Part-2 (Automated Deployments): Automated Deployment of Enterprise Application (EAR) Updates

Barry Searle [mailto:searle@ca.ibm.com]
Architect, WebSphere Tools for Automated Build and Deployment
IBM Toronto Lab

Ellen Matheson McKay [mailto:ecmckay@ca.ibm.com]
WebSphere Information Developer
IBM Toronto Lab

Abstract

This two-part article discusses application deployment, particularly automated updates, to IBM® WebSphere® Application Server in a large-scale enterprise environment. It applies to WebSphere Application Server version 5.0, version 5.1, and version 6.0, and also includes an introduction to a few version 6.0 enhancements. This article is not intended to be used as a reference for all the details of WebSphere Application Server administration, but it does describe the key concepts used, and contains a list of references. Although the beginning of the article reviews some fairly basic base server and managed server concepts and operations, much of the remainder of the article will discuss certain complex concepts or operational considerations that will be new even to very experienced enterprise application server administrators.

Part one of this article discussed **`wsadmin`** deployment to base and managed servers. It discussed why phased deployments are needed to maintain applications in a WebSphere Application Server Network-Deployment managed cell, and how to maintain high availability in such an environment.

This part two of the article discusses pre- and post-deployment validation, and it discusses gradual deployment of incompatible versions. It also discusses the design and implementation of a downloadable **`Automated Deployment`** example program that illustrates how to automate the deployment of randomly built collections of enterprise applications or updates, and how to automatically target those applications or updates to the correct servers, including stage-specific application setup.

Pre- and post-validation to maximize availability

In all of the deployment scenarios mentioned in the part 1 of this article, it makes no sense to start deploying an application update if it contains problems that are likely to cause the deployment to fail.

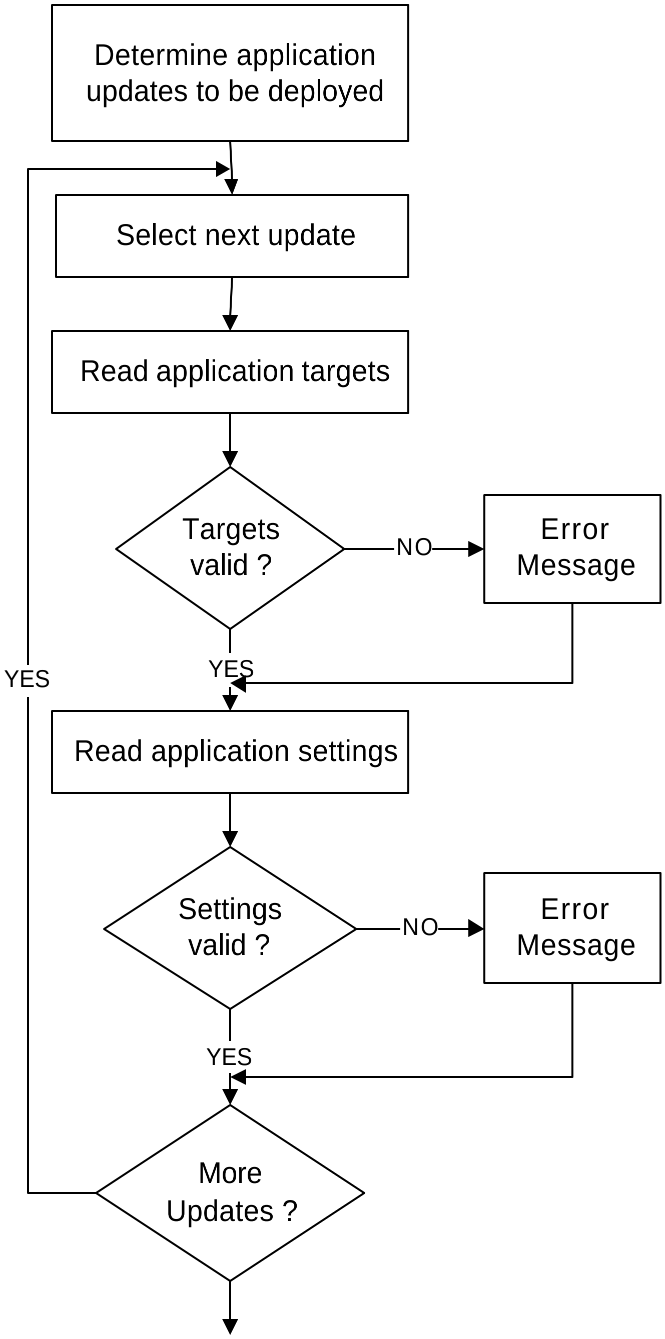
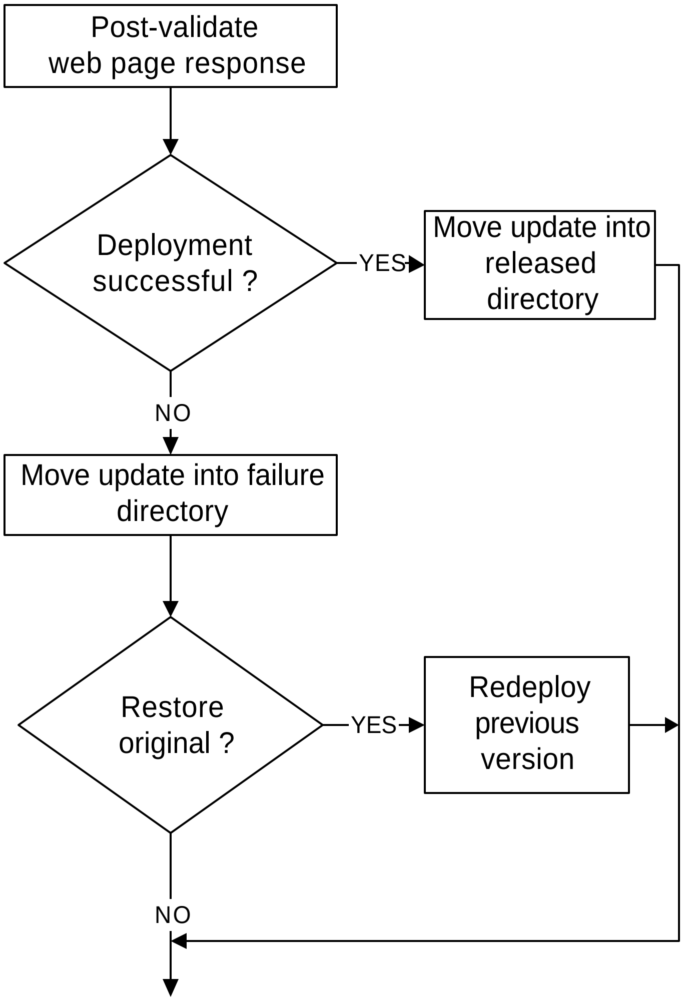


Figure-6: Pre-Validate Deployments

Typical pre-validations can include validating that:

- The application (EAR) appears to be complete and well formed.
- The deployment targets and the required application settings are known.
- The deployment target nodes and servers, or clusters, are valid.
- The deployment target nodes and servers, or clusters, are running and are accessible.
- The application settings are valid.
- The application is already installed, in the case of an application update.
- The application is not already installed, in the case of an application installation.
- Dependencies (prerequisites or resources or interdependent application versions) are met.


Post-validate web page response
Deployment successful ?
YES
Move update into released directory
NO
Move update into failure directory
Restore original ?
YES
Redeploy previous version
NO

Typical post-validations can include validating that:

- The application installed correctly.
- The application started.
- The application appears to be the correct application and is running correctly.
  - One or more Web page HTTP requests return the expected HTTP response.
  - Note: This is not a complete function test of the application, just validation that it is running.

Figure-7: Post-Validate Deployment

Gradual deployment (incompatible Version Migration) using versioned cells

Sometimes a set of interrelated application updates has significant interrelated changes, and the rollout of all those updates must be done at the same time. Of course, enterprise application availability must still be maintained. There is a similar version incompatibility problem if the application update changes the user experience, or uses different database schemas, or uses different HTTP session persistent data. All of these major updates typically require a gradual and carefully controlled rollout of the new version. Many large organizations (financial institutions, hospitals, etc.) require that all updates be handled this way.

An incompatible version update is typically accomplished using two sets of independent production cells. The current production cell is running version=N of all applications, and handling all user requests. Another independent cell then receives deployment updates for version=N+1 of all applications, but it is not active. Network Dispatchers (IP sprayers) provide session affinity, which means that multiple request-response HTTP messages from the same user will be routed to the same Web server and application server for processing. Thus the old version=N cell can be quiesced (allowed to continue processing ongoing HTTP sessions) but the Network Dispatchers (IP sprayers) route new requests to the new version=N+1 cell.

The transition to the new version=N cell can be done either slowly, by gradually increasing the work management load on the new cell, or relatively quickly. Some organizations will initially route selected risk-tolerant users (from selected source HTTP addresses) to the new version=N+1 cell, and only after a successful operation will they reroute the balance of incoming requests. If there is a problem, the work can be routed back to the old cell. Once the transition to the new version=N+1 cell is complete, the old cell is then typically updated to the same version=N+1, or it is used to start a new version=N+2 rollout.

There are many variations on the preceding approach. Some organizations run the parallel cells on separate machines. Others install parallel (but independent) nodes on the same machines. Some organizations just use a single cell with redundant (but independent) clusters, and do a gradual deployment one cluster at a time, using the Web HTTP server (configured with session affinity) to carefully quiesce and reactivate workflow to each of the clusters.

Of course, nearly every organization first extensively tests the new updates in a pilot-production stage, including functional testing and performance/stress testing, before the production rollout is even started.

For an excellent case study of how the New York Stock Exchange (NYSE) and the Securities Industries Automation Corporation (SIAC) achieves extreme availability using gradual rollouts, see the article *"Extreme Availability with WebSphere and DB2"* in the reference section.

Automating enterprise application update deployments

The downloadable *"Automated Deployment"* example program shows a way to simplify the automated deployment of enterprise updates while helping to maximize enterprise application availability. The **`Automated Deployment`** example is designed and implemented to handle the typical enterprise concept of deployment **`Stages`**.

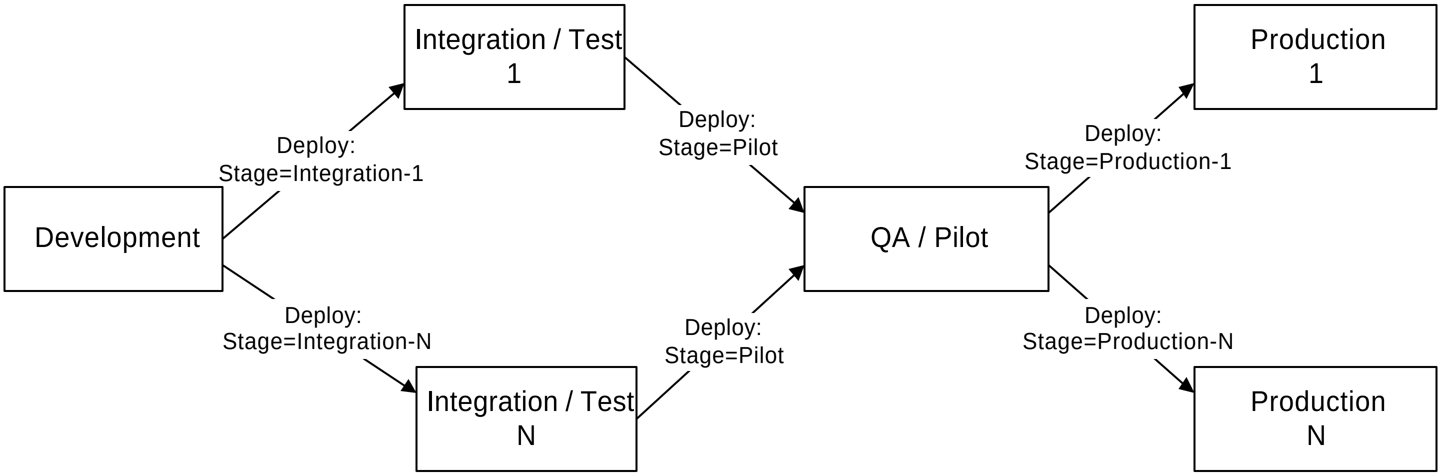


Figure-8: Typical Staged Deployments

The **Automated Deployment** example may be invoked manually, or by some regularly scheduled system program. The invocation must specify the input **Distribution Directory**, the deployment **Action** to perform, and the deployment **Stage** qualifier:

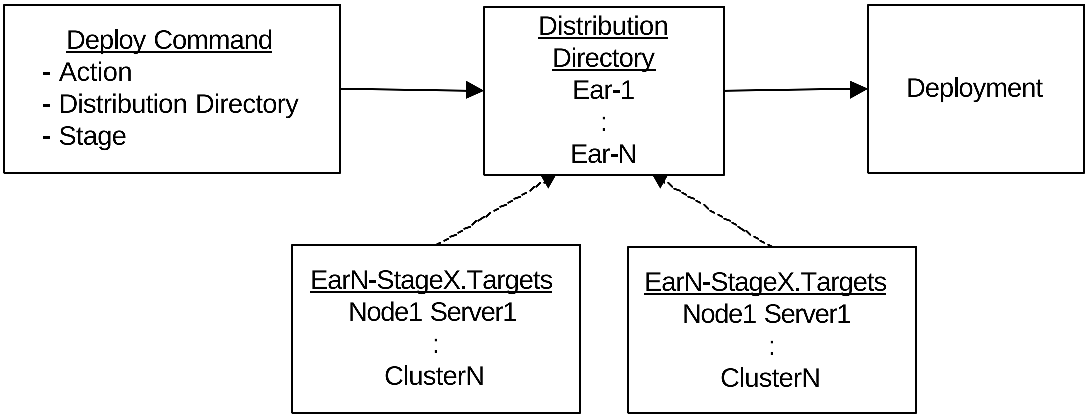


Figure-9: Automated Deployment Inputs

The stage name is just a text qualifier to the application **Target** and **Settings** files, and may represent quality stages (Integration, Pilot, Production, etc.), platforms (Windows®, Linux®, z/OS®, etc.), organization departments (Sales, Finance, etc.), or whatever names or combinations a particular enterprise uses.

If an updated application **myApp.ear** is to be deployed into the **pilot** stage, then a typical **MyApp-pilot.targets** server targets property file (or XML file or database entry, etc.) might be:

```
# multiple entries: nodeserver=nodeName,serverName
# multiple entries: cluster=clusterName
nodeserver = myNode,myServer1
cluster = myCluster

# multiple entries: testURL = URL
# multiple entries: testResult = resultString
testURL = http://myHost:9081/myAppWeb/AdderTest.jsp
testResponse = Java adder(3,4) = 7
testURL = http://myHost:9085/myAppWeb/AdderTest.jsp
testResponse = EJBsessionbean adder(5,6) = 11
```

Similarly, a typical **MyApp-pilot.settings** application settings property file might be:

```
# myApp application EAR
ApplicationName=myApp
startingWeight=9
warClassLoaderPolicy=SINGLE

# myApp webmodule WAR
ModuleName=myAppWeb.war
startingWeight=12121
classloaderMode=PARENT_LAST
```

The overall automated deployment sequence is:

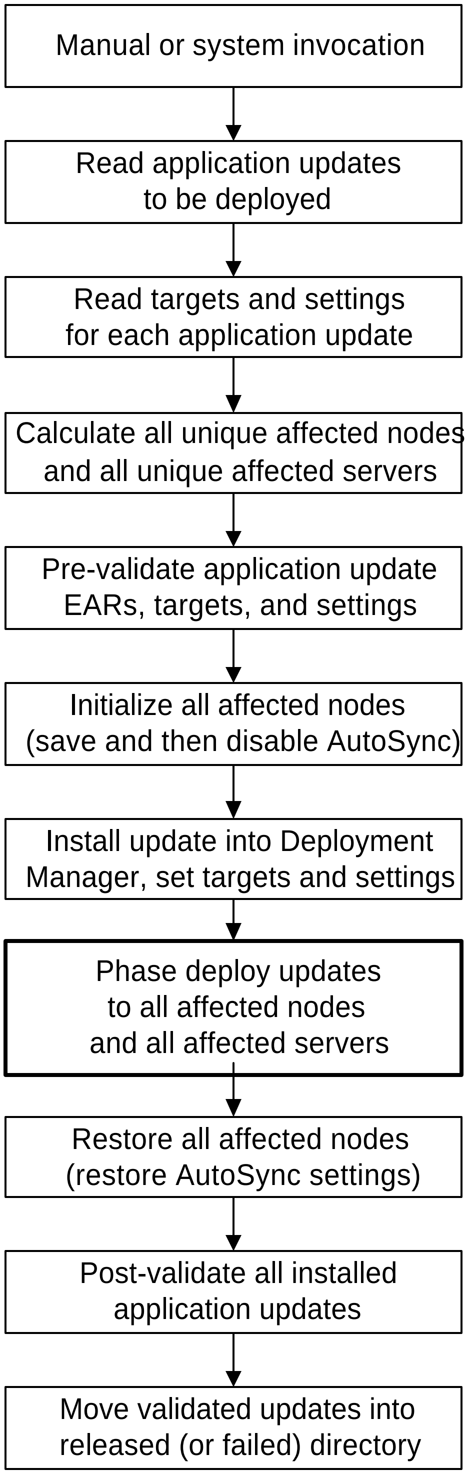

Manual or system invocation
Read application updates to be deployed
Read targets and settings for each application update
Calculate all unique affected nodes and all unique affected servers
Pre-validate application update EARs, targets, and settings
Initialize all affected nodes (save and then disable AutoSync)
Install update into Deployment Manager, set targets and settings
Phase deploy updates to all affected nodes and all affected servers
Restore all affected nodes (restore AutoSync settings)
Post-validate all installed application updates
Move validated updates into released (or failed) directory

Figure-10: Automated Enterprise Update Deployment

1. Invoke the **`automated distribution`** program.
    a. This is typically done from an automated system **`cron`** job, but may be manually invoked.
    b. The invocation command also specifies the **`stage`** to be deployed to.
2. Read the **`distribution directory`** to determine the new application updates to be deployed.
3. For each application update, read its stage-specific server targets and application settings.
4. From the total set of affected nodes and servers, calculate the subset of unique affected nodes and unique affected servers.
5. Pre-validate that the applications and targets and settings are valid.
6. Save and then disable **`AutoSync`** on all affected nodes.
    a. Optionally, you can save and disable **`SyncOnStartup.`**
7. Install the applications into the Deployment Manager repository:
    a. Set the stage-specific application settings.
    b. Set the stage-specific target servers or clusters.
8. Sequentially, for each affected node, phase distribute the updates:
    a. Optionally, quiesce all its affected servers (reroute new work requests).
    b. Stop all its affected servers.
    c. **`NodeSync`** that node to retrieve <u>all</u> updates and install them into the affected servers.
        - *Note:* Wait to ensure the EAR expansion is complete.
    d. Restart the affected servers.
        - *Note:* Test and wait to ensure the server is running.
    e. Optionally, reactivate the affected servers (to process new work requests).
    f. Optionally, validate the installed application operation and request manual confirmation.
9. Restore the previous **`AutoSync`** settings for all affected nodes
    a. Including **`SyncOnStartup`** if it was optionally disabled.
10. Post-validate all applications.
11. Move validated applications into the **`released directory.`**
    a. If it failed, you can attempt to restore the previously released application.
12. Optionally, e-mail the deployment log to a notification list.

As before, there are two special notes in the above steps. First, after performing the **`NodeSync`**, the application update (EAR) has been distributed down to the node, but the EAR file must still be expanded into the server **`installed application directory`**. Until this EAR expansion is complete, attempting to start the server may produce indeterminate results. There is an IBM Problem Report about this, and in the future, there may be a downloadable WebSphere Application Server Interim Fix to allow scripts to test for the completion of the EAR expansion. Second, after returning from the **`wsadmin startServer`** command, the command has been processed by the Node Agent, but the actual server startup may not yet be complete. Scripts need to test that the server has completed startup and is running.

Review of requirements and benefits of automated update deployments

Automated update deployments are intended to:

- Automatically detect current updates to be deployed
- Automatically read stage-specific targets and application settings
- Optionally, read application requirements (prerequisites, resources, interdependent versions, etc)
- Minimize failures by pre-validating as much as possible
- Only update affected nodes and affected servers
- Only cycle affected nodes and servers once each (even in the case of many concurrent updates)
- Only cycle one affected node at a time to maximize application availability:
  - Optionally, quiesce affected servers (to allow complex in-progress HTTP sessions to complete)
  - Stop affected servers (to allow **`server-failover`** and **`session-recovery)`**
  - **`NodeSync`** to retrieve all updates and install them into the affected servers
  - Restart all affected servers, and optionally, reactivate if quiesced
- Minimize the effect of failures by post-validating

Deployment actions

The primary focus of this article has been the deployment of application updates since that is by far the most complex deployment problem. Generically, the most common deployment actions are:

1. Install (a new, non-existent application)
2. Update (a rebuilt application, or rebuilt application components)
3. Reconfigure (change application settings or targets)
4. Uninstall (remove an existing application)

Automated Deployment example program updates and limitation

It is expected that any updates to the current version 1.1 (December 2004) **`Automated Deployment`** example program may be available from the WebSphere Application Server "*Sample Scripts*" Web page in the Developer Domain WebSphere Application Server Library Samples [www.ibm.com/developerworks/websphere/library/samples/SampleScripts.html].

Some limitations in the current example program include:

1. Limited error detection and exception handling (not production quality code).
2. No exploitation of Application Server security (no **`userid`** nor **`password`** passed to **`wsadmin`**).
3. No handling of nested or complex attributes (for application settings).
4. Errors and Warnings are logged and summarized, but no e-mail notification is generated.
5. No handling of incompatible applications or versions (no interdependency analysis).
   a. Today, many customers handle this by having version=N and version=N+1 cells.

Reference material:

WebSphere Application Server Administration:

IBM WebSphere Application Server Network Deployment V5.1: **System Administration**
ftp://ftp.software.ibm.com/software/webserver/appserv/library/wasv51nd_admin.pdf
Ch-4: Welcome to Scripting, Ch-5: Deploying and managing using scripting

Article: **System Administration for WebSphere Application Server V5 -- Part 1: Overview of V5 Administration**
http://www.ibm.com/developerworks/websphere/techjournal/0301_williamson/williamson.html
This is a five-part series that describes a variety of ways to use the WebSphere Application Server V5 management features. Part 1 introduces the basic system administration concepts necessary to understand V5 features.

Book: ***IBM WebSphere System Administration***, ISBN-0131446045, Authors: Leigh Williamson, Lavena Chan, Roger Cundiff, (et al.).
IBM WebSphere Application Server Version 5 offers a completely rewritten, improved infrastructure for administering servers and applications. The authors systematically cover all four Application Server administration toolsets: command-line utilities, the new Administrative Console, scripting tools, and Java™ management APIs. There is a complete library of code examples, plus new insider's tips for maximizing your productivity as an Application Server administrator.

Book: ***IBM WebSphere: Deployment and Advanced Configuration***, ISBN-0131468626, Authors: Roland Barcia, Bill Hines, Tom Alcott, Keys Botzum.
This is an advanced guide for delivering applications rapidly, running them smoothly, and administering them efficiently.

WSADMIN Scripting:

WebSphere Application Server Information Center: **Deploying and managing using scripting**
http://publib.boulder.ibm.com/infocenter/wasinfo/index.jsp?topic=/com.ibm.websphere.nd.doc/info/ae/ae/trun_wlm.html

WebSphere Application Server Technical Library: **Sample scripts**
http://www.ibm.com/developerworks/websphere/library/samples/SampleScripts.html

RedBook: **WebSphere Version 5 for z/OS - WSADMIN Primer**
http://www.ibm.com/support/docview.wss?uid=tss1wp100421
This white paper offers an introduction to WSADMIN, and it does it in the form of a "primer." Simple step-by-step exercises are given that will guide you from the simplest WSADMIN tasks up to some fairly complex tasks. Although some parts are z/OS-specific, most of it applies to WSADMIN scripting in general.

WebSphere Application Server Availability and Workload Management:

Article: **Extreme Availability with WebSphere and DB2**
[http://www.ibm.com/websphere/developer/zones/hipods/]
The New York Stock Exchange (NYSE) has extraordinary availability requirements. It relies on the Securities Industries Automation Corporation (SIAC) to ensure those requirements are met. SIAC chose IBM to provide the hardware and software infrastructure for a new trading floor application. This paper describes this unique challenge, the resulting availability partnership and solution that began a successful deployment in April 2004, and enhancements to IBM software that demonstrate IBM's strengths in the realm of extreme availability.

WebSphere Application Server Information Center: **Setting up a multi-node environment**
http://publib.boulder.ibm.com/infocenter/wasinfo/index.jsp?topic=/com.ibm.websphere.nd.doc/info/ae/ae/tins_mutier.html

WebSphere Application Server Information Center: **Balancing workloads with clusters**
http://publib.boulder.ibm.com/infocenter/wasinfo/index.jsp?topic=/com.ibm.websphere.nd.doc/info/ae/ae/trun_wlm.html

High Performance On Demand Solutions: **Library of Best Practices**
http://www-128.ibm.com/developerworks/websphere/zones/hvws/library.html

RedBook: **IBM - Redbook - More about High-Volume Web Sites**
http://www.redbooks.ibm.com/abstracts/SG246562.html
The High-Volume Web Sites (HVWS) team was chartered to work with customers and IBM internal teams involved in the creation and deployment of very large Web sites. The team would learn from this experience and document proven best practices so that customers could learn how to make high-volume Web sites that worked well, and so that IBM developers could improve their products to better support high-volume sites. As it accumulates experience and knowledge, the HVWS team publishes papers aimed at helping you understand and meet the new challenges presented. This IBM Redbook is a compilation of the HVWS papers, which are available individually at the HVWS Web page.

RedBook: **Best Practices for High-Volume Web Sites**
http://www.redbooks.ibm.com/abstracts/sg246562.html
This IBM Redbook is a compilation of papers from IBM's High-Volume Web Sites team. The articles on managing performance and planning for growth are particularly recommended.

RedBook: **WebSphere V5 Performance, Scalability, and High Availability**
http://www.redbooks.ibm.com/abstracts/sg246198.html
This RedBook discusses various options for scaling applications based on Network Deployment V5.0. It explores how a basic WebSphere Application Server configuration can be extended to provide more computing power by better exploiting the power of each machine and by using multiple machines. The book illustrates most of the workload management and scalability features of version 5.0.

RedPaper: **Server Clusters For High Availability in WebSphere Application Server Network Deployment V5**
http://www.ibm.com/support/docview.wss?uid=swg27002473
This paper discusses the various trade-offs involved with, and the options available for, building highly available production e-business solutions utilizing IBM's WebSphere Application Server Network Deployment V5.0 (ND) product.

RedPaper: **Failover and Recovery in WebSphere Application Server Advanced Edition 4.0**
Applies to Version 4.0, Replaced for V5 by the preceding "*Server Clusters for High Availability*" document.

Article: **Maintain continuous availability while updating WebSphere Application Server enterprise applications**
http://www.ibm.com/developerworks/websphere/techjournal/0412_vansickel/0412_vansickel.html
This article describes a method for rolling out a new version of an enterprise application into a production environment where continuous availability of the application is desired. Applications with browser-based clients and Java-based clients are discussed.

Article: **Hardware Configurations for WebSphere Application Server Production Environments**
http://www.ibm.com/developerworks/websphere/library/techarticles/0212_vansickel/0212_vansickel.html
This article describes the most common production hardware configurations, and provides the reasons for choosing each one. It begins with a single machine configuration, and then proceeds with additional configurations that have higher fault tolerance, horizontal scaling, and a separation of Web and enterprise bean servers.

Article: **Implementing a Highly Available Infrastructure for WebSphere Application Server Network Deployment, Version 5 without Clustering**
http://www.ibm.com/developerworks/websphere/library/techarticles/0304_alcott/alcott.html
This article discusses how to implement a highly available infrastructure for the Deployment Manager component of WebSphere Application Server Network Deployment, Version 5.0. Learn how to do this without having to purchase multiple servers and hardware clustering software for each component in the infrastructure.

Advanced Clustering Techniques for Maximizing Web Site Availability with WebSphere Application Server, Version 5
http://www-128.ibm.com/developerworks/websphere/library/techarticles/hvws/advclustering.html
Server clustering is critical to the on-demand operating environment and, in particular, to the Web infrastructure. Server clustering can be used to help achieve continuous availability of Web sites in the always-on global marketplace. This paper discusses advanced techniques for Web and application server clustering using IBM WebSphere Application Server, Version 5. The information comes from our experiences with High-Volume Web Sites (HVWS) customer projects of the last few years.

Article: **Using WebSphere Application Server V5 for load balancing and failover**
http://www.ibm.com/developerworks/ibm/library/i-wasldbal/index.html
This article describes how to set up a cluster that supports load balancing and failover with IBM WebSphere Application Server Version 5 (Application Server). The author gives a brief introduction to clusters, and then describes in detail how to set up a cluster for load balancing. He also describes failover and shows how to test the failover mechanism.

Article: **A technique for monitoring WebSphere Application Server workload management**
http://www.ibm.com/developerworks/websphere/library/techarticles/0403_russell/0403_russell.html
This article describes a technique for using standard Web server directives to log workload management activity in the WebSphere Application Server cluster, and a mechanism for summarizing the behavior of WebSphere Application Server workload management in real time.

Book: ***Maximizing performance and scalability with IBM WebSphere*** ISBN: 1590591305, Author: Adam Neat. This book walks you through proven steps to plan, design, implement, optimize, and manage your platform's performance and scalability. It focuses on proven architectures, optimization approaches, scalability considerations, and best practices.

Ant build tool:

WebSphere Studio Application Developer Information Center: **Working with Ant**
http://publib.boulder.ibm.com/infocenter/wsphelp/index.jsp?topic=/com.ibm.etools.j2ee.doc/html/tjant.htm

WebSphere Studio Application Developer Information Center: **J2EE Ant Tasks**
http://publib.boulder.ibm.com/infocenter/wsphelp/topic/com.ibm.etools.j2ee.doc/html/tanthome.htm

WebSphere Studio Application Developer Information Center: **Eclipse Ant Tasks**
http://publib.boulder.ibm.com/infocenter/wsphelp/topic/org.eclipse.platform.doc.isv/guide/ant_eclipse_tasks.htm

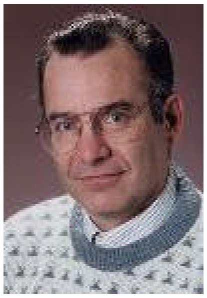
WebSphere Application Server Information Center: **Working with Ant**
http://publib.boulder.ibm.com/infocenter/ws51help/topic/com.ibm.websphere.nd.doc/info/ae/ae/rovr_antapi.html

WebSphere Application Server Information Center: **Ant Tasks**
http://publib.boulder.ibm.com/infocenter/ws51help/index.jsp?topic=/com.ibm.websphere.nd.doc/info/ae/javadoc/ae/com/ibm/websphere/ant/tasks/package-summary.html

Article: **Using Ant with WebSphere Studio Application Developer -- Part 1 of 3**
You can use WebSphere Studio Application Developer to support command-line production build environments in conjunction with common build tools such as Ant. This article explains how to run Ant both inside and outside Application Developer.
http://www.software.ibm.com/wsdd/library/techarticles/0203_searle/searle1.html

Article: **Performing unattended daily builds with WebSphere Studio and Ant -- Part 1 of 3**
The first of three articles on build automation in WebSphere Studio, this article shows you how to do unattended centralized daily builds in a 'clean' environment using Ant and a specific repository.
http://www.ibm.com/developerworks/websphere/library/techarticles/0404_bowker/0404_bowker1.html

About the authors



Barry Searle is the Architect for WebSphere Tools for Automated Build and Deployment. He is a Professional Engineer who has worked at the IBM Canada Lab for over fifteen years on various application development tools. Prior to that he has many years of industry experience developing command and control systems and leading complex communications development projects. You can reach Barry at searle@ca.ibm.com

Ellen Matheson McKay is an Information Developer for IBM Canada Ltd. She writes online help and publications for WebSphere Studio Application Developer. You can reach Ellen at ecmckay@ca.ibm.com

Trademarks

IBM and WebSphere are registered trademarks of International Business Machines Corporation in the United States, other countries, or both.