this topology only when you cannot move your existing system to a topology as described in Section 1.7, “Silo cell topology” on page 29 or Section 1.6, “Isolated cells topology” on page 28.

1.5.1 Topology overview

Figure 1-5 shows the large single cell topology.

The number of JVMs in one core group should be less than 50. When the number of JVMs grows beyond this, you need to create extra core groups.

It is not recommended that you consolidate the ODRs in one core group because the workload from ODR to dynamic cluster members should not completely depend on the core group bridge. When you create extra core groups, you need to create an extra ODR for each core group.
In Figure 1-5, if you create a Core Group 3, create ODR3 which belongs to the Core Group 3.

- The load balancer, IBM HTTP Servers, and deployment managers are located and configured as described in “Standalone topology” on page 10.
- Core groups need to be created manually. Refer to Section 1.2.5, “Core groups” on page 5 for instructions on how to determine the core group scaling, and the considerations for planning core group infrastructure.
- At least one ODR should be created in each core group.
- Two core group bridge interface points (access points) need to be created for each core group in order to ensure high availability. The default access point is set to the nodeagent process, but it is recommended that you create dedicated application server processes as access points instead of using the node agent. This is because Virtual Enterprise features like ARFM, APC, dWLM, and the health controller need to grab the status of all JVMs within the cell through the core group bridge. This is in addition to the HA Manager and the heavy management traffic that occurs on the bridges.

Advantages of this topology
This topology is well aligned to infrastructure growth. When you want to extend the current standalone topology with minimum change, this topology will be easy for you to adopt.

Disadvantages of this topology
You need to consider the additional infrastructure planning and increase in management traffic due to the core group bridge and its services. When a problem occurs on one of the members or on the core group bridge, performance or other unexpected problems may appear.

1.5.2 Configuring a large single cell topology

Here is the steps required to configure this topology.

1. Follow the steps in Section 1.4.2, “Configuring a standalone topology” on page 12 and configure a standalone topology. We assume the standalone topology has already been created when you think of this large single cell topology.

2. Divide the existing processes that belong to the defaultCoreGoup, into multiple core groups. There are two options to separate the JVMs to multiple core groups.
   - Create a new core group and move existing JVMs to the new core group manually.
     See “Create and configure a new core group manually” on page 16.
   - Divide the core group using coregroupsplit.py script (recommended).
     See “Create and configure a new core group with coregroupsplit.py” on page 18.

3. Change the access points of the DefaultAccessPointGroup for each new core group from the node agent to a dedicated application server process.
   See “Changing the access points” on page 24.

4. (optional) Add new dynamic clusters to the new core groups.
   See “Add dynamic clusters to new core groups” on page 26.
Create and configure a new core group manually

In this scenario, you need to create a new core group manually. Then you can divide the JVMs of the existing core group to the new core group. Be sure that you move all members of one dynamic cluster to the same core group.

The following articles in the WebSphere Application Server v6.1 Information Center provide instructions on how to create a new core group, and move the existing JVMs to the new core groups.

- Creating a new core group (high availability domain)
  

- Moving core group members
  

A core group bridge is created automatically when you create a new core group. For example, when you create a new core group named TestCoreGroup, you can see from the administrative console that the core group bridge is created and the two core groups are connected to each other.

You can see each core group and the core groups it is bridged to by selecting **Servers** → **Core groups** → **Core group settings** (Figure 1-6).

![List of core groups and connections](image)
The new core group bridge can be seen by selecting **Servers → Core groups → Core group bridge settings** (Figure 1-7).

![Core group bridge settings](image1)

**Figure 1-7  Core group bridge settings**

If you create an additional core group named TestCoreGroup2, the new core group bridge is created automatically and the three core groups are connected to each other. You can see the new core group settings selecting **Servers → Core groups → Core group settings** (Figure 1-8).

![Core groups and connections](image2)

**Figure 1-8  Core groups and connections**
Select **Servers → Core groups → Core group bridge settings** (Figure 1-9) to view the core group bridge settings.

![Core group bridge settings](image)

**Figure 1-9  New core group bridges**

### Create and configure a new core group with coregroupssplit.py

With Virtual Enterprise 6.1.0.1 and above, you can use the coregroupssplit.py script to separate the JVMs to an adequate number of core groups. You do not need to create new core groups manually because this script tries to satisfy the following best practices for core groups automatically.

- Each core group must have fewer than 40 servers. This is possible only when you have an adequate ratio of nodes to servers in your cell.
- Each core group must be equipped with at least three core group bridges. This is possible only when you have an adequate ratio of nodes to servers in your cell.
- Each cluster must remain fully mapped to a single core group.

The coregroupssplit.py provides the following options:

- `-reconfig` performs a full reconfiguration to rebalance the distribution of servers among the core groups.
- `-linked` creates a ring topology of core group bridges.
- `-createbridges` creates separate core group bridge processes instead of creating the bridge in the node agent.
- `-numcoregroups` specifies the number of core groups to create.
- `-datastacksize` specifies a number of megabytes that overrides the default data stack size.
- `-proxycoregroup` places the ODRs and proxy servers in a separate core group.
- `-odrcoregroup` places the ODRs and proxy servers in a separate core group.
Chapter 1. Virtual enterprise runtime topologies

See the Information Center article Core group split script, for more information about the coregroupssplit.py script. The article is available at the following Web page:


Figure 1-10 on page 20 shows the output when you execute coregroupssplit.py script. In this example, 57 JVMs with four clusters in ten nodes are split into two core groups. The result is the existing DefaultCoreGroup and the newly created BridgedCoreGroup1 core group.

Notes:

- Each core group must contain at least one deployment manager or node agent process. It is recommended that each core group contains at least one ODR in order to ensure workload flows from the ODR to application servers even if the core group bridge fails.

- You cannot create new dynamic clusters as a member of a newly created core group from the administrative console. When you need to increase the number of JVMs, create the JVMs from administrative console as a member of the defaultCoreGroup and split the core group members again using coregroupssplit.py script, or move them manually.
You can rerun the script any time the number of JVMs grows and you need to separate the core group again.

Figure 1-11 on page 21 shows 105 JVMs with seven clusters in ten nodes are split into three core groups: the existing DefaultCoreGroup, BridgedCoreGroup1 and newly created BridgedCoreGroup2 core group.
C:\IBM\WebSphere\AppServer\bin>wsadmin -lang jython -f coregroupsplit.py
WASX7209I: Connected to process "dmgr" on node WNDVE1CellManager01 using SOAP
co
nector; The type of process is: DeploymentManager
### Automatic coregroup configuration
### Version: 01/28/08
###nodes: 10
wasNodes: 10
nonWasNodes: 0
servers: 105
clusters: 7
desired number of coregroups: 3
retrieving existing coregroups
coregroup: DefaultCoreGroup
cgap CGAP_1(DefaultCoreGroup)
cgap BridgedCoreGroup1 CGAP(BridgedCoreGroup1)
### Final CoreGroup configuration
###
coregroup BridgedCoreGroup1 servers: 35 preferred coordinator(s): WNDVE1/nodeagent WNDVE3/nodeagent WNDVE6/nodeagent
  NODE_AGENT:WNDVE1
  NODE_AGENT:WNDVE3
  NODE_AGENT:WNDVE6
  TestDC
  TestDC4
  WNDVECluster01
coregroup BridgedCoreGroup2 servers: 35 preferred coordinator(s): WNDVE1Node01/nodeagent WNDVE4/nodeagent WNDVE7/nodeagent
  NODE_AGENT:WNDVE1Node01
  NODE_AGENT:WNDVE4
  NODE_AGENT:WNDVE7
  TestDC2
  TestDC5
coregroup DefaultCoreGroup servers: 34 preferred coordinator(s): WNDVE2/nodeagent WNDVE5/nodeagent WNDVE8/nodeagent
  NODE_AGENT:WNDVE2
  NODE_AGENT:WNDVE5
  NODE_AGENT:WNDVE8
  TestDC3
  TestWsadminDC
  WS_DEPLOYMENT_MANAGER
  WS_PROXY_SERVERS
  WS_UNCLUSTERED_SERVERS
saving workspace
finished.
C:\IBM\WebSphere\AppServer\bin>

Figure 1-11  Output from the coregroupsplit.py script
From the administrative console, you can see the new core groups and core group bridges.

To view the core group settings, go to **Servers → Core groups → Core group settings** (Figure 1-12).

![Core groups settings](image)

**Figure 1-12** New core groups and bridges

To view the core group bridge settings, go to **Servers → Core groups → Core group bridge settings** (Figure 1-13).

![Core group bridge settings](image)

**Figure 1-13** New core group bridge settings
The core group split best practices are enabled only when you have an adequate ratio of nodes to servers in your cell. If this condition is not detected, the core group members are not divided, even if the number of JVMs in the DefaultCoreGroup is more than 40. See Figure 1-14 on page 23 for an example of output from the script when the split does not occur.

```bash
C:\IBM\WebSphere\AppServer\bin>wsadmin -lang jython -f coregroupssplit.py
WASX7209I: Connected to process "dmgr" on node WNDVE1CellManager01 using SOAP
co
nector; The type of process is: DeploymentManager

#######################################################
## Automatic coregroup configuration
## Version: 01/28/08
#######################################################
nodes: 4
wasNodes: 4
nonWasNodes: 0
servers: 63
clusters: 6
desired number of coregroups: 1
retrieving existing coregroups
coregroup: DefaultCoreGroup
cgap CGAP_1(DefaultCoreGroup)
unable to find cluster for server: WNDVE1/webserver1

###############################################################
# Final CoreGroup configuration
###############################################################
coregroup DefaultCoreGroup servers: 62 preferred coordinator(s): WNDVE1/nodeagent WNDVE1Node01/nodeagent WNDVE2/nodeagent NODE_AGENT:WNDVE1 NODE_AGENT:WNDVE1Node01 NODE_AGENT:WNDVE2 TestDC TestDC2 TestDC3 TestDC4 TestDC5 WNDVECluster01 WS_DEPLOYMENT_MANAGER WS_PROXY_SERVERS
saving workspace
finished.
C:\IBM\WebSphere\AppServer\bin>
```

*Figure 1-14  Output from the coregroupssplit.py script when the split does not occur*
Changing the access points
When you create new core groups, you should change the access point of DefaultAccessPointGroup from the nodeagent (default) to dedicated application server processes.

1. Go to Servers → Core groups → Core group bridge settings and click DefaultAccessPointGroup (Figure 1-15).

2. Click Core group access points, then select CGAP_1\DefaultCoreGroup and click Show Detail (Figure 1-16).

Leading practice:
We recommend that you use the coregroupssplit.py script for the following reasons:

- The script tries to satisfy the best practices for core group split.
- If you feel the result is unbalanced, or you need to rebalance the core groups again for some reason, you can rerun the script with the -reconfig option to rebalance the distribution of servers among the core groups.
- When the number of JVMs is changed and you want to rebalance, just rerun the coregroupssplit.py script again.
3. Select **Bridge interfaces** and delete the listed node agent and/or deployment manager processes (Figure 1-17).

```
<table>
<thead>
<tr>
<th>Select</th>
<th>Node</th>
<th>Server</th>
<th>Transport channel chain</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>WADVS1</td>
<td>nodemailer</td>
<td>DCS</td>
</tr>
<tr>
<td></td>
<td>WADVS1CellManager01</td>
<td>dmgr</td>
<td>DCS</td>
</tr>
<tr>
<td></td>
<td>WADVS1node01</td>
<td>nodemailer</td>
<td>DCS</td>
</tr>
<tr>
<td></td>
<td>WADVS2</td>
<td>nodemailer</td>
<td>DCS</td>
</tr>
</tbody>
</table>

**Figure 1-17  Delete the node agent and deployment manager as bridge interfaces**
```

4. Click **New** on this panel and select the application server processes (at least two or three) to act as dedicated core group bridge access points. Click **OK**.

If you need to create new application servers as access points for each core group before this step, use the instructions in the IBM Information Center article *Specifying a core group when creating an application server*, available at the following Web site:


5. The new list of bridge interfaces should include the application servers you selected (Figure 1-18).

```
<table>
<thead>
<tr>
<th>Select</th>
<th>Node</th>
<th>Server</th>
<th>Transport channel chain</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>WADVS1</td>
<td>AP_DCG1</td>
<td>DCS</td>
</tr>
<tr>
<td></td>
<td>WADVS2</td>
<td>AP_DCG2</td>
<td>DCS</td>
</tr>
</tbody>
</table>

**Figure 1-18  Bridge interfaces**
6. Repeat this for TestCoreGroup. Go to Servers → Core groups → Core group bridge settings → DefaultAccessPointGroup → Core group access points.

Select CGAP_2\TestCoreGroup and click Show Details → Bridge interfaces. Delete the existing nodeagent/deployment manager processes and select the application servers to use as access points. The results are shown in Figure 1-19.

![Figure 1-19 Core group access points](image)

7. As a result of updating the bridge interfaces for these two core groups, there are four new dedicated application servers designated as access points of DefaultAccessPointGroup.

Go to Servers → Core groups → Core group bridge settings and expand the tree of DefaultAccessPointGroup to see the new bridge settings.

![Figure 1-20 Core group bridge settings](image)

**Add dynamic clusters to new core groups**

If you need to create new dynamic clusters as a member of newly created core groups, you can use the wsadmin command instead of creating it from the console.

See the IBM Information Center article *Dynamic cluster administrative tasks*, available at the following Web page, for more information:

Figure 1-21 on page 27 shows an example of using wsadmin to create a new dynamic cluster and assign it to a core group other than the default core group.

![wsadmin script to create a new dynamic cluster](image)

After the script completes, you can view the new dynamic cluster in the administrative console by selecting Servers → Dynamic Clusters (Figure 1-22).

![List of dynamic clusters](image)

To see the core group assignment, select Servers → Core groups → Core group settings → TestCoreGroup → Core group servers (Figure 1-23).

![Display the core group servers](image)
1.6 Isolated cells topology

Simple topologies are often preferred when you consider manageability and stability. However, they often have limitations when it comes to growth. In this section, we introduce a topology that overcomes these limitations by combining multiple standalone topology environments (see Section 1.4, “Standalone topology” on page 10), to create a simplified, large system.

1.6.1 Topology overview

Figure 1-24 shows the isolated cells topology.

This is a large topology with simple communication and a workload that completes within one cell and one core group. Requests sent to the IBM HTTP Server are simply routed to the ODRs. No communication occurs between the cells, or between the core groups.

- Content Based Routing (CBR) needs to be placed as an entry point into this environment. You can choose any product that supports the CBR function to route content-based requests to the appropriate IBM HTTP Servers. You can configure the routing rules based on the target application, client IP addresses, or any other rules that maximize the use of your system. The CBR must be highly available because this is an entry point.

- IBM HTTP Server distributes workloads to the ODRs for the cell. You need to configure multiple IBM HTTP Servers that route to the same ODRs (IHS1.1 and IHS1.2 in Figure 1-24,) to ensure the routing to the associated cell. If you increase the number of cells in parallel, you need to create IBM HTTP Servers in front of each cell environment, like IHS2.1/IHS2.2, IHSn.1/IHSn.2, and so forth, as shown in Figure 1-24.
Each cell is configured as a standalone topology system, which consists of one cell and one core group. The plug-in information generated by the ODRs is copied only to the associated IBM HTTP Servers. Because each cell is a standalone system (without multiple core groups), you need to be careful to keep the number of JVMs in each cell to less than 50.

You can expand the system by increasing the number of standalone cells and IBM HTTP Servers in parallel, then change the rules on the CBR to route subsequent requests to the newly configured cell topology. You can deploy the same application on multiple cells, but each application instance runs individually. You also need to plan the upgrade or maintenance of the cell systems for each cell individually.

**Advantages of this topology**
In this topology, each cell is self-contained and less management traffic is required. This allows you to realize easy manageability. This is a leading topology to exploit Virtual Enterprise features to existing topologies by introducing CBR components.

**Disadvantages of this topology**
When you expand the system, you need to create additional cells and IBM HTTP Servers. This topology also requires additional planning to use content based routing to maximize the use of your whole system.

### 1.6.2 Configuring an isolated cells topology

Perform the following steps to configure an isolated cells topology.

1. Create multiple cells using the steps outlined in Section 1.4.2, “Configuring a standalone topology” on page 12.
2. Configure a CBR in front of the IBM HTTP Servers to distribute requests to the appropriate IBM HTTP Servers.

### 1.7 Silo cell topology

In a RAS (Reliability, Availability and Serviceability) topology, systems are located in different places with the same applications. In this section, we introduce a topology that routes requests from IBM HTTP Servers to two different cells (like multiple remote data centers) with the same applications.

This topology enables you to configure unbridged, ODR topologies which preserve sessions even in the event of ODR outages. With this function, it is possible to configure your topology such that, when an ODR receives misrouted, in-session traffic, it will reroute the traffic to an ODR in another cell that can satisfy the session requests. Thus, it is possible to configure an IBM HTTP Server to route to ODRs in multiple cells and still preserve session affinity.
1.7.1 Topology overview

Figure 1-25 on page 30 shows the silo cell topology.

This topology usually consists of two identical cells with the same applications deployed.

- The load balancer, IBM HTTP Servers, and deployment managers are located and configured as in the standalone topology.
- The two cells are configured individually, each as a standalone topology. You can configure these two cells as clones, or not, as long as the same applications are deployed in the two cells.
- IBM HTTP Servers distribute the workload and keep session affinity to the ODRs.

You can have the workload distributed across all the ODRs in all the cells with load balancing. You can have requests sent to a primary set of dynamic cluster servers and route requests to servers in another cell only when none of the primary servers are available.

The plug-in routing policy is determined by the option you specify when you merge the two plugin-cfg.xml files generated by the ODRs.

- The ODRs route requests within one cell. An ODR in one cell can route to the dynamic clusters in the other cell through the ODRs in the other cell, but only in the case of an outage. This feature is called **Cell Affinity Forwarding**. This feature solves the problem of losing sessions when there are multiple ODRs within multiple, unbridged cells, and the IBM HTTP Server is configured to forward session traffic, either through load-balancing or failover, to more than one ODR.
Cell affinity forwarding

Table 1-1 on page 31 contains an overview of the consequences of a failure in this environment.

<table>
<thead>
<tr>
<th>Failover scenario</th>
<th>IHS / ODR affinity</th>
<th>ODR Cell Affinity Forwarding</th>
</tr>
</thead>
<tbody>
<tr>
<td>Single ODR failure</td>
<td>Session likely lost (66% chance)</td>
<td>Maintained</td>
</tr>
<tr>
<td>Cell failure (all ODRs)</td>
<td>Lost</td>
<td>Lost</td>
</tr>
<tr>
<td>Single ODR maintenance</td>
<td>Not lost, requires quiesce</td>
<td>Maintained</td>
</tr>
<tr>
<td>Cell maintenance (all ODR)</td>
<td>Not lost, requires quiesce</td>
<td>Not lost, requires quiesce</td>
</tr>
</tbody>
</table>

With cell affinity forwarding, a request that arrives at an ODR that is not the original ODR for the session, and is not in the cell of the original ODR, the receiving ODR will attempt to forward the request to an ODR in the original cell. If no ODR in the original cell is available, the ODR in the second cell will take ownership of the request and the session is lost.

In order to preserve sessions when requests are directed to the non-original cell, the ODRs in the recipient cells must have defined generic server clusters (GSCs) for the original cell. You need to configure cell-scoped session failover in each cell using memory-to-memory, a database, or eXtreme Scale.

**Note:** You need to upgrade the system Fix Pack4 (6.1.0.4) in order to maximize the use of these features.

Figure 1-26 shows a session failover scenario where one ODR is down.
The sequence of actions in this scenario are as follows.

1. The IHS routes a request to ODR1.1 in Cell1, which establishes a session with the application server in Cell1.

2. Sometime later, ODR1.1 fails and the IHS routes the request to another ODR, for example, ODR2.1 in Cell2 in an attempt to failover.

3. ODR2.1, seeing that Cell1 handled the original request (through an ODRSESSIONID cookie set by the ODR1.1), forwards the request to ODR1.2 in Cell1.

4. ODR1.2 forwards the request to the original back-end application server, and on the response, sets a new ODRSESSIONID cookie with its own cloneId=1.2 causing the session to be adopted by ODR1.2.

5. The IHS starts using the new ODR1.2 in Cell1 which has "adopted" the session. If all the ODRs in the Cell1 are down, IBM HTTP Server starts routing all requests to the ODRs in Cell2. In this case, the current sessions are lost as shown in Figure 1-27.

---

**Figure 1-27**  Failure scenario: All ODRs in a cell fail

The sequence of actions in this scenario are as follows.

1. The IHS routes a request to ODR1.1 in Cell1, which establishes a session with the application server in Cell1.

2. Sometime later, ODR1.1 and ODR1.2 fail and the IHS routes the request to an ODR in Cell2 in an attempt to failover.

3. ODR2.1 sees that Cell1 handled the original request (through an ODRSESSIONID cookie set by the ODR1.1), but that all ODRs in Cell1 are down. It takes ownership of this request and routes it to the servers in Cell2. The current session is lost in this case.
Advantages of this topology
This topology enables you to keep session affinity and session failover between multiple cells without configuring a core group bridge. It also reduces core group traffic.

Disadvantages of this topology
Session affinity is not maintained between the cells when all ODRs in one cell are down.

1.7.2 Configuring a Silo cell topology

Perform the following steps to configure a silo cell topology.

1. Create a high availability deployment manager in each cell.
   See Step 1 of Section 1.4.2, “Configuring a standalone topology” on page 12.

2. Create at least two ODRs in each cell for high availability without clustering.
   See Step 2 of Section 1.3.2, “Configuring a simple test topology” on page 8.

3. Create the dynamic clusters.
   See Step 3 of Section 1.3.2, “Configuring a simple test topology” on page 8.

4. Install the Web server plug-in on the IBM HTTP Server machines but do not define the IBM HTTP Servers to the WebSphere configuration.

5. Define each IBM HTTP Server as a trusted proxy on each ODR.
   See Step 5 of Section 1.3.2, “Configuring a simple test topology” on page 8.

6. Enable the custom cell property “odrSessionAffinityEnabled.” This property enables IBM HTTP Servers to always route to a specific ODR once a session has been established.

   Save the configuration change and restart WebSphere Virtual Enterprise servers. The plug-in configuration files generated by ODRs instruct the IBM HTTP Server plug-in to use the ODRSESSIONID cookie for its session ID, thereby enabling session affinity to the ODRs spread in multiple cells. You need to enable this custom property in both cells.

   From the administrative console, go to System administration → Cell → Custom properties → New and add the “odrSessionAffinityEnabled” custom property as shown in Figure 1-28.

![Image](Figure 1-28 Create the odrSessionAffinityEnabled custom property)
7. (Optional) Configure a custom ODR cookie name by creating a `odrSessionAffinityCookieName` custom server property.

This will cause the custom cookie name to be used for the ODR session identifier cookie, instead of the default `ODRSESSIONID` cookie. This step is required if your environment is a multi-tiered ODR environment.

A multi-tiered environment is one where ODRs are located in multiple tiers in order to route requests to different target servers placed in different tiers. For example, there may be a cluster of ODRs configured to distribute load across a number of IBM Portal Application Servers at the front end of the network, while at the back end there may be clusters of ODRs to distribute load across multiple Web content managers.

In multi-tiered configurations, the ODRs in one tier must be configured to operate independently from the ODRs in the other tiers. The Cell Affinity Forwarding function must not cross the tier boundaries, and ODRs from one tier should not forward traffic directly to ODRs in another tier.

In a multi-tiered environment, you need to fulfill following three requirements.

- Within a given tier, any Generic Server Cluster configurations created for Cell Affinity must refer only to ODRs within the that tier.

- Within a given tier, the ODR session identifier cookie name (which is, by default, `ODRSESSIONID`) must be unique across all the other tiers. This cookie name must be configured as a custom property on each ODR within that tier. Thus, define a unique-per-tier cookie name for each ODR within your network, using the custom property, `odrSessionIdCookieName`.

From administrative console, go to `Servers → On Demand Routers → odr_name → On Demand Router Properties → On Demand Router Settings → Custom Properties → New`. Add the “`odrSessionIdCookieName`” property, then define a unique-per-tier cookie name as shown in Figure 1-29. You need to repeat this step for all ODRs within the tier.

![Figure 1-29  Defining a custom ODR session cookie name](image)

- If using the Highly Available Plugin Configuration, each plug-in configuration file must define the ODR session identifier cookie name to use for IBM HTTP Server to ODR affinity.

  To configure this, define an additional cell-wide custom property associated with the `plugin-cfg.xml` generation configuration:

  `ODCPluginCfgODRSessionIdCookie<configname>=<CookieName>`
For more information about configuring the Highly Available Plugin Configuration see the IBM Information Center article *Generating the plug-in configuration in a high availability environment*, available at the following Web page:


8. Create a generic server cluster and define the ODRs in the other cell as generic server cluster members.

   a. Go to the administrative console of Cell1, and define ODR2.1 and ODR2.2 as members of a generic server cluster. Do the same from the administrative console of Cell2 to define ODR1.1 and ODR1.2 as members of a generic server cluster.

   b. Go to **Servers** → **Generic Server Clusters** → **New** and create a generic server cluster.(Figure 1-30)

   ![Figure 1-30  Create a new generic cluster](image)

   c. Go to **Servers** → **Generic Server Clusters** → **cluster_name** → **Ports** → **New** and add the ODRs in Cell2 by specifying the host name and port number of the ODRs (Figure 1-31).

   ![Figure 1-31  Add the ODRs to the generic cluster](image)

   d. Execute the same step from the administrative console of Cell2 to define ODR1.1 and ODR1.2 as members of generic server cluster in Cell2.
9. Configure the ODRs to generate the plug-in configuration file automatically, and create a script to copy the updated file to the other location in each cell.

See Steps 6 and 7 of Section 1.4.2, “Configuring a standalone topology” on page 12.

It is recommended that you have all the ODRs generate the plug-in configuration file automatically at the Cell scope in order to keep the plug-in information up to date.

10. Merge the two plug-in configuration files generated by the ODRs in each cell. The Plugin Merge tool, pluginMerge.sh / pluginMerge.bat is provided in the WAS_HOME/bin directory of your WebSphere Virtual Enterprise. This script combines the plugin-cfg.xml files from two, unbridged cells so that the IBM HTTP Server plug-in will either load-balance across the cells, or fail-over to a second (backup) cell.

Usage of this tool is as follows.

```
pluginMerge.sh \[-L\] plugin-cfg1.xml plugin-cfg2.xml merged-plugin-cfg.xml
```

- \(-L\): If specified, the generated merged-plugin-cfg.xml will instruct the IBM HTTP Server Plugin to load-balance traffic to applications which are common to both cells. The IBM HTTP Server will still affine to a particular server for in-session requests, but for out-of-session requests, it will simply select a different ODR for each new requests. (Load balancing)

If -L is not specified, the IBM HTTP Server will configure the servers in plugin-cfg1.xml as the primary servers and the servers in plugin-cfg2.xml as the backup servers. With this configuration, the behavior of the IBM HTTP Server is to send requests to the backup servers only if the primary are unavailable. This is true even if the ODR in the primary cell is routing traffic to the secondary (e.g. original) cell. (Failover merge)

plugin-cfg1.xml: This is the plugin-cfg.xml file generated by an ODR in the first cluster.
plugin-cfg2.xml: This is the plugin-cfg.xml file generated by an ODR in the second cluster.
merged-plugin-cfg.xml: This is the output, merged plugin-cfg.xml.

Note: You need to use WebSphere Virtual Enterprise version 6.1.0.4 or later to maximize the use of this script.

11. Change the plug-in configuration file path in the IBM HTTP Server configuration (httpd.conf file) to point to the above merged-plugin-cfg.xml. The httpd.conf file is located under IHS_HOME/conf directory.

12. Deploy an application and create service policies, transaction classes, work classes, and health policies appropriately.

See Steps 8–10 of Section 1.3.2, “Configuring a simple test topology” on page 8.
1.8 Unlinked cell topology

This topology introduces enhanced support for environments outside of the WebSphere Virtual Enterprise product domain. Application servers that run on other middleware platforms where a WebSphere middleware agent is installed are represented in the product administrative domain. The application placement controller, which provides dynamic application placement, is capable of managing dynamic clusters that are made up of these server types. Some health management is also supported for the other middleware platform servers.

This topology is recommended for customers who want to exploit WebSphere Virtual Enterprise features on existing WebSphere Application Server 5.1 or 6.0 systems, or non-WebSphere systems.

1.8.1 Middleware types

The term middleware server refers to a server on any middleware platform. There are three types of middleware servers:

- Servers with full life cycle support
  This type includes the following servers:
  - WebSphere Virtual Enterprise 6.1 servers and WebSphere Extended Deployment 6.0 servers that are members of WebSphere Virtual Enterprise 6.1 cell
  - ODR
  - Generic server
  - Proxy server
  - Web server
  - Application server
  - WebSphere Application Server Community Edition Version 2 and later
  - PHP servers

- Servers with assisted life cycle support
  This type includes the following servers:
  - Apache Tomcat
  - JBoss Application Server
  - Custom HTTP servers
  - BEA WebLogic Server
  - WebSphere Application Server Community Edition Version 1 (all releases)
  - Apache HTTP Server
  - External WebSphere application server (supported by WebSphere Virtual Enterprise 6.1.0.1 and later)
  - WebSphere Application Server 5.1
  - WebSphere Application Server 6.0

- Discovered servers
  This type includes existing WebSphere Application Server Community Edition Version 1 and later and Version 2 and later servers. These servers are represented as assisted life cycle servers.
The Virtual Enterprise features that apply to these server types depend on the middleware type. Before you begin, you need to understand which Virtual Enterprise management features you can exploit on your existing environment. For more information about understanding Virtual Enterprise features, see the following resources:

- IBM Information Center article *Middleware nodes and servers*, available from the following Web page:
  

- Redbooks publication *Optimizing Operations with WebSphere Extended Deployment V6.1*, SG24-7422

### 1.8.2 Topology overview

Figure 1-32 shows the unlinked cell topology.

![Unlinked cell topology diagram](image)

**Figure 1-32** Unlinked cell topology

In this topology, a WebSphere Virtual Enterprise cell is configured between IBM HTTP Servers and existing middleware platform systems. You can manage multiple middleware nodes by installing the middleware agent on each node.

- The Load Balancer and IBM HTTP Servers are located and configured as described in Section 1.4, “Standalone topology” on page 10.

- The WebSphere Virtual Enterprise 6.1 cell (Cell1 in Figure 1-32) has high availability deployment managers and ODRs. You can also configure WebSphere Virtual Enterprise 6.1 dynamic clusters in this cell to route requests to WebSphere Virtual Enterprise 6.1 dynamic clusters as well as dynamic clusters on other middleware platforms.
A middleware agent is installed on every node that middleware servers (including those with full life cycle support) are running on. The middleware agent communicates with the deployment managers for repository content, like server, application, and policy definitions. The middleware agent also communicates with the ODRs and other agent processes with On Demand Configuration (ODC) messages regarding the runtime status of servers and applications.

**Middleware agent**

The middleware agent is a server process that is used to manage servers that are not running WebSphere Virtual Enterprise. The middleware agent can run on any node. Nodes that run the middleware agent do not need WebSphere Application Server or WebSphere Virtual Enterprise installations on the node. The agent allows you to create representations of these nodes and servers in the administrative console for management purposes.

For more information about the middleware nodes supported by WebSphere Virtual Enterprise 6.1, see the IBM Information Center article *Middleware agent*, available at the following Web page:


**About middleware agents:**

- Middleware agents are not supported on z/OS systems. You cannot federate middleware agents from distributed platforms into a z/OS deployment manager.
- When you install a WebSphere Virtual Enterprise fix pack to your deployment manager, you must also apply the fix pack to your middleware agent installations. The deployment manager and the middleware agents must be at the same fix pack level.

Advantages of this topology

You can introduce WebSphere Virtual Enterprise autonomic manager features into your existing environment with minimum change to your current environment. Configuration is minimal. This includes installing the middleware agent and configuring middleware servers in the administration tools, and creating the dynamic clusters. This topology provides a single administrative domain for multiple middleware platforms, and allows a heterogeneous topology, including multiple WebSphere Application Server releases.

Disadvantages of this topology

With assisted life cycle servers and discovered servers, you cannot exploit the full range of WebSphere Virtual Enterprise features.

1.8.3 Configuring an unlinked cell topology

The following sections detail how to configure an unlinked cell topology.

Configuring the Virtual Enterprise 6.1 cell (Cell1) and IBM HTTP Servers

Follow the steps described in Section 1.4.2, “Configuring a standalone topology” on page 12 and configure a standalone topology. You can ignore the step to create dynamic clusters if you do not need to create and manage WebSphere Virtual Enterprise 6.1 dynamic clusters.
Configuring the other middleware cells

Configure the non-WebSphere or WebSphere Application Server 5.1 or 6.0 cells that you will manage from WebSphere Virtual Enterprise 6.1 by performing the following steps.

1. Install the middleware agent on each middleware system.

   When you install WebSphere Virtual Enterprise to a location in which WebSphere Application Server does not exist, only the middleware agent is installed. The middleware agent on each node must be at the same fix pack level as the deployment manager.

   For details on installing the middleware agent, see the IBM Information Center article *Installing the middleware agent*, available at the following Web page:


   If you need to install the middleware agent on multiple systems, it is useful to use Centralized Installation Manager (CIM). Using CIM, you can install the middleware agent to multiple nodes concurrently from WebSphere Virtual Enterprise administrative console.

   Follow the steps below in order to install the middleware agent using CIM.

   a. To install the middleware agent using CIM, the WebSphere Extended Deployment Optimizations Operation Version 6.1 product must be added to your CIM repository.

      You can add it to your CIM repository when you install the product on your deployment manager node, by selecting the Install Centralized Installation Manager repository check box (Figure 1-33).

      ![Figure 1-33 Adding a product to the Centralized Installation Manager repository](image)

      Alternatively, you can add the product to the repository later by using the Centralized Installation Manager repository tool. To add the product to the repository using CIM repository tool, run the XD_HOME/bin/xd_cimgrrepository.sh (bat) command on your deployment manager node. Then select the WebSphere Extended Deployment Operations Optimization check box to include it in the CIM repository, as shown in Figure 1-34 on page 41.
Chapter 1. Virtual enterprise runtime topologies

In the Installation image directory path text box, click **Browse** and select the path of the product CD package location (Figure 1-35).

For more information about the CIM repository tool, see the IBM Information Center article *Using the centralized installation manager repository tool to add products*, available at the following Web page:

From the administrative console, you can then use the Centralized Installation Manager to install the product and the middleware agent from the repository to the nodes (Figure 1-36).

![Available installations](image)

Figure 1-36  Select the packages to install on the nodes

**Adding eXtreme Scale and Compute Grid to the CIM repository:** You can also add WebSphere Compute Grid (WebSphere Extended Deployment Compute Grid) Version 6.1 and WebSphere eXtreme Scale (WebSphere Extended Deployment Data Grid) Version 6.1 to your CIM repository during the product installation, or using the `xd_cimgrrepository` command.

b. Register your middleware system as an installation target. Go to **System administration → Centralized Installation Manager → Installation targets** and click **Add Installation Target**. You need to specify the host name, user name, password and platform type of the target node on which you want to install the middleware agent. The user name and password must be an operating system administrative user, like root or Administrator. You need to repeat this step for all middleware nodes. For more information, see the IBM Information Center article **Managing installation targets**, available at the following Web page:


c. Install the middleware agent. From the administrative console, go to **System administration → Centralized Installation Manager → Available installations**.

Select Product install as a package type, and WebSphere Extended Deployment middleware agent for non WebSphere servers - 6.1 as an installation package. Click **Show Installation Target** and you can see the hosts you added. Select the check box of the target hosts and click **Install**.

During the installation process, you will need to specify an authentication method to access the installation target. If you select the Use user name and password option, you need to specify an authenticated user name and password on the target host at the next step. If you select the Use Secure Shell (SSH) public/private key authentication option, you need to specify the location and password of SSH key file. At step 4, specify the path of installation target directory and work directory.

For more information about the procedure, see the IBM Information Center article **Installing packages**, available at the following Web page:

2. Federate the middleware nodes to the WebSphere Virtual Enterprise 6.1 cell.

**Note:** Before you federate the middleware nodes, the deployment manager profile must be augmented to a WebSphere Virtual Enterprise (WebSphere Extended Deployment Operations Optimization) profile.

You can federate the node by running the `addAgent.sh` or `addAgent.bat` command from the middleware node computer. The deployment manager needs to be started for the federation to be successful. The command has the following format:

```
addAgent.bat -host dmgr_host -port dmgr_admin_port -user username -password password
```

If security is disabled in the cell, you do not need to specify the `-user` and `-password` option.

**Note:** You cannot federate a middleware node into a deployment manager that is running on the z/OS operating system.

When you execute the `addAgent` command, you will see output that resembles the following script:

```
C:\IBM\WebSphere\MA\bin>addAgent.bat -host WNDVE1 -port 9060
CWXDA0015I: Federating the agent, Tool information is logged in file
C:\IBM\WebSphere\MA\logs\addAgent.log.
CWXDA0018I: Tool actions are being logged in file
C:\IBM\WebSphere\MA\logs\addAgent.log.
CWXDA0014I: Changing agent name to local host name from Default.
CWXDA0005I: Agent node name is updated to WNDVE1_1 as node name with same name detected.
CWXDA0006I: Agent WNDVE1_1 is successfully federated into the cell.
```

*Figure 1-37  addAgent command*
From the administrative console, you can see the new middleware agent node (WNDVE1_1) is added. Select System administration → Middleware nodes (Figure 1-38 on page 44).

From the Select operational action menu, select Start agent and click Run, or execute agent_install_root/bin/StartAgent.bat(sh) to start the middleware agent. When you start the middleware agent from the administrative console, you need to specify a user name and password to start the middleware agent on the target node.

3. Define the existing application servers as WebSphere Virtual Enterprise middleware servers. You can create a new middleware server definition from Servers → Add a server. Select Add an existing server. You may also navigate to Servers → Other middleware servers → server_type → New.

Additional configuration steps for the middleware server vary depending on the middleware server type. You will need to adjust the values for the WebSphere variables for each server type, and configure server operations to stop and start your servers. For detailed steps, see the following articles on the IBM Information Center:

- For a PHP server, see the article Creating PHP servers and PHP dynamic clusters, at the following Web page:

- For an Apache Tomcat servers, see the article Configuring Apache Tomcat servers, at the following Web page:

- For a BEA WebLogic servers, see the article Configuring BEA WebLogic servers, at the following Web page:
– For JBoss servers, see the article *Configuring JBoss servers*, at the following Web page:

– For WebSphere Application Server Community Edition servers, see the article *Configuring assisted lifecycle WebSphere Application Server Community Edition servers*, at the following Web page:

– For custom HTTP servers, see the article *Configuring custom HTTP servers*, at the following Web page:

– For an external WebSphere application server

With 6.1.0.1 and above, you can manage previous versions of WebSphere application servers, from Version 5.1 and later using this server type. See the article *Configuring external WebSphere application servers*, at the following Web page:

4. Create dynamic clusters and add the middleware servers as dynamic cluster members in order to easily manage the servers to host an application.

Be aware that all of the middleware servers you define as dynamic cluster members of one dynamic cluster must have the same applications installed and have the same version of middleware software installed.

From the administrative console, go to **Servers** → **Dynamic clusters** and click **New**. Select the server type (Figure 1-39). If you select WebSphere Application Server or PHP server (these two server types have full life cycle support), you can define cluster members with rules automatically. When you select other server types, you need to define cluster members manually.

![Figure 1-39 Select the server type for the dynamic cluster](image-url)
Select a dynamic cluster template only for dynamic clusters that consist of servers with full life cycle support. Then specify the dynamic cluster properties and finish the dynamic server creation.

For more details on this process, see the IBM Information Center article Creating dynamic clusters, available from the following Web page.

5. Deploy, or define the middleware applications. Different processes are involved for the different middleware application types. When you are using servers with full life cycle support, you can deploy applications from the administrative console. If you are using assisted life cycle servers, you need to deploy applications on the middleware servers first, and then define the applications from the administrative console.

To complete this step, go to Applications → All applications. Click Add, or Applications, then Install New Middleware Application. The Select the type of middleware applications window (Figure 1-40) is displayed.

From the Select application type you can chose from the following options:

- J2EE™ applications
  You can deploy enterprise applications to dynamic clusters. See the IBM Information Center article Deploying enterprise applications, available at the following Web page:

- PHP Hypertext Preprocessor (PHP) applications:
  You can deploy PHP applications to PHP deployment targets, including PHP servers and PHP dynamic clusters. See the IBM Information Center article Deploying PHP applications, available at the following Web page:

- Unmanaged Web applications
  Unmanaged Web applications are installed and configured outside of the product environment, typically on assisted life cycle middleware servers. Configure unmanaged Web applications so that you can route HTTP requests to assisted life cycle servers. By providing some basic information about the application, including context roots, virtual hosts, and servers, the ODR can route HTTP requests to these applications.

  For more information, see the IBM Information Center article Deploying unmanaged Web applications, available at the following Web page:
WebSphere Application Server Community Edition applications

This is supported by WebSphere Virtual Enterprise 6.1.0.1 and later. You can deploy Java 2 Platform, Enterprise Edition (J2EE) applications and Geronimo modules with Geronimo artifacts to WebSphere Application Server Community Edition deployment targets, including servers and dynamic clusters.

For more information, see the IBM Information Center article Deploying WebSphere Application Server Community Edition applications, available from the following Web page:


6. Define service policies, transaction policies, and work classes with realistic response times for your business goal. The steps for defining service policies, transaction policies, and work classes are described in the Redbooks publication Optimizing Operations with WebSphere Extended Deployment V6.1, SG24-7422, or the IBM Information Center article Defining a service policy, available from the following Web page:


7. Implement health policies in order to monitor your application server environment and execute an appropriate action when the predefined health condition is detected as broken. For more information, refer to Redbooks publication Optimizing Operations with WebSphere Extended Deployment V6.1, SG24-7422, or the IBM Information Center article Configuring health management, available from the following Web page:


1.9 Virtualization

As you design your topology, you should be aware of the virtualization features available for use.

- Application infrastructure virtualization

  In a WebSphere Virtual Enterprise environment, you deploy an application to a dynamic cluster rather than to specific application servers. Autonomic managers control the placement of the server instances and how workload is routed for each application. If the workload increases for a specific application, the number of server instances for the dynamic cluster that is hosting the application can increase, using available resources from other applications that are not experiencing increased workload.

- Hardware virtualization

  You can deploy WebSphere Virtual Enterprise on virtualized hardware, such as VMware® ESX, to take advantage of the capabilities provided by the hosting environment.
When you combine application virtualization and hardware virtualization, you can receive more benefit than if you use each of the virtualizations. Figure 1-41 on page 48 illustrates the advantages of application virtualization, hardware virtualization, and the combination of the two virtualizations.

**Figure 1-41  Virtual Enterprise and hardware virtualization together**

For more information about virtualization, see the IBM Information Center article *Virtualization and WebSphere Virtual Enterprise*, available from the following Web page:


**Memory accounting changes for the deployment manager**

Traditionally, Virtual Enterprise does not account for the deployment manager's memory usage. Because the best practice recommendation is to host the deployment manager on a separate node, dynamic cluster instances are not candidates to run on that node. With the advent of support for hardware virtualization platforms like VMware in 6.1.0.3, however, it is quite possible that the deployment manager may reside on the same physical machine (but in a separate virtual machine) as dynamic cluster instances. Therefore, their memory needs to be taken into account when dealing with memory overload protection.

Keep in mind, especially when hardware resources are limited, that the default memory protection (designed to prevent paging) may engage with 6.1.0.3 when it did not engage in previous releases (if the deployment manager was co-located on machines with application servers). Changes to the overload protection customer properties, or the node.memory custom property, can be used to increase the memory the placement controller allocates to a node (at the sacrifice of paging) to get dynamic cluster instances started when in automatic mode in these constrained environments.
Chapter 2. Service policies

Imagine an environment with several applications where all client requests are given the same priority. As an administrator, this makes it difficult to manage your system and to provide the resources where they are most needed. One solution is to install your critical applications in a separate machine to enhance the performance. This probably would not make the most efficient use of your resources. A better solution is to use Virtual Enterprise and to define service policies to categorize and prioritize work requests. Service policies allow you to designate performance goals and the business importance of different request types. WebSphere Virtual Enterprise cannot guarantee that your service goals are met with absolute certainty, but it will strive to manage the system in the best way possible to make it happen.

This chapter discusses best practices for implementing service policies.
2.1 How are service policies used?

A service policy consists of a user-defined performance goal and an importance level. There are three performance goal types in IBM WebSphere Virtual Enterprise:

- **Discretionary**
  This goal type indicates work that does not have significant value. Requests are processed when no higher request is waiting. As a result, work of this type can see a degradation in performance when resources are constrained. This is the default service goal.

- **Average response time**
  This goal type allows you to specify the average response time goal in milliseconds or seconds. The system will try to achieve this goal at a target percentage of 90%.

- **Percentile response time**
  This goal type allows you to specify both the average response time goal and the target percentage. For example, 95% of all requests should be answered in less than 1000 milliseconds. This performance goal type is useful for applications that have application response times that occasionally deviate from the norm and can skew the average response time.

For example, if you specify two seconds as the goal value, 90% of the requests must be answered in less than two seconds to meet the goal. If you need a different percentage, use the percentile response time goal to set it.

As requests enter the On Demand Router (ODR), they are mapped to a transaction class and, by extension, to a service policy. For most work requests, work classes are used to map incoming requests to transaction classes. Each work class is attached to an application and a basic request feature:

- URI prefix for HTTP
- Method name for IIOP
- Bus and destination for Java Message Service (JMS)

Each work class specifies how the relevant requests are classified into transaction classes. For generic server clusters and for SIP, work classes are not used. The rules for classifying requests to transaction classes are configured on the ODRs.

Each work request belongs to exactly one transaction class, and each transaction class belongs to exactly one service policy.

Figure 2-1 on page 51 shows the relationship between service policies, work classes, and transaction classes. The URIs are grouped together in work classes. When a request for a specific URI arrives, this URI is checked against the classification rules. Based on these rules, different transaction classes are addressed. These transaction classes are uniquely assigned to a service policy. The request is processed based on the service policy. A request filter in the ODR handles these steps and classifies the incoming requests into associated service policies.
2.1.1 Work classes

A work class is a grouping of work that is to be done by an application server. WebSphere Virtual Enterprise determines how to handle that work through a set of rules that each work class contains.

There are two main types of work classes:

- Service policy work classes
  Work class rules associate incoming work with a service policy, thus indicating to WebSphere Virtual Enterprise when to forward the work to the application server.

- Routing work classes
  Work class rules associate incoming work with a routing policy, thus indicating to WebSphere Virtual Enterprise where to send the work.

Each unit of work (request, message, or call) is associated with a single service policy work class.

HTTP requests and SIP messages are also associated with a single routing work class. Routing work classes do not exist for IIOP and JMS. Because these protocols do not flow through the ODR, no routing policy is needed.

Work classes combined with classification rules allow the Autonomic Request Flow Manager (ARFM) to prioritize a request. For example the /shop/checkout URI should get more resources than the /shop/info URI. This could be necessary because the checkout takes more time or because you think the checkout is more important.
There are four possible types of work classes based on the supported protocols in the application:

- HTTP work classes
- SOAP work classes
- IIOP work classes
- JMS work classes

Note: For applications running on platforms other than WebSphere Application Server, only work classes based on the HTTP protocol are supported.

2.1.2 Transaction classes

Transaction classes provide the link between applications and service policies. The service policy creates the goal, while the transaction and work classes are used to map requests to that goal. Transaction classes are defined in service policies. The relationship between service policies and transaction classes is one to many. A single service policy can have multiple transaction class definitions, but each transaction class belongs to exactly one service policy.

Transaction classes are a subcontainer of the service policy for work being classified into the service policy that can be used for finer-grained monitoring. They can also be used as a mechanism of grouping cross application work together for common monitoring.

Every service policy has a default transaction class, which in most scenarios is sufficient. Additional transaction classes are created when finer-grained monitoring is necessary for the environment. Each transaction class name must be unique within the cell.

2.2 Leading practices

This section discusses how to plan for service policies based on business needs.

2.2.1 Approach to defining service policies

When you install an application in a Virtual Enterprise environment, it will be assigned the default service policy, Default_SP. Requests for applications with this service policy are not given any priority over other requests.

Start by analyzing the importance level and performance goal of each application. Then run performance tests for the applications to determine their performance profiles on your production environment. The analysis of these tests, should help you determine which applications or parts of applications should be given a higher priority and would benefit from a new service policy.
Implementing a service policy

The process to implement a service policy is as follows:

1. Evaluate the applications and the desired service levels for your environment, and design a strategy for assigning service policies to your applications.
2. Create a service policy and define the transaction classes for the policy.
3. For each work class in the application, define work class rules to classify requests and associate the request to a transaction class (and by extension, a service policy). If the URL of the request matches the pattern defined in the rule, then the request is classified as belonging to the transaction class defined for the rule.

2.2.2 Defining service policy performance goals

The importance of applications with regard to each other and their performance goal is defined by service policies. Service policies are used when trade-off decisions must be made between applications, giving priority for capacity to the applications with the highest importance (that is, when there is not enough capacity to service all incoming requests).

Check service policy goals against the actual performance of the service class and adjust the goals if necessary. For example, if your service policy has an average response time goal of 200 milliseconds and the measured average service time is 300 milliseconds, you might consider adjusting your response time goal to 500 milliseconds, considering the time that the request is queued and waiting to be delivered.

Relevant definitions:

- **Service time**: How long it takes to execute the request on the app server.
- **Response time**: Total time it takes to handle a request including any queuing time.

(\text{Response time} = \text{wait time} + \text{service time})

The performance goal is used as by the ARFM as a guideline when computing how long it is acceptable to queue a request before submitting it for servicing (when queuing is necessary).

It is a leading practice to set the performance goal as the maximum acceptable response time for an application. A baseline service time for the application should be taken when the machine CPU nears the maximum CPU setting that is configured on ARFM. The service policy goal should not be set lower than this service time because WebSphere Virtual Enterprise will not guarantee this performance goal. WebSphere Virtual Enterprise cannot make an application respond faster than it is capable of responding.

2.2.3 Defining work classes

Use work classes to group applications or specific URIs by business priority. Because priorities often change in business, grouping the URIs makes the transition simpler to manage. For example, a company has a Web page that allows employees to see their bonus. This Web page is accessed more often in a specific month of the year, so in that month, the priority of the page is higher.
2.2.4 Defining transaction classes

A URI is associated with a work class, which in turn, is associated with a transaction class defined in a specific service policy. A default transaction class is automatically defined in each service policy. The question is, when do you need to create additional transaction classes? One reason to create a transaction class is for monitoring purposes. Having a specific transaction class for work allows you to create reports specific to a set of URIs to monitor, for example, average response time. This can be helpful in narrowing down application performance for diagnostic purposes.

2.2.5 Defining maximum CPU usage when using vertical stacking

With vertical stacking, multiple server instances can start on a node. By configuring multiple server instances, you can use all the power that is available on the nodes when a large workload exists for the application. You specify whether a dynamic cluster can be vertically stacked when you create the dynamic cluster (Figure 2-2 on page 54).

The ARFM uses the Maximum CPU Usage value defined in the ARFM settings to determine when a CPU is overloaded. When you configure vertically stacked cluster members on a node, the ARFM uses the lesser of the following options as the maximum CPU.

- The value configured as Maximum CPU usage for the ARFM.

Configure this value by selecting Operational policies → Autonomic Managers → Autonomic Request Flow Manager → CPU overload protection (Figure 2-3).

The value calculated from the number of running instances divided by the number of vertical stacking instances configured on the node.
For example, consider the case where the configured CPU limit is 90% and there is 1 server started with a stacking number of 2 instances. In this case, the first value (the first option above) is 0.9, or 90%. The second value (the second option above) is 0.5, or 50%. Because 50% is the lower value, it is used as a limit of the machine CPU.

2.3 Creating service policies

Use the following steps to create a service policy:

1. In the administrative console, select Operational policies → Service policies and click New (Figure 2-4 on page 55).
2. Define the general property values for the service policy (Figure 2-5).
   Enter a name and description for the new service policy and select a goal type:
   – Average response time
   – Discretionary
   – Percentile response time
   Click Next.

![Figure 2-5 Define the service policy properties](image)

3. Define the properties specific to the goal type (Figure 2-6 on page 57).
   a. Enter a realistic goal value.
   b. Select the importance level.
   c. If you want to monitor for persistent service policy violations and have a runtime task created, check the Monitor for persistent violation box, and enter values for the goal delta and time period.
      - Goal Delta Value
        This is the allowable amount of time difference between the configured goal value and the actual average response time of requests that are served.
      - Time Period Value
        This value signifies how long that goal delta value can be violated before it is considered breached, and a runtime task is generated.
   Click Next.
4. Define service policy membership (Figure 2-7 on page 57).

This step defines a transaction class for the service policy. You can select the default transaction class, or create a new one. You can have more than one transaction class. This is usually done in order to chart metrics at a more fine-grained level.

5. Confirm the service policy settings and click **Finish**.
2.4 Associating service policies with an application

With the service policies and transaction classes created, the next step is to define work classes for each application and associate the work class with the transaction class for the service policy. Work classes are associated with each application.

1. In the administrative console, select Applications → Enterprise Applications → application_name → and click the Service Policies tab (Figure 2-8).

![Figure 2-8  Specifying service policy settings for the application](image)

2. Click New to define a new work class for HTTP requests.

3. Enter a name for the new work class and click Next (Figure 2-9).

![Figure 2-9  Define the work class name](image)

4. Define the HTTP patterns that will be mapped to this work class (Figure 2-10).
   a. Select the application module.
   b. Select the HTTP patterns and click Add.
   c. Click Next.

   Note that you can add custom HTTP patterns using the Add Pattern button.
Figure 2-10 Define the HTTP patterns for the work class
4. Confirm work class creation and click **Finish** (Figure 2-11 on page 60)

5. Select an appropriate transaction class this work class.

6. Click **OK** and save the configuration.
2.5 References

The following articles and Web sites provide relevant information for defining service policies:

- IBM developerWorks® article *Service Policies, ARFM & APC - Understanding their relationships*, available from the following Web page:


- IBM developerWorks article *Optimizing operations with WebSphere Extended Development: A checklist for moving solutions into a goals-oriented production environment*, available from the following Web page:


- IBM developerWorks article *Policy-based request routing and quality of service in WebSphere Extended Deployment V6*, available from the following Web page:


- A search on “Service policies” on the developerWorks website delivers the links found at the following Web page:

Health management

The health management subsystem provided with WebSphere Virtual Enterprise allows you to take a policy-driven approach to monitoring the application server environment and to define actions to be taken when certain criteria are discovered.

The health management subsystem consists of two main elements:

- Health policies that define conditions that can indicate a problem, where to monitor for this problem, the action to take, and whether the action is done automatically or by an operator.
- A health controller that monitors the Virtual Enterprise environment for conditions defined by the policies and performs the appropriate actions.
3.1 Using the health management subsystem

Health monitoring should be used carefully, and only defined and assigned to servers if you think that a particular health policy is actually needed. It can help to make your environment more reliable, but can also have performance impacts on the system. Understanding the environment, including its capacity, usage, and loads will help you plan your policies.

Health monitoring is not meant to replace the testing and benchmarking phases of the application development life cycle. We therefore recommend that every application prior to being deployed an in WebSphere Virtual Enterprise environment, should be tested and benchmarked for performance.

3.2 Health policies

A health policy is the definition of specific health criteria that you want your WebSphere Virtual Enterprise to protect itself against. The health management function uses the defined policy to search the environment for software malfunctions.

There are default health policies available for a specific set of conditions. With WebSphere Virtual Enterprise you can use these cell-level default policies, modify the criteria for the default policy, or create your own.

3.2.1 Defining a health policy

To define a health policy using the administrative console, select Operational policies → Health policies (Figure 3-1).

In the figure above you can see the predefined health policies for the following conditions.

- Excessive memory consumption, which can indicate a memory leak. The default policy sets this at 95% of the JVMN heap size for 15 minutes.
3.2.2 Using Custom health policies

The ability to create a custom health policy is new in V6.1. This was mainly introduced to address the health monitoring needs of the growing number of supported middleware servers. Using a custom health policy enables you to create expressions, defining what “unhealthy” means in your environment, rather than having Extended Deployment define it. For example, because excessive memory consumption (which often indicates a memory leak) is not a predefined health condition supported for WebSphere Application Server Community Edition, you have to write a custom health condition in order to monitor it.

Custom conditions are supported on all platforms, but with varying levels of support. For example, metrics from an ODR can be used for a health policy for all server types. However, a custom health condition for a non-WebSphere server cannot make use of the WebSphere PMI server metrics.

While health policies using predefined health conditions can be configured from the administrative console, a health policy using custom conditions must be defined using jython scripts or interactively using the createHealthPolicy AdminTask command.

For information about using the createHealthPolicy AdminTask command, see the following article:

3.2.3 Leading practices

WebSphere Virtual Enterprise provides a set of default health policies that run in supervise mode. You can use these policies on all the new installations to monitor for any possible issues. The first few days are very critical for all the new applications.

For some shakeout periods we recommend setting up the following health policies for all new or updated applications:

- Memory leak condition
- Excessive request timeout condition

When defining a custom condition, consider the following aspects of your custom health condition:

- The cost of collecting the data.
- Analyzing the data, and if needed, enforcing the health policy.
- The amount of traffic going over your network, especially when you scale out the number of servers that produce data.
Before introducing new health policies into the production environment, perform an analysis of these aspects of your custom health conditions.

When defining new health policies, consider the effect it will have on your production system. For example, if you have only two JVMs supporting a high usage application, you do not want to configure a health policy that could recycle one of them during peak times. This could cause the remaining JVM™ to crack under the heavy load. If you do see a need to recycle a JVM, you can plan a manual recycle before peak time.

### 3.2.4 Reaction mode

The health management subsystem functions, in reaction mode, define the level of user-interaction when the health condition determines corrective action is needed. There are two possible reaction modes:

- **Automatic mode**
  
  When the reaction mode on the policy is set to automatic, the health management system takes action when a health policy violation is detected. The logging data, and the defined reaction are performed automatically.

- **Supervised mode**
  
  The health management system creates a runtime task that proposes one or more reactions. The system administrator can approve or deny the proposed actions. The recommendations on actions are sent to the administrator. If the administrator follows the recommendations, the only action required is selecting a button, and the actions are performed. This option is widely preferred by the administrators who are not yet comfortable with giving the WebSphere Virtual Enterprise total control in performing autonomic actions.

After selecting the policy type and the action that should be performed in the event of a health policy breach, you must specify the reaction mode (Figure 3-2).

![Figure 3-2  Selecting the reaction mode](image)

**Leading practices**

Like any automation feature, the policy action should start off in supervised mode until you feel comfortable that the action is appropriately triggered and is the right level of response. Once you feel comfortable with the action, you can move to automatic reaction mode.
3.2.5 Actions

Actions are ordered steps in the health policy action plan that are carried out by the health controller when the health condition triggers for one or more application servers. The actions in the action plan collectively should resolve the problem, or isolate the server from the runtime environment until deeper diagnostics can be performed. WebSphere Virtual Enterprise provides several predefined health actions that can be assembled into action plans, and provides the option for operators to define custom actions to be executed as well. Table 3-1 shows the possible actions.

Table 3-1   Health actions

<table>
<thead>
<tr>
<th>Health Action</th>
<th>Other Middleware Servers</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Restarting the application server.</td>
<td>Supported</td>
<td>This action is available for all policy types. You can define prohibited restart times in peak load times.</td>
</tr>
<tr>
<td>Taking thread dumps</td>
<td>Not supported</td>
<td>This action is only available for the excessive request timeout condition.</td>
</tr>
<tr>
<td>Take JVM heap dumps</td>
<td>Not supported</td>
<td>This action is only available for memory leak policy type.</td>
</tr>
<tr>
<td>Put server into maintenance mode</td>
<td>Supported</td>
<td>Maintenance mode is used to perform diagnostics, maintenance, or tuning on a node or server without disrupting incoming traffic. Putting a server into maintenance mode allows the remaining requests on the server to be processed. Any requests that have an open session on the server are routed to the server until the session ends or times out. After all requests are completed, the server is moved to maintenance mode. Any new requests are routed to servers that are not in maintenance mode.</td>
</tr>
<tr>
<td>Put server into maintenance mode, and break affinity</td>
<td>Supported</td>
<td>The HTTP and SIP session affinity is broken, and the session is moved to another server running in normal mode.</td>
</tr>
</tbody>
</table>
Leading practices
When selecting an action, consider the implications of the action during peak load times. Make sure that a health policy-triggered action does not cause more problems than it solves during this time.

When you choose to take a thread dump, it is advisable to restart the server also.

3.3 Monitoring health management tasks
To see actions taken by the health controller, navigate to System Administration → Task Management → Runtime Tasks (Figure 3-3).

<table>
<thead>
<tr>
<th>Health Action</th>
<th>Other Middleware Servers</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Take server out of maintenance mode</td>
<td>Supported</td>
<td>After the server reaches a healthy state, it can be reinstated to serve requests. When a server exceeds a memory threshold, putting it in maintenance mode gives it a chance to recover through garbage collection while no new requests were being sent to it. After heap utilization is back to below the threshold, you can then have the server taken out of maintenance mode. Custom action, you define an executable file to run when the health condition breaches. A Custom action must be created before you can use it in a health policy.</td>
</tr>
</tbody>
</table>

Figure 3-3  Monitoring health management tasks
This view shows you the conditions that were detected and the actions taken as a result of your health policies. If there are any actions waiting in supervise mode you will see those here and can act on them.

**Event notification**

You can avoid manual administrative operation for tasks generated by the health controller by using the event notification feature in WebSphere Virtual Enterprise. When notifications are enabled and a task is generated, an e-mail notification is sent to each of the e-mail addresses specified.

To configure and enable the notification feature, go to **System Administration → Task Management → Notifications**.

### 3.4 Health controller

The default settings of the health controller in WebSphere Virtual Enterprise are a good start. To fully utilize the system, however, there are settings that can be adjusted. As with any tuning activity, it is important that you understand the basic needs of your system, the peaks and lows for system usage, and the capacity of the hardware infrastructure.

The health controller settings can be configured by going to **Operational policies → Autonomic managers → Health controller**.

- **Enable health monitoring**
  
  This setting enables or disables the operation of the health controller. When enabled, the health controller continuously monitors the health policies in the system. You can disable the health controller without removing the health policies from the system.

- **Control cycle length**

  This setting specifies the time between consecutive health checks. The value is specified in minutes and ranges from 1 to 60 minutes. Longer control cycles reduce the health monitoring load but could be less efficient, depending on the condition for which you are monitoring.

- **Maximum consecutive restarts**

  This setting specifies the number of attempts made to revive an application server after a restart decision is made. If this number is exceeded, the restarts are disabled for the server.

- **Minimum restart interval**

  This setting controls the minimum amount of time that must elapse between consecutive restarts of an application server instance. If a health condition for an application server is breached during that time, the restart is set to a pending state. When the minimum restart interval passes, the server restart occurs. The value can range from 15 minutes to 365 days, inclusive. A value of 0 disables the minimum restart value.

- **Restart timeout**

  The restart timeout specifies how long to wait for a server to stop before explicitly checking its state and attempting startup. The value for this should take into account the time taken by your application to stop and start.

- **Prohibited restart times**
This setting can be used to specify the time and day of the week during which a restart of an application server instance is prohibited. You can also specify multiple time blocks, if needed.

**Leading practices**
Apply your changes to the Runtime tab and test the changes before committing them. Once you are comfortable with the new settings, save to configuration.

Use the prohibited restart times setting to prevent server restarts automatically in certain situations or during specific times. For example, prohibit restart during troubleshooting or during peak times when a recycle of the JVM would put extra load on the other servers in the cluster, thus creating a domino effect.

### 3.4.1 Custom actions

You can create custom actions that define corrective tasks to perform when a health condition breaches. A custom action can be a Java or non-Java executable file. Custom actions can run on the deployment manager, a node that is hosting servers that breach health conditions, the node where the health management controller is running, or a node that you specify.

**Creating custom actions**

To create a custom action, perform the following steps:

1. From the administrative console, select **Operational Policies** → **Custom Action** and click **New** to define a new custom health action (Figure 3-4).

   Select an action type from one of the following options:
   - Java action
     Can include executable JAR or Java class files
   - Non-Java action
     Can include shell script on UNIX® platforms or batch files (.bat)

   ![Health Management Custom Action](image)
   
   **Figure 3-4  Create a new custom action**

   Click Next

2. Provide the information for the action, including the executable code location, the user name and password to be used, and the type of operating system (Figure 3-5 on page 71).
Click **Next**.

3. Confirm your settings and click **Finish**.

### 3.5 ITCAM and Virtual Enterprise

IBM Tivoli® Composite Application Manager (ITCAM) for WebSphere has advanced capabilities for WebSphere application monitoring. This tool can be used to analyze the cause of problems and provides the deep-dive view into the state of discreet transaction events. This information can be used by support teams to identify the root cause of problems in the system.

By using ITCAM for WebSphere and Virtual Enterprise together, you can build solutions to increase productivity and customer satisfaction. WebSphere Virtual Enterprise focuses on keeping the environment healthy. ITCAM uncovers the source of application performance problems and lowers the amount of time and effort needed to resolve them.

Table 3-2 on page 72 shows a comparison of monitoring functions available in Virtual Enterprise and ITCAM.
<table>
<thead>
<tr>
<th>Application Health Monitoring/Management Function</th>
<th>WebSphere VE</th>
<th>ITM / ITCAM</th>
</tr>
</thead>
<tbody>
<tr>
<td>Virtualized Environment (ODR, dynamic cluster, node group, etc.) monitoring</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>Application service level goal based monitoring, reporting and take action</td>
<td>X</td>
<td></td>
</tr>
<tr>
<td>Long running/Batch Application (job execution, job queuing, and so forth) metrics</td>
<td>X</td>
<td></td>
</tr>
<tr>
<td>Cluster Storm Drain detection and take action (rephrase to be clear)</td>
<td>X</td>
<td></td>
</tr>
<tr>
<td>Metrics derived by algorithms and calculations</td>
<td>X</td>
<td>In Tivoli Enterprise</td>
</tr>
<tr>
<td>Application Health Policies</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>Throughput and Response Time</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>J2EE Resource Consumption Trends (PMI)</td>
<td>\ runtime extensions</td>
<td>X</td>
</tr>
<tr>
<td>Server / App availability</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>Memory Leak Detection</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>Hung Server detection and take action</td>
<td>(autonomic)</td>
<td>(manual with ITCAM and autonomic with ITM situations)</td>
</tr>
<tr>
<td>CPU and JVM Memory Stats/Health</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>Charting, visualization, graphical and tabular data views</td>
<td>VE environment and apps</td>
<td>J2EE runtime and apps</td>
</tr>
<tr>
<td>Lock Contention Analysis</td>
<td></td>
<td>X</td>
</tr>
<tr>
<td>Memory Leak Isolation/root cause identification/object allocation</td>
<td></td>
<td>X</td>
</tr>
<tr>
<td>Transaction Decomposition (by transaction, instance, server, app etc.)</td>
<td></td>
<td>X</td>
</tr>
<tr>
<td>Java Component level analysis (JDBC™, JNDI, JMS etc.)</td>
<td></td>
<td>X</td>
</tr>
<tr>
<td>Application Health Monitoring/Management Function</td>
<td>WebSphere VE</td>
<td>ITM / ITCAM</td>
</tr>
<tr>
<td>--------------------------------------------------</td>
<td>-------------</td>
<td>-------------</td>
</tr>
<tr>
<td>Method-level analysis including CPU and timing metrics</td>
<td>When you are using Virtual Enterprise together ITCAM, the best practice at this time, to configure into ITCAM situations/traps that mimic the same unhealthy conditions defined into WebSphere VE's runtime. Having the unhealthy criteria by ITCAM in addition to WebSphere VE allows for correlation of application problems between the two products while leveraging ITCAM's powerful capabilities of root diagnosis and deeper metric collection across the multiple tiers the application may span.</td>
<td></td>
</tr>
</tbody>
</table>
Application hosting and chargeback

WebSphere Virtual Enterprise can be used to provide a shared infrastructure for multiple business units. Moving to a shared infrastructure means subscribing to a shared services model with shared resources, enabling new processes around application development, deployment, and management in a new environment.

Chargeback is the concept of determining the system resources each business unit is using in order to charge for the use. Not only does a shared environment need a mechanism for actually sharing the cost, but to calculate and charge the cost based on consumption. The challenge of chargeback revolves around devising an equitable chargeback model that is agreeable to all participating business units.
4.1 Collecting chargeback metrics

Through its visualization features, Virtual Enterprise can log historical performance metrics. This facility can be used in production to produce statistics for both transactional and batch work. The logs are simple comma-separated-value (CSV) text files. While Virtual Enterprise will generate a raw CSV file with chargeback information by application (and resources consumed by them), the file will have to be imported into a tool (or at least a spreadsheet) to tabulate the data. IBM Tivoli Usage and Accounting Manager (ITUAM) can be used for this purpose.

4.2 Leading practices

When designing a system for chargeback, use the following guidelines:

- Keep your metrics simple. That way you and your business users can understand them.
- Favor business value metrics over IT usage metrics. Your end users will prefer them. Also it gives IT an incentive to become more efficient.
- Ensure that your chargeout policy is compatible with the accounting standards and procedures for your business.
- Calculate your chargeout rates based on expected usage patterns. (Build a spreadsheet model or similar.)
- Negotiate with your end users to agree on the basis of re-charge rather than trying to dictate. But keep it simple, and make sure you can implement anything you agree on.
- Perform risk analysis based on possible usage patterns.
- Perform volume tests to stress test your system, and ensure that you get expected chargeout values.
- If you can, run a trial period to generate statistics and reports before using them as a way of automatically generating chargeback.
Product maintenance

Periodic product maintenance is important to keep your system environment healthy and to avoid trouble caused by known issues. When you plan for WebSphere Virtual Enterprise 6.1 product maintenance, you also have to consider maintenance for your WebSphere Application Server environment and the Java 5 SDK/Java 2 SDK environment. IBM HTTP Servers and Plug-in products also need to be maintained periodically.

WebSphere Virtual Enterprise V6.1 features, such as Centralized Installation Manager (CIM) and maintenance modes ease the maintenance process. Using CIM, you can install the fixes to all of the WebSphere Virtual Enterprise environment from the administrative console. Using Maintenance Mode, you can stop routing from the On Demand Router (ODR) to the nodes or servers placed into Maintenance Mode and maintain these nodes or servers with minimum disruption to your environment.
5.1 Approach to maintenance

The following information outlines an approach for applying maintenance in a Virtual Enterprise environment.

General guidelines include:

- Be sure to get the latest Update Installer before starting to apply maintenance.
- Before you apply maintenance to your production environment, perform a test run by applying it to a test environment and ensure your applications are not affected.

Maintenance should be applied in the following order:

- Apply maintenance to the deployment manager.
- Apply maintenance to the middleware agents so that they are at the same fix level as the deployment manager.
- Apply maintenance to the WebSphere Application Servers, JEE 5 SDK/Java 2 SDK, IBM HTTP Servers and WebSphere Application Server Plug-in environments as well as WebSphere Virtual Enterprise product.

To perform the update, complete the following steps:

1. Stop all WebSphere and associated processes on the deployment manager node, then apply fixes to the deployment manager node and restart.
2. Move a node into Maintenance Mode and stop all servers in the node. Apply the maintenance to the node.
3. Verify the installation was successful and restart the servers.
4. Repeat these steps for each node until all the nodes are at the same fix level as the deployment manager.

You can also use CIM to apply maintenance packages to multiple nodes concurrently. This approach enables you to keep provide uninterrupted service during the product maintenance cycle.

5.2 Installing maintenance

WebSphere Virtual Enterprise 6.1 and WebSphere Application Server 6.1 product fixes are provided as fix packs or Fixes. For WebSphere V6.0 users, be aware that the terminology has changed in V6.1 with regard to maintenance packaging. See Table 5-1. The V6.1 terminology is explained in detail in the sections that follow.

<table>
<thead>
<tr>
<th>Table 5-1</th>
<th>Solution terminology changes from V6.0 to V6.1</th>
</tr>
</thead>
<tbody>
<tr>
<td>V6.0</td>
<td>V6.1</td>
</tr>
<tr>
<td>Release</td>
<td>Release</td>
</tr>
<tr>
<td>N/A</td>
<td>Feature Pack</td>
</tr>
<tr>
<td>Refresh Pack</td>
<td>N/A</td>
</tr>
<tr>
<td>fix pack</td>
<td>fix pack</td>
</tr>
<tr>
<td>Fix</td>
<td>Fix</td>
</tr>
</tbody>
</table>
Feature packs
WebSphere Application Server V6.1 is introducing Feature Packs, which are free, downloadable product extensions for special environments needing Web services, EJB3 requirements, or Web 2.0. The Feature Packs can be installed on top of WebSphere Application Server V6.1.

Fix packs
Fix packs are currently the standard delivery mechanism for updates. Fix packs have the following characteristics:

- A fix pack is a cumulative package of fixes, such as fix pack 2 (6.1.0.2).
- Fix packs are regression tested.
- Fix packs can be installed on top of a previous fix pack, such as applying V6.1.0.2 on top of V6.1.0.1.
- Fix packs are cumulative. For example, V6.1.0.2 includes all fixes in V6.1.0.1.
- Fix packs uninstall all interim fixes applied to the release since the last fix pack was installed. Therefore, it is necessary to check the list of delivered fixes to determine if an interim fix needs to be reinstalled.
- Brief testing of critical functions with the new fix pack is recommended.

Fix (interim fix)
An interim fix is a single published emergency fix, such as PK12345.

- A fix is an interim fix that resolves one or more product defects.
- A fix can be applied to a release or fix pack where applicable.
- Interim fixes are created when a stand-alone fix is required between fix packs. They are validated by at least one customer prior to being published.
- It is recommended that you test functions affected by the WebSphere Virtual Enterprise or WebSphere Application Server component fixed.

5.2.1 Finding and downloading fixes and fix packs
The latest and recommended fix packs and urgent interim fixes for WebSphere Virtual Enterprise 6.1 and WebSphere Application Server 6.1 are can be found on the following support sites:

- Recommended fixes for WebSphere Extended Deployment
  http://www-01.ibm.com/support/docview.wss?rs=3023&uid=swg27005709
- Recommended fixes for WebSphere Application Server
  http://www-01.ibm.com/support/docview.wss?rs=180&uid=swg27004980

The list of fixes included in each fix pack is also provided for both WebSphere Virtual Enterprise and WebSphere Application Server.

- Fixes by version for WebSphere Extended Deployment
  http://www-01.ibm.com/support/docview.wss?rs=3023&uid=swg27007650
- Fixes by version for WebSphere Application Server
  http://www-01.ibm.com/support/docview.wss?rs=180&uid=swg27006899
Fix packs for WebSphere Compute Grid and WebSphere Extreme Scale are provided at the same time as the fix packs for WebSphere Virtual Enterprise is released.

Fix packs for Java SDK, IBM HTTP Server and Web server plug-ins are also provided at the same time as the fix pack for WebSphere Application Server, and can be downloaded from the WebSphere Application Server support page.

When you apply a fix pack on WebSphere Virtual Enterprise, the required WebSphere Application Server fixes must be applied first. You can download the Network Deployment V6.1 fixes for WebSphere Extended Deployment V6.1 fix packs from the following Web page: http://www.ibm.com/support/docview.wss?rs=3023&uid=swg24018185

Using the Maintenance Download Wizard to download maintenance

You can download fixes for multiple products using the Maintenance Download Wizard for WebSphere Application Server V6.1. Using this wizard, you can download fixes and the corresponding Update Installer.

1. To start the Maintenance Download Wizard, go to the Maintenance Download Wizard for WebSphere Application Server V6.1 at the following Web page:
   http://www-01.ibm.com/support/docview.wss?rs=180&uid=swg27009393

2. Select a WebSphere Application Server fix pack level to which you want to upgrade (Figure 5-1).

   ![Figure 5-1](image)

   Figure 5-1  Selecting a fix pack level from the Maintenance Download Wizard

3. Click Next and select the platform for your environment.
4. Click **Next**, and select the **Custom** check box as your target product environment, as shown in Figure 5-2 on page 81.

![Figure 5-2 Selecting the target product environment](image)

5. Select the fix packs you want to apply and the Update Installer (Figure 5-3). You will need to scroll down through the list to find all the selection options. Product fix packs that relate to WebSphere Virtual Enterprise 6.1 environments are:

- Application Server fix pack
- IBM HTTP Server fix pack
- Web Server Plug-in fix pack
- Java SDK 1.5.0 fix pack
- Update Installer

**Note:** Feature pack fix packs are selected by default. Scroll down to deselect that option. You should also refresh the Update Installer for V6.1. You can select that feature on this same page.

![Figure 5-3 Selecting fix packs](image)
6. Click **Next** to select interim fixes and the update installer (Figure 5-4 on page 82).

![Figure 5-4 Selecting interim fixes](image)

7. Click **Next** and agree to the Terms and Conditions. Download the packages you selected.

**Note:** You still need to download WebSphere Virtual Enterprise fix packs and the prerequisite WebSphere Application Server fixes separately from the following Web pages:
- Recommended fixes for WebSphere Extended Deployment
- Network Deployment V6.1 fixes for WebSphere Extended Deployment V6.1 fix packs

### 5.2.2 Installing the Update Installer

When you install fixes on WebSphere Virtual Enterprise and WebSphere Application Server, you first need to download and install the Update Installer. You can download the Update Installer using the Maintenance Download Wizard with product fixes, or individually from the following Web page:


You need to use the latest Update Installer every time you apply fixes. Only one copy of the Update Installer should be installed on your system at any one time for use with all Version 6.x products. If your environment has an existing Update Installer, you must first remove the existing Update Installer before installing a newer version of the Update Installer.

**Important:** Before you install the Update Installer, all WebSphere Virtual Enterprise processes, WebSphere Application Server processes, and related processes should be stopped.
To install the Update Installer, extract the downloaded Update Installer zip files, and execute the install (install.exe) command located in the UpdateInstaller directory. Start the installation wizard. For more information about the steps to install the Update Installer, see the IBM Information Center article *Installing the Update Installer for WebSphere Software*, available at the following Web page:


### 5.2.3 Applying the maintenance using the Update Installer

When you complete the installation of the Update Installer, go to the installation root of the Update Installer, and execute the `update.sh` (*update.bat*) command.

The Update Installer wizard will ask you to select the WebSphere Virtual Enterprise product installation path, the maintenance operation type, and the directory where the fixes are located. If you downloaded the fixes into the same directory, you can apply the multiple fixes to your WebSphere Virtual Enterprise environment simultaneously using the Update Installer.

Figure 5-5 shows an example of how you can select WebSphere Virtual Enterprise fix pack 4, WebSphere Application Server fix pack 19 and Java SDK fix pack1.

![Image of Update Installer](image.png)

*Figure 5-5  Selecting multiple maintenance packages to install*

The default fix location the Update Installer search for is `Update_Installer_Home/maintenance`.

### 5.3 Using the Centralized Installation Manager

When your WebSphere Virtual Enterprise environment consists of multiple nodes, it is a complex step to apply the all product fixes on each node individually. WebSphere Virtual Enterprise provides the Centralized Installation Manager (CIM) feature, which enables you to install the WebSphere Virtual Enterprise product, the Update Installer and product maintenance packages to the federated WebSphere Virtual Enterprise nodes from the administrative console. The CIM is installed and you can select to add the products to its
repository when you install WebSphere Virtual Enterprise. After installation, you can also use
the CIM with your WebSphere Compute Grid Version 6.1 product and WebSphere eXtreme
Scale (WebSphere Extended Deployment Data Grid) Version 6.1 product.

Notes:

In order to use the CIM feature, you must install WebSphere Virtual Enterprise on your
deployment manager.

The CIM does not install products and maintenance on the deployment manager, so you
have to install WebSphere Virtual Enterprise products or maintenance packages on the
deployment manager first.

When you install the CIM on the deployment manager, the WXD_HOME/cimrepos directory is
created as the CIM repository.

You can perform the following installation types using the CIM.

- Product install
- Refresh pack, fix pack, or maintenance tool
- Interim fix

5.3.1 Installing products

You can install the following products using the CIM. In order to add the products to your CIM
repository, you need to select to add the products during the product installation steps. You
may add the products later using the CIM repository tool. For an example of adding products
to your CIM repository, see Section 1.8.3, “Configuring an unlinked cell topology” on page 39.

- WebSphere Virtual Enterprise (WebSphere Extended Deployment Operations
  Optimization)
- WebSphere Extended Deployment Compute Grid
- WebSphere eXtreme Scale (WebSphere Extended Deployment Data Grid)
- WebSphere Application Server Community Edition Version 2.0 (supported by WebSphere
  Virtual Enterprise 6.1.0.1 and later)

To use CIM, from the administrative console, select System administration → Centralized
Installation Manager → Available installations. From the Select a package type drop-down
menu, select Product install (Figure 5-6 on page 85).
If you choose a package that includes features, select each feature from the Select a feature drop down menu. This menu does not appear if your installation package that does not include separate features (Figure 5-7).

5.3.2 Installing a middleware agent

To install the middleware agent or the eXtreme Scale client, you need to register the installation target nodes to CIM before the install. To do this, select System administration → Centralized Installation Manager → Installation Targets. For an example of registering installation target nodes, see Section 1.8.3, “Configuring an unlinked cell topology” on page 39.

5.3.3 Installing refresh packs, fix packs, maintenance tools, or Interim fixes

Using the Centralized Installation Manager, you can install following maintenance items.

- Update Installer for WebSphere Application Server Version 6.1
- WebSphere Virtual Enterprise refresh packs, fix packs, and interim fixes
- WebSphere Application Server Version 6.1 refresh packs, fix packs, and interim fixes
Getting the maintenance files

To install maintenance using CIM you first need to determine if you have the files on your system. You can check this through the administrative console. If you do not have the files, you can use the CIM interface to find and download them.

1. Select **System administration → Centralized Installation Manager → Available installations**.
2. Select **Refresh pack, fix pack, or maintenance tool** (Figure 5-8) or **Interim fix** (Figure 5-9) as the package type and select the package to install.

![Figure 5-8 Selecting a refresh pack, fix pack, or maintenance tool](image1)

![Figure 5-9 Selecting an interim fix](image2)
3. If the package you want to install shows in the options, skip to “Installing maintenance packages” on page 89. Otherwise, Add the package by selecting Add or Remove Packages, or go to System administration → Centralized Installation Manager → Add or Remove Packages (Figure 5-10 on page 87).

![Figure 5-10   Adding installation packages](image)

4. Click Add Packages and download the installation package descriptors from the IBM support page, or from an alternative ftp URL.

5. Click Go and select the package descriptors you want to download (Figure 5-11).

![Figure 5-11   Download the package](image)

6. Click Download and download the package descriptors. The downloaded package descriptor is placed in WAS_HOME/properties/cim directory.

   If your deployment manager cannot access the network, you need to download the package descriptor files from another machine that can connect to internet, and copy the descriptor file manually under the WAS_HOME/properties/cim directory of the deployment manager node.

   For more information about this step, see the IBM Information Center article Manually adding files to the repository, available at the following Web page:

7. Download the binary files (.pak files) associated with the package descriptor by navigating to **System administration → Centralized Installation Manager → Add or Remove Packages** and click the package descriptor.

To download a fix pack or the Update Installer, select one or more platforms (or Multi-platforms) in the table and click **Download** (Figure 5-12 on page 88).

![Figure 5-12   Download a fix pack](image1)

To download an Interim Fix, in the package descriptor page, click **Add files** (Figure 5-13). You can select and download Interim Fixes from the IBM support ftp site.

![Figure 5-13   Download an interim fix](image2)
For more information about how to download package descriptors and binary files from administrative console, see the IBM Information Center article *Downloading package descriptors and the associated binary files*, available from the following Web page:


You can also add binary maintenance package files by copying the packages to the CIM repository directory manually. Then you can select the packages from the administrative console. The WXD_HOME/cimrepos directory can contain the following sub-directories. You need to add the packages to appropriate directories.

- **ND61Updates**: WebSphere Application Server Network Deployment interim fixes
- **ND61FP_n**: WebSphere Application Server Network Deployment fix packs, including Java SDK fix pack.
- **UPD61**: Update Installer
- **WXD61**: WebSphere Extended Deployment ppproducts
- **XD61Updates**: WebSphere Extended Deployment interim fixes
- **XD61FP_n**: WebSphere Extended Deployment fix pack.

**Installing maintenance packages**

Once you have the files, install the maintenance packages by performing the following steps:

1. Select **System administration** → **Centralized Installation Manager** → **Available installations**.

2. Select **Refresh pack, fix pack, or maintenance tool** (Figure 5-8 on page 86), or **Interim fix** (Figure 5-9 on page 86) as the package type and select the package to install.

3. Click **Show installation targets** to list applicable target nodes, select one or more installation targets, and then click **Install** to start the Installation wizard (Figure 5-14).

![Figure 5-14   Select the installation targets and start the installation](image)
During the installation process, you will need to specify an authentication method to access the installation target. If you select the **Use user name and password** option, you need to specify an authenticated user name and password on the target host at the next step. If you select the **Use Secure Shell (SSH) public/private key authentication** option, you need to specify the location and password of the SSH key file. At step 4, specify the path of the installation target directory and work directory.

For more information about the installation procedure, see the IBM Information Center article *Installing packages*, available at the following Web page:


### 5.4 Using maintenance mode

WebSphere Virtual Enterprise product enables you to maintain your environment without disrupting incoming traffic to a node or server in a production environment by setting the node or server to maintenance mode. When a node or server is in maintenance mode, the ODR stops routing requests to that server. The Application Placement Controller also excludes this node or server from all automatic application placement.

There are four options for the server level maintenance mode setting, and three options for the node level maintenance mode setting.

- **Maintenance mode**
  
  Use this option to move a node or server into maintenance mode after all requests which have an open session to the node/server are completed. Any new requests are not routed to a node or server set into maintenance mode.

- **Maintenance mode—break affinity**
  
  This option applies to server level maintenance mode setting only. This option moves the selected servers into maintenance mode and breaks HTTP and Session Initiation Protocol (SIP) request affinity to the server.

- **Maintenance immediate stop**
  
  This option stops a server or all processes in a node or server and immediately moves the node or server to maintenance mode. Remaining requests which have an open session are not routed to the node or server. The node or server continues to be in maintenance mode when you restart.

- **Normal**
  
  This option moves the node/server out of maintenance mode, and ODR begins to route new requests to the node or server.

You can put a node or server in maintenance mode from the administrative console or by using a wsadmin script.
5.4.1 Putting a node into maintenance mode

Perform the following steps to put a node into maintenance mode from the administrative console:
1. Go to **System administration → Middleware nodes**,
2. Select the nodes you want to move into maintenance mode and type of maintenance mode.
3. Click **Set mode**.

Putting a server into maintenance mode

Perform the following steps to put a server into maintenance mode from the administrative console:
1. Go to **Servers → All servers**.
2. Select the server you want to move into maintenance mode, and select a type of maintenance mode.
3. Click **Set mode**.

5.4.2 Future reference

For more information about how to set a server or node into maintenance mode from administrative console and from wsadmin command, see the following IBM Information Center articles:

- **Setting maintenance mode**
  This article is available from the following Web page:
- **Maintenance mode administrative tasks**
  This article is available from the following Web page:

5.5 Verifying the product maintenance level

You can check the results of the installation in a variety of ways:

- result from the Update Installer wizard
- result of the CIM wizard
- SystemOut log
- Updatelog.txt file
- By verifying the node level from the administrative console
- By using the **versionInfo** command to verify the installation is at the level you installed.

We recommend that you check the updatelog.txt log and use the **versionInfo** command.
5.5.1 Update Installer wizard

When you install maintenance packages using the Update Installer, you can see the result (success or failure) on the installation wizard (Figure 5-15 on page 92). If you see “Partial Success” or “Failed” in the results, check the updatelog.txt file for the reason.

![IBM Update Installer for WebSphere Software 6.1.0.19](image)

Figure 5-15  Installation results from the Update Installer

5.5.2 Centralized Installation Manager wizard

When you install a maintenance package using CIM, you can see the installation history by navigating to System administration → Centralized Installation Manager → Installation History (Figure 5-16).

![Installation history](image)

Figure 5-16  Installation results in CIM

You can also check the status of installation progress by navigating to System administration → Centralized Installation Manager → Installations in Progress while the installation progress is running (Figure 5-17 on page 93).
5.5.3 The updatelog.txt file

If there are any problems during installation, or when you try to start the environment the first time, check the results in the updatelog.txt file. The updatelog.txt file is located in WAS_HOME/logs/update/package_name.install directory. For example, when you install 6.1.0-WS-WXDOP-FP0000004.pak file, the updatelog.txt file is created in WAS_HOME/logs/update/6.1.0-WS-WXDOP-FP0000004.install. If you uninstall a fix with the Update Installer, updatelog.txt is created in WAS_HOME/logs/update/package_name.uninstall. Look for “INSTCONFSUCCESS” at the bottom of the file. If the result is “INSTCONFPARTIALSUCCESS” or “INSTCONFFAILED,” check the error description and contact IBM if the problem persists.

Example 5-1 Successful completion indicated in the log

(Sep 18, 2008 10:43:36 AM), Install, com.ibm.ws.install.nl.ismp.actions.ISMPLogFileAction, msg1, Starting a new update installer run
(Sep 18, 2008 10:43:36 AM), Install, com.ibm.ws.install.nl.ismp.actions.ISMPLogFileAction, msg1, Java Install Path: C:\IBM\WebSphere\AppServer\UpdateInstaller\java\jre
(Sep 18, 2008 10:43:36 AM), Install, com.ibm.ws.install.nl.ismp.actions.ISMPLogFileAction, msg1, OS Name: Windows® XP
(Sep 18, 2008 10:43:36 AM), Install, com.ibm.ws.install.nl.ismp.actions.ISMPLogFileAction, msg1, OS Architecture: x86
(Sep 18, 2008 10:43:36 AM), Install, com.ibm.ws.install.nl.ismp.actions.ISMPLogFileAction, msg1, OS Version: 5.1 build 2600 Service Pack 2
(Sep 18, 2008 10:43:36 AM), Install, com.ibm.ws.install.nl.ismp.actions.ISMPLogFileAction, msg1, Current User ID: Administrator
(Sep 18, 2008 10:43:36 AM), Install, com.ibm.ws.install.nl.ismp.actions.ISMPLogFileAction, msg1, Current User Home: C:\Documents and Settings\Administrator
(Sep 18, 2008 10:43:36 AM), Install, com.ibm.ws.install.nl.ismp.actions.ISMPLogFileAction, msg1, Current Working Directory: C:\IBM\WebSphere\AppServer\UpdateInstaller
(Sep 18, 2008 10:43:36 AM), Install, com.ibm.ws.install.nl.ismp.actions.ISMPLogFileAction, msg1,

(Sep 18, 2008 10:50:50 AM), Install, com.ibm.ws.install.nl.ismp.actions.SetExitCodeAction, msg1, CWUPI0000I: EXITCODE=0
(Sep 18, 2008 10:50:50 AM), Install, com.ibm.ws.install.nl.ismp.actions.ISMPLogSuccessMessageAction, msg1, {color:blue} INSTCONFSUCCESS
If you select to install multiple fixes at one time, the updatelog.txt log is created in the WAS_HOME/logs/update/install directory. In this case, you can check the result for each fix and the entire install. Look for “INSTCONFSUCCESS” at the end.

Example 5-2 Checking the results of multiple installs

(Sep 18, 2008 9:50:23 AM), Install, com.ibm.ws.install.ni.ismp.actions.ISMPLLogFileAction, msg1, Starting a new update installer run
(Sep 18, 2008 9:50:23 AM), Install, com.ibm.ws.install.ni.ismp.actions.ISMPLLogFileAction, msg1, OS Name: Windows XP
(Sep 18, 2008 9:50:23 AM), Install, com.ibm.ws.install.ni.ismp.actions.ISMPLLogFileAction, msg1, Java Install Path: C:\IBM\WebSphere\AppServer\UpdateInstaller\java\jre
(Sep 18, 2008 9:50:23 AM), Install, com.ibm.ws.install.ni.ismp.actions.ISMPLLogFileAction, msg1, OS Architecture: x86
(Sep 18, 2008 9:50:23 AM), Install, com.ibm.ws.install.ni.ismp.actions.ISMPLLogFileAction, msg1, OS Version: 5.1 build 2600 Service Pack 2
(Sep 18, 2008 9:50:23 AM), Install, com.ibm.ws.install.ni.ismp.actions.ISMPLLogFileAction, msg1, Current User ID: Administrator
(Sep 18, 2008 9:50:23 AM), Install, com.ibm.ws.install.ni.ismp.actions.ISMPLLogFileAction, msg1, Current User Home: C:\Documents and Settings\Administrator
(Sep 18, 2008 9:50:23 AM), Install, com.ibm.ws.install.ni.ismp.actions.ISMPLLogFileAction, msg1, Current Working Directory: C:\IBM\WebSphere\AppServer\UpdateInstaller
(Sep 18, 2008 9:50:23 AM), Install, com.ibm.ws.install.ni.ismp.actions.ISMPLLogFileAction, msg1,

(Sep 18, 2008 9:56:01 AM), Install, com.ibm.ws.install.ismp.actions.InstallListOfMaintenances, msg1, Initializing 6.1.0-WS-WAS-WinX32-FP0000019.pak ......

INSTCONFSUCCESS
**5.5.4 versionInfo command**

You can check the current maintenance level of your environment using the `versionInfo.bat` (sh) command. When you execute the `versionInfo` command without options, it simply shows the current maintenance level of WebSphere Application Server, WebSphere Virtual Enterprise and other WebSphere Extended Deployment products installed on the system.

*Example 5-3  versionInfo command*

```bash
C:\IBM\WebSphere\AppServer\bin>versionInfo.bat
WVER0010I: Copyright (c) IBM Corporation 2002, 2005; All rights reserved.
WVER0012I: VersionInfo reporter version 1.15.4.2, dated 6/5/08

Installed Product
--------------------------------------------------------------------------------
Name IBM WebSphere Application Server - ND
Version {color:blue} 6.1.0.19
ID ND
Build Level cf190836.04
Build Date 9/9/08

Installed Product
--------------------------------------------------------------------------------
Name IBM WebSphere Extended Deployment Compute Grid
Version {color:blue} 6.1.0.4
ID WXDCG
Build Level cf30831.34618
Build Date 7/29/08

Installed Product
--------------------------------------------------------------------------------
Name IBM WebSphere Extended Deployment Data Grid
Version {color:blue} 6.1.0.4
ID WXDDG
Build Level cf30831.34618
Build Date 7/29/08

Installed Product
--------------------------------------------------------------------------------
Name ObjectGrid Component of WebSphere Extended Deployment
Version {color:blue} 6.1.0.4
ID WXDOG
Build Level cf30831.34618
Build Date 7/29/08

Installed Product
--------------------------------------------------------------------------------
Name IBM WebSphere Extended Deployment Operations Optimization
Version {color:blue} 6.1.0.4
ID WXDOP
Build Level cf40836.35253
Build Date 9/3/08

End Installation Status Report
--------------------------------------------------------------------------------
```

If you want to list all installed maintenance packages including Java SDK fixes and interim fixes, run the `versionInfo` command with `-maintenancePackages` option.
The `versionInfo` command has the options shown in Table 5-2 on page 96.

<table>
<thead>
<tr>
<th>Option</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>-?</code> or <code>/?</code></td>
<td>Displays command syntax.</td>
</tr>
<tr>
<td><code>-help</code> or <code>/help</code></td>
<td>Adds a list of installed components to the report.</td>
</tr>
<tr>
<td><code>-usage</code></td>
<td>Adds details about installed components to the report.</td>
</tr>
<tr>
<td><code>-components</code></td>
<td>Adds a list of installed components to the report.</td>
</tr>
<tr>
<td><code>-componentDetail</code></td>
<td>Adds details about installed components to the report.</td>
</tr>
<tr>
<td><code>-file file_name</code></td>
<td>Specifies the output file name. The report goes to standard output (stdout) by default.</td>
</tr>
<tr>
<td>`-format text</td>
<td>html`</td>
</tr>
<tr>
<td><code>-long</code></td>
<td>Creates the long version of the report.</td>
</tr>
<tr>
<td><code>-maintenancePackages</code></td>
<td>Adds a list of applied maintenance packages to the report.</td>
</tr>
<tr>
<td><code>-maintenancePackageDetail</code></td>
<td>Adds details about an applied maintenance package to the report.</td>
</tr>
</tbody>
</table>

For more information about the `infoVersion` command, see the IBM Information Center article `versionInfo command`, available from the following Web page:


### 5.5.5 Administrative console

You can check the current maintenance level of each node from administrative console. Go to **System administration → Nodes** and check the version listed in node column (Figure 5-18). Be aware that you need to restart the node agent and middleware server processes on each node after you install the package, in order to reflect the latest version.

![Figure 5-18 Finding the installation level of the nodes](image-url)
5.5.6 The SystemOut.log file

SystemOut.log is located in WAS_HOME/profiles/profile_name/logs/server_name directory. When a process starts or SystemOut.log file is rotated and newly created, version information of the server is logged in the SystemOut.log file as shown Example 5-4 on page 97.

Example 5-4 SystemOut.log

Start Display Current Environment

**WebSphere Platform 6.1 \[ND 6.1.0.19 cf190836.04\] \[WXDOP 6.1.0.4 cf40836.35253\]** running with process name WNDVE1Cell01\WNDVE1CellManager01\dmgr and process id 4532
Host Operating System is Windows XP, version 5.1 build 2600 Service Pack 2
Java version = J2RE 1.5.0 IBM J9 2.3 Windows XP x86-32 j9vmwi3223fx-20080811 (JIT enabled)
J9VM - 20080809_21892_iHzSMr
JIT - 20080620_1845_r8
GC - 200806_19, Java Compiler = j9jit23, Java VM name = IBM J9 VM
was.install.root = C:/IBM/WebSphere/AppServer
Related publications

The publications listed in this section are considered particularly suitable for a more detailed discussion of the topics covered in this paper.

IBM Redbooks

For information about ordering these publications, see "How to get Redbooks" on page 100. Note that some of the documents referenced here may be available in softcopy only.

- Optimizing Operations with WebSphere Extended Deployment V6.1, SG24-7422
- Best Practices for Implementing WebSphere Extended Deployment, SG24-7343
- Using WebSphere Extended Deployment V6.0 To Build an On Demand Production Environment, SG24-7153

Online resources

These Web sites are also relevant as further information sources:

- WebSphere Virtual Enterprise documentation
- Optimizing operations with WebSphere Extended Development: A checklist for moving solutions into a goals-oriented production environment
- Taming the business and cultural challenges of a shared infrastructure with WVE
- Optimizing operations with WebSphere XD: A checklist
- Policy based request routing and quality of service in WebSphere XD v6
- Dynamic middleware and the 6 attributes of virtualized application serving environments
- Configuring WebSphere XD multi-cell routing
> Optimize resource usage and reduce costs

> Extended WebSphere XD v6.1 functions to previous WebSphere deployments

> Service Policies, ARFM & APC - Understanding their relationships

> IBM developerWorks: Wikis - WebSphere Virtual Enterprise - Home
  http://www.ibm.com/developerworks.wikis/display/xdoo/Home

> XD architects blog

> Education Assistant
  http://publib.boulder.ibm.com/infocenter/iedusst/v1r1m0/index.jsp?topic=com.ibm.iea.doc/welcome.htm

### How to get Redbooks

You can search for, view, or download Redbooks, Redpapers, Technotes, draft publications and Additional materials, as well as order hardcopy Redbooks, at this Web site:

ibm.com/redbooks

### Help from IBM

IBM Support and downloads

ibm.com/support

IBM Global Services

ibm.com/services
WebSphere Virtual Enterprise extends WebSphere Application Server Network Deployment, providing an enhanced quality of service in dynamic operations and extended manageability. WebSphere Virtual Enterprise provides application server virtualization, resource management, and a host of advanced operational facilities, such as performance visualization, health monitoring, and application editions. This combination of capabilities is sometimes referred to collectively as dynamic operations.

This IBM Redpaper publication discusses leading practices for WebSphere Virtual Enterprise Version 6.1. It provides complementary documentation to the WebSphere Virtual Enterprise Information Center. The paper starts by discussing some of the more common topologies for setting up a Virtual Enterprise environment. It provides information about advantages, disadvantages, and guidance on how to implement the topology. It also touches on high focus areas in WebSphere Virtual Enterprise, including service policies, health management, application hosting and chargeback, and product maintenance.

Prior to V6.1.0.3, WebSphere Virtual Enterprise was known as WebSphere Extended Deployment Operations Optimization. WebSphere Virtual Enterprise is now available as part of WebSphere Extended Deployment, or as a separate package.