IBM WebSphere® Application Server V7 vs. JBoss® Application Server V5 TCO Analysis

Authors:
Rick Kotermanski, Jason Armstrong, Matt Holloway (Summa Technologies)
Roman Kharkovski (IBM)

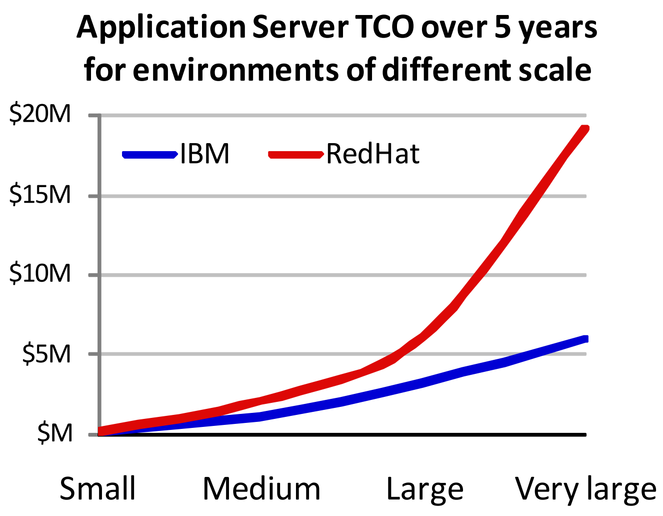
Summa Technologies, Inc.
925 Liberty Avenue
Pittsburgh, PA 15222, USA
(412) 258-3300

May 12, 2009
Document version 2.26

Table of Contents

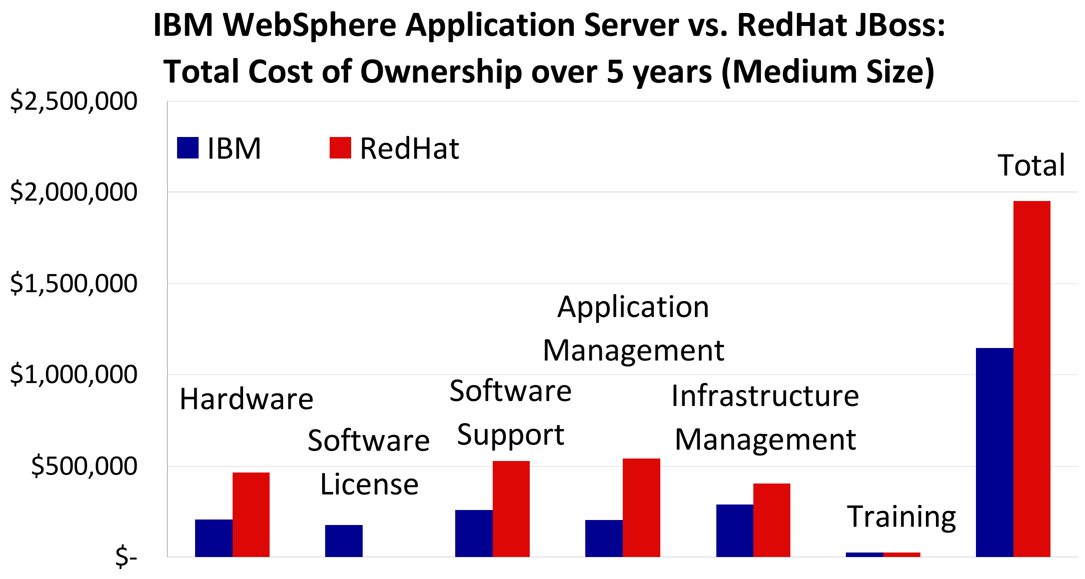
Executive Summary

In recent years, Open Source Software has been receiving significant consideration in the IT industry. As the maturity and reliability of open source application servers increases, more organizations will be looking to introduce them into their environment. While some concerns regarding open source are still being debated, a number of companies are experimenting with and deploying open source application servers in their datacenters. While some are attracted to the ability to view and modify source code as needed, a large majority are attracted to the perceived cost savings of open source. There is no question that for certain types of deployment environments products such as Apache Tomcat, Geronimo, JBoss® Application Server (JBoss or JBoss AS) and IBM WebSphere® Application Server Community Edition (WAS CE) may be cost-effective alternatives, and it is important for IT managers to take advantage of the cost savings they can offer. In many environments, however, commercial application servers have cost advantages over OSS alternatives. As our study demonstrates, using an OSS application server under those circumstances can actually be higher. For our analysis, we divided application server usage into four categories:



- **Small configuration** - these are departmental configurations without high requirements for Quality of Services with single server running one or two web applications. Some of the products that compete in this category are JBoss AS, IBM WebSphere® Application Server Base (WAS) or WAS CE, Tomcat, JOnAS, and GlassFish. We found that in cases when production software support is purchased, IBM WAS Base has up to a double digit percentage cost advantage over JBoss.
- **Medium configuration** – these are departmental configurations with one or more clusters of application servers; some times with dedicated servers for caching, WLM, HTTP serving, requirements for moderate scalability, failover and security among others. We found that JBoss is approximately 70% more expensive than IBM WebSphere® Application Server Network Deployment (WAS ND) in this category.
- **Large configuration** – these are configurations with increased scale and QoS requirements compared to the Medium configuration. These configurations are typical in large datacenters in which a dozen or more applications are deployed on twenty or more servers configured in multiple clusters for redundancy. The cost difference between WAS ND and JBoss in these configurations is approximately the same 70% as in the Medium configuration.
- **Very large configuration** – this is where it gets really interesting. The very large configuration is typical for a large data center. A number of different applications coexist on a shared pool of hardware with strict requirements for high scalability, manageability and monitoring. Spikes in peak workloads are significant and are often hard to predict. Request prioritization and advanced traffic routing must be in place to obey the pre-set SLA levels.

This study focuses on the Medium and Large configuration categories. These categories represent the needs of a large segment of companies requiring the use of an application server. Given recent economic realities and organizations' need to become more cost effective, the debate between the perceived costs of commercial products and OSS offerings has increased. Below are detailed TCO results for Medium configurations:



While JBoss has an advantage with zero initial software acquisition costs, our results indicate that over the course of a five-year period the TCO clearly favors WAS and WAS ND. Additionally, JBoss is missing critical functionality and has significant issues in its current release, as discussed later in the paper. Projecting over five years, our study results demonstrate a total 70% cost advantage in favor of WAS ND:

- **Hardware Costs** - Based on the performance benchmark we performed in the lab, the hardware cost for JBoss V5 is 224% of the cost of IBM WAS ND V7 due to performance differences between the environments. This leads to a larger number of JBoss machines and a more complex environment, which in turn requires more software support subscriptions and increased cost of administration as shown later. Please note that in our previous study the performance difference between WebSphere V6.1 and JBoss V4.2 was only 39% in favor of WebSphere. As of this writing IBM leads SPECjAppServer2004 benchmark in terms of scalability (22,634.13 JOPS) and also shares the top result for performance (299.38 JOPS/core). IBM has been at the top of the list more than a half of the time since the benchmark's inception. JBoss has never submitted a single SPECjAppServer2004 result.

- **License, Subscription, and Support Costs** - While one would expect a significantly lower acquisition cost for JBoss as an open source product, annual subscription costs for JBoss are 206% of IBM WAS ND license and support costs over a five-year period. (Note: Because the JBoss V5 support cost is not announced yet, pricing for the JBoss 4 was used.) This is partly because JBoss requires more hardware and more licenses to run the same workload as WebSphere. Another contributing factor is the fact that IBM bundles LDAP, Cache, WLM servers with WAS ND at no additional charge while JBoss customers must purchase this software separately.

- **Application Administration Costs** - The application specific administration labor cost for support of JBoss is 269% of the cost for IBM WAS ND. This is due to differences required in administrator skill level, the time required to perform similar administrative tasks, and the larger size of the environment that must be managed. The calculations in this study were done using the US based labor costs. In general, initial software investment favors commercial software, when licenses are less expensive than people.

- **Infrastructure Administration Costs** - The infrastructure administration labor cost for JBoss support is 140% of the cost for IBM WAS ND. The installation and management of the core application server is more difficult with JBoss and similar to the application administration described above, requires higher skill level and more effort to maintain.

Changes from the previous report

In 2007, Summa conducted a comparative hands-on analysis of IBM WebSphere Application Server V6.1 and JBoss 4 in a lab environment. In early 2009, Summa revised this report to compare key factors of the TCO for IBM WebSphere Application Server V7 (WAS) and Red Hat® JBoss Application Server V5 (JBoss AS). This newer version of the document includes updates to many of the scenarios executed in the previous version, but also adds new tests for migrating from IBM WAS V6.1 to V7 as well as migrating from JBoss AS V4 to V5. The test application used was also updated to take advantage of new features in Java Enterprise Edition 5.

IBM WAS V7 was released in September 2008. JBoss AS 5 was officially released in December 2008, however, paid support for JBoss AS 5 will not be available until mid-2009. Several key components of JBoss AS 5, including the JBoss Operations Network (JON), are not currently available, and no official release date for this missing functionality is available.

The analysis focuses on both new installations of WAS 7, WAS ND 7, and JBoss 5 as well as migrations from the previous versions of the products. The analysis also included a test of the DayTrader application which is developed as part of the Apache Geronimo project. The DayTrader application was selected because it is well built, implements a number of best practices and showcases many of the technologies used in the JEE5 stack. The Day Trader application is known to run on many different application servers, however, during the course of our testing we found that it did require code changes to run on JBoss 5. More importantly, the J2EE 1.4 version that worked in JBoss AS 4 failed to work as part of the migration to JBoss 5. Conversely, both versions of the application worked without issue on WAS 7.

Why Use TCO in Selecting a Software Vendor?

In its most basic economic sense, a cost is whatever you must give up to get something else, and a return is anything good that you get as a result, whether that return is measured in financial terms or otherwise. IT organizations must frequently launch software implementations and select products based on a limited set of information. The assessment of cost is frequently based exclusively on initial product acquisition costs. In reality, the total cost to a business for developing, deploying and maintaining critical software applications is much higher.

The real total costs to a business over the full lifetime of an application include factors such as requirements analysis and capture, application design and development, deployment, software license costs, operational support, runtime price/performance analysis, hardware and networking costs, software maintenance fees, product upgrades and downtime costs (planned and unplanned). Also, direct labor cost estimates typically fail to account for the actual skill levels required, especially for advanced configuration and administration tasks. Finally, managers frequently neglect estimates for the development of additional non-functional support code needed to maintain the deployment, care and feeding of an environment.

We believe that, just like in the construction business, good tools (and runtimes for that matter) go a long way. Even when the initial acquisition cost is higher, capabilities of more expensive commercial tools can deliver faster and higher ROI compared to “free” open source alternatives. However, each project is unique and the selection of tools must be appropriate. One does not need a heavy duty excavator to dig a hole for a shrub. On the other hand using a pick and shovel will make it difficult to dig a mile long irrigation channel (although human history had this happen more than once).

Cost Factors

The relative impact of TCO factors can vary widely between projects depending on the environment and type of application. Total cost of ownership can be broken down into several factors. This study focuses on the following factors: Software Licensing, Software Support, Application Management, Infrastructure Management, Training, Hardware.

Software development costs are an important portion of total cost of ownership. It was not directly measured in this study due to the high degree of variability across organizations regarding software development practices, team sizes, and skill levels within teams.

This paper focuses on common factors for TCO in enterprise application software development and maintenance. In addition to the quantitative analysis derived from time motion studies, it provides a more detailed analysis of factors where important differences were uncovered through the course of testing.

TCO Area	***Example Costs***
Software acquisition and support costs	▪ Initial license and subscription acquisition costs ▪ Ongoing product support and subscription costs
Hardware	▪ Hardware, hardware support and other infrastructure costs, cooling and power
System administration	▪ Installation and configuration costs, upgrade and migration from previous versions ▪ Deployment, monitoring and routine maintenance of application software ▪ External system interface integration, support and testing costs ▪ Maintenance and version upgrade-related downtime and migration costs ▪ Unplanned downtime and recovery costs ▪ Troubleshooting time for server environment and application issues
Application upgrade	▪ Upgrade related application rework ▪ Troubleshooting time for applications and development environments
Training	▪ Training/education for administrators and developers
Additional TCO considerations	▪ Costs to extend products to support service level agreements (SLAs) ▪ System integration costs for development, deployment of integrated services such as packaged applications (e.g. SAP), messaging products (e.g. WebSphere MQ, TIBCO) and LDAP and database servers ▪ Costs for custom development and testing of non-supported features

TCO Analysis Findings

Details of the hands-on analysis performed and results are contained in following subsections. The following table provides a summary of key findings for each TCO area that were measured in the study.

TCO Category		**IBM**		**RedHat**	**RedHat as % of IBM**
Hardware	$	205,456.39	$	460,409.73	224.1%
Software License	$	174,000.00	$	-	0.0%
Software Support	$	254,715.00	$	524,478.00	205.9%
Application Management	$	200,150.30	$	538,267.35	268.9%
Infrastructure Management	$	288,009.38	$	404,374.26	140.4%
Training	$	22,500.00	$	23,982.00	106.6%
Total	**$**	**1,144,831.07**	**$**	**1,951,511.34**	**170.5%**

The section “Costs and Benefits Not Covered in the Hands-on Study” below contains description of additional considerations that were not included in the scope of this research. If we were to assign approximate conservative cost to those additional factors, the overall TCO advantage of WAS over JBoss increases by additional **37%** over 5 years mainly due to the additional administrative labor costs needed to operate JBoss.

In addition to the quantitative results shown above, several qualitative issues were identified as well. These issues all favor WAS. The following is a partial list of the most critical issues:

- In a cluster environment, planned and unplanned restarts of a single JBoss server required all servers in the cluster to be restarted. In a production environment, this would have resulted in several outages per day. WAS does not require restarts of other cluster members when one of the nodes restarts.
- Memory leaks within the JBoss application server core code required the application servers to be restarted every 12 hours during the performance test. Higher numbers of concurrent clients caused even more frequent crashes. Combined with the clustering issue, this would result in further production outages on a daily basis. WAS does not crash when the number of concurrent clients increase. It simply queues those requests.
- Red Hat does not have explicit statements of support for various technologies, nor does it provide support or troubleshooting tools. In contrast, IBM provides detailed statements of support for various technologies which are, or are not, supported.
- In our previous study we remarked about the lack of JBoss 4 documentation after the product had been available for a couple of years. In this new release JBoss 5 documentation is remarkably sparse even compared to version 4 and has many traces of old data that does not apply to version 5 of the product.
- JBoss 5 administration and configuration must be done at the command line and file level. The scripting administration is very limited and in most cases administrators are required to make direct changes in XML files located in several JBoss directories. Furthermore, these changes must be done individually on all members of a cluster. In contrast, WAS provides GUI and scripting versions of the administrative tool that allows flexible and well documented centralized management of a multi-server environment with a number of sample administrative scripts and help system out of the box.
- We found that JBoss 5 is not backward compatible with the J2EE 1.4 version of the application that we ran on JBoss 4 in our previous study. After several days of troubleshooting we gave up the compatibility test.
- The JEE5 version of our test application that runs on Geronimo, WAS CE, WAS, WAS ND and BEA WebLogic failed to deploy on JBoss 5 due to improper SQL generated by Hibernate JPA and issues with standard JEE5 annotations in the source code.

Hardware Costs

TCO Advantage: IBM WebSphere Application Server

We executed a performance test for each environment using the test application running under a load simulator. The test was executed for a 48-hour period. These tests were intended to identify a performance index that could be used to gauge relative loads that each server could handle. These tests are not a certified benchmark test, and it is important to note that results will vary for different application types. Our performance testing indicated that WAS ND has a 240% advantage over JBoss.

The WAS ND cluster could handle twice as many concurrent no-think-time clients compared to the JBoss environment while average CPU utilization for WebSphere was lower than JBoss. The test on WAS ND executed without incident while the JBoss servers had to be restarted every 12-hours due to out of memory errors, again, with half the number of clients. When the number of concurrent clients for JBoss was raised to the same level as WebSphere, it resulted in the prompt crash of the JBoss servers after only few minutes of the test. It is worth noting that during our performance tests executed in the previous study of JBoss 4 we observed far fewer stability issues.

Based on the results of our testing we found that JBoss would require about 2.41 times as many servers compared to WAS to handle the same load. This difference in performance resulted in a higher overall TCO for hardware of 224% ($205,456 vs. $460,410). These dollar amounts also include cost of dedicated LDAP, WLM and cache servers that we assumed to be the same for both environments.

Software License, Maintenance and Subscription Costs

TCO Advantage: IBM WebSphere Application Server

Comparisons of free OSS to paid commercial software often stop short of considering the reality that paid support for a product is usually both necessary and cost effective for critical applications. Our analysis examines an initial deployment and five-year operating period with paid support subscriptions for comparable environments using list licensing and support subscription models for each environment.

At the time this report was prepared, paid support for JBoss 5 was not available. Additionally, the JBoss Operations Network (JON) is not currently available for JBoss 5, and the freely available version of the administration tool, JOPR was still listed as experimental. Not having a management and monitoring tool is a risk we recommend our customers avoid.

For the purposes of this analysis we assumed that once JBoss 5 paid support is available, pricing will be similar to that of JBoss 4. Additionally, Red Hat provides JON free of charge with paid JBoss support of more than 32 CPUs.

To achieve a leveled comparison between software and support costs, we obtained list JBoss Enterprise Application Platform subscription pricing from the Red Hat [JBOSS 04] and list IBM pricing for WAS ND from the IBM [IBM 02]. There are several noteworthy factors in pricing that will affect TCO calculations and are worth noting when considering the alternatives.

- Both Red Hat and IBM will discount pricing based on volume of business and application environment deployment, affecting individual cases very differently. The analysis performed in the study are based on list prices and do not include any discounts for volume or environment specific cases.
- JBoss pricing requires purchasing capacity in groups of four CPUs and counts CPUs by “sockets,” (i.e. multi-core CPUs count as one socket) per Red Hat policy. This fact will cause JBoss pricing differences to vary on a non-linear basis to WAS and WAS ND based on CPU count. This could benefit JBoss users.
- Red Hat does not have a publicly stated sub-capacity pricing policy in software or hardware based partitioning scenarios (using VMWare or other hypervisors). In certain usage scenarios, WAS or WAS ND will have additional benefit where sub-capacity licensing is applied. Sub-capacity pricing is also not reflected in the results of this study.
- WAS-ND includes WebSphere Edge Components which have been included in our TCO analysis. Configurations that use the WebSphere Edge Components will have additional cost benefits over JBoss AS for functions such as load balancing and edge caching, potentially eliminating the need for other hardware or software products and custom integration effort.
- WAS-ND includes Tivoli® Access Manager, which can act as an LDAP server; JBoss AS does not include a bundled LDAP server so one will be required. The cost of a third-party LDAP server was factored into the analysis for both platforms.

Based on the results of our performance test and the need for additional hardware for JBoss to support the same load, this performance impact was reflected in the software costs because of the net requirement for more CPUs. For our test environment and including capacity factors based on performance comparisons, the net TCO for software licenses, support/subscriptions favors WAS ND by 206% ($428,715 vs. $1,062,745).

Training Costs

TCO Advantage: IBM WebSphere Application Server

At the time of the analysis Red Hat training is not available for JBoss 5. It is expected that when training becomes available, it will be similar in price to previous releases.

The available training from IBM for WAS 7 and Red Hat for JBoss 4 is similarly divided in focus between application development and administration. The TCO analysis includes a base assumption that training for administrators and developers will be provided at the beginning of the five-year analysis period and periodically throughout based on turnover assumptions. The quantity and skill level of administrators is based on the administration TCO costs analysis discussed later in this report. The total TCO component for training over five years is $22,500 for WAS-ND training vs. $23,982 for JBoss AS, a difference of 7%.

Software Development Costs

TCO Advantage: IBM WebSphere Application Server

In our previous study, we noted that differences in developer efficiencies were mostly leveled due to the availability of high quality integrated development environments (IDEs) when developing on JBoss AS 4 and WAS 6.1. However, when comparing WAS 7 to JBoss AS 5 there is a gap between WAS 7 and JBoss AS 5. Over the five years, the development labor component of TCO for support of JBoss AS is estimated to equal approximately a 20% loss of productivity when using JBoss AS over WAS.

Using the same tooling, the process of configuring the test application took five times longer to successfully deploy the application on JBoss. The extra time required to configure the application for use on JBoss was due to inaccurate or missing documentation, as well as the lack of problem support and diagnostic tools.

Troubleshooting issues with JBoss required trial-and-error based on incomplete or incorrect information in the blogosphere and did require code changes due to bugs in JBoss's implementation of the EJB3 specification, and issues with the underlying Hibernate dialect for IBM DB2®. It is important to note, that the time to fix the DB2 dialect issue is not included in the 500% since presumably a different database platform may not have had the same issue.

In contrast, when configuring the application for WAS and WAS ND, no code modifications were required, and issues encountered when configuring the application were readily identifiable using IBM's InfoCenter and problem determination tools.

Through the course of verifying the test application for both JBoss and WAS we also found that significant improvements have been made to WAS and WAS ND from previous versions. No restarts of the application server were required during the application server configuration process, such as configuring JDBC providers and data sources as well as JMS resources. In previous versions of WAS, frequent restarts were required and were quite time consuming.

JBoss required frequent restarts during the configuration process. In some cases, these restarts were to allow configuration changes to take effect. During our testing we also tested the effects of introducing mistakes into the manually edited configuration files. In several of these tests, we had to stop the server, delete temporary directories and redeploy the application prior to starting the server again.

Restarts of JBoss took between 60 and 90 seconds depending on whether the test application was deployed. Restarts of WAS 7 were consistently 30 seconds regardless of whether the test application was deployed or not. Deployment of the application was consistently 10 seconds on WAS while JBoss varied between 15 and 30 seconds per deployment.

In our testing we also tested the effects of common configuration mistakes, such as the misspelling of JNDI names in EJB references. This test on JBoss resulted in a NullPointerException being thrown during the deployment process. The exact cause of the error was not readily identifiable because of the terse nature of the error reported. Using online support forums for JBoss did not help with resolving this issue. If paid support were available, this could require frequent problem support requests and potentially lead to lost productivity while waiting for responses. Conversely, WAS provided an error code which could be used to determine the cause of the issue.

Skilled developers prefer the rapid adoption of new APIs, pluggable customization capabilities and the direct accessibility of the internals of JBoss AS. However, with IBM's adoption of a pluggable architecture and free feature pack releases for WAS customers, developers are now able to take advantage of newer functionality sooner than they could with previous releases of WAS. Furthermore, Red Hat has been the last vendor (out of total 12 vendors listed on Sun's site) to certify for JEE5 compliance and does not commercially support this version of the product. Meanwhile IBM, Oracle and other vendors are shipping production ready supported versions of their JEE5 certified products since 2008.

Application and Infrastructure Administration Costs

This section provides background on the test cases that support the infrastructure and application administration TCO results for the study. The testing and timing of administrative operations included the installation, upgrade, migration, configuration, clustering, securing and a number of other day-to-day maintenance functions for a moderately sized application server. For each test case, an estimated five-year comparative TCO difference, a brief description of the test, and an overview of the findings are provided.

1. **Application Administration Costs -** comparable common administration operations in JBoss could require more highly skilled administrators/developers due to an the lack of an administration console and unclear and inaccurate documentation. As size, complexity and number of staff involved in maintaining an application increase, the relative cost of WAS and WAS ND is further reduced compared to JBoss giving WAS and WAS ND a larger advantage. It is also important to note that the JBoss release tested had limited documentation for configuring clustered environments. **Over the five years, the administration labor cost for support of JBoss is estimated at 269% compared to IBM WAS ND ($200,150 vs. $538,267).**
2. **Infrastructure Administration Costs –** As previously mentioned, due to the lack of administration and migration tools for JBoss, the cost of upgrading to a new version of JBoss could be higher than performing an upgrade and migration from version to version in WAS and WAS ND. Additionally, WAS's and WAS ND's tools for common administrative tasks such as backups, restores, cluster management, and application deployment. make complicated administrative tasks far simpler to perform than JBoss. Because of this, the day-to-day operational support required to keep a JBoss infrastructure running is estimated to be higher than that of WAS and WAS ND. **Over the five-year period, the infrastructure administration costs for JBoss are estimated to be 140% of IBM WAS ND ($288,009 vs. $404,374).**

The overall TCO amount is reduced in the model by factoring in time and cost for common background tasks that are either the same between environments or do not have a significant difference.

Installation and Configuration

TCO Advantage: IBM WebSphere Application Server

This test focused on the initial installation and configuration of the application servers. The test case captured, measured, and compared the initial product acquisition, download, installation and configuration experience for JBoss, WAS and WAS ND. It also covered initial deployment (non-clustered) of the test application. The following list outlines key factors and findings that affected the test case TCO outcome:

- The initial download, single server out-of-box installation and startup of JBoss AS sample applications are simpler, lighter and faster than WAS ND.
- WAS ND requires installation and configuration of a deployment manager component which is then used to manage the federation process. Various wizards and scripting options are available to simplify the task of creating multiple servers. JBoss AS clustering requires the manual editing of XML configuration files and copying those files to each server.
- Compared to WAS and WAS ND, JBoss allows more flexibility in its a la carte approach for installation and configuration of core components and extensions. WAS Feature Packs offer a competitive approach to supporting new technologies over time.
- The initial WAS and WAS ND installations took 20 minutes longer than JBoss AS because of the size of the pre-packaged Java development environment and other pre-packaged content.
- The web server configuration took 40 minutes longer in JBoss than it did for WAS and WAS ND due to issues encountered with the configuration of the Apache plug-in.
- Once both products were installed, next-step configuration operations were much easier to perform in WAS and WAS ND due to the features in the administrative console compared to the XML file-based configuration of JBoss. To administer JBoss, direct editing of XML files and an understanding of JEE5 and

JBoss's internal architecture are necessary. WAS ND administration may be performed completely through a web administrative interface or a command line scripting interface. In addition, because JBoss AS requires direct file editing compared to web-based/constrained configuration settings. Accidental errors in JBoss configuration tended to occur more frequently and took longer to troubleshoot.

Backwards Compatibility

TCO Advantage: IBM WebSphere Application Server

Beyond the server upgrade and its addition to installation and configuration costs, there may be development costs associated with an upgrade. JBoss has not provided the same level of backward compatibility as WAS and WAS ND for applications. IBM provides commitment to an N-2 release support for product features. JBoss does not make a public statement about backwards compatibility in its documentation and in the JBoss 5 release notes indicates several areas where backwards compatibility has been broken. As part of the analysis we upgraded both the WAS and JBoss environments. The results were compelling in that the J2EE 1.4 version of the test application used in the previous analysis (WAS V6.x versus JBoss 4.x) was able to run with no modification or configuration changes in WAS 7 while the exact same application would no longer run in JBoss 5.

Despite significant time spent on this issue, the exact cause of the failure to deploy the J2EE 1.4 based version of the DayTrader application from JBoss V4 into JBoss V5 was not identified. Furthermore, the upgrade path for JBoss was closer to a full reinstallation and reconfiguration of the application server as described in the sections below.

Server Upgrade and Migration Path

TCO Advantage: IBM WebSphere Application Server

Our study results are based upon the assumption that there will be one upgrade of the entire environment performed over the five-year evaluation timeframe to the same application server and our tests included migration scenarios from previous versions of JBoss, WAS, and WAS ND. The amount of time and skills used to perform initial installation and configuration is included in the results of the study.

In our scenario, we upgraded from JBoss 4 to JBoss 5, using JBoss's migration approach of installing and configuring a new server and manually redeploying all applications to the new server. This included the manual migration of data sources and other JEE resources. All of these configuration files had to be tracked and configured manually.

By comparison, the WAS installation media includes a migration tool that will export the settings from a previous version of WebSphere Application Server. For an organization with a small number of application servers this may be a minor issue, but any organization with more than a few servers will find this to be time consuming and potentially error prone. Also, combined with some of the backwards compatibility issues encountered, the upgrade path from JBoss 4 to JBoss 5 may be murkier.

Application Migration

TCO Advantage: IBM WebSphere Application Server

This test focused on upgrading and migrating an application from the previous release to the most current server release. This involved migrating the legacy J2EE 1.4 version of the DayTrader application from WAS and WAS ND 6.1 to version 7, and migrating from JBoss 4 to JBoss 5.

Deployment of the J2EE 1.4 version of DayTrader on WAS and WAS ND used the migration wizard provided with the software installation media. The application worked without modification or reconfiguration of JDBC or JMS resources.

Deployment of the J2EE 1.4 version of DayTrader failed to execute on JBoss 5. JBoss does not provide a migration wizard; this means configuration settings and deployed applications must be manually copied from several XML files in the previous installation to the new XML files. The configuration was validated against the JBoss 4 environment to ensure that the configuration settings and application were the same. In essence, this means deploy and debug cycles must be repeated for every application. For larger installations this could become quite time consuming.

This same concept applies to any new version of JBoss – often within the same major revision number. Since JBoss install does not upgrade previous configuration, for most major upgrades you would need to migrate applications manually as described above.

Cluster Administration

TCO Advantage: IBM WebSphere Application Server

Three test cases related to cluster administration were conducted.

1. Established a baseline of two clustered application servers and then added a third server measuring time and issues with each step. The configuration required load balancing at the HTTP server and secure connectivity between cluster peers. The test was required to use HTTP session replication and validate failover of sessions and connectivity upon server failure.
2. Secure cluster and administration connectivity
3. Evaluate the effort to add a new application server to the cluster.

The following list outlines key factors and findings that affected the test case TCO outcome:

- Application server cluster configurations for both JBoss AS and WAS ND require advanced skills in planning and architecting the clustered environment. This includes understanding the security design and configuring communications among web servers and clustered application servers.
- The initial clustering of JBoss AS servers for a development environment (using default settings) is simpler than installing and configuring WAS ND servers into a cluster. If the JBoss AS clusters are to be partitioned by environment, the tasks are more complicated and require customizations in XML files and installation changes related to its JGroups-based core. Additional detail about securing the cluster is provided below.
- Unsecured cluster node addition is much easier in JBoss AS than WAS ND. The impacts of this are discussed in further detail later in this document.
- Use of HTTP session replication in JBoss AS took a large amount of time to research and implement due to difficulties locating the documentation we found. Unlike WAS ND, JBoss AS does not offer an administrative means to control “buddy” nodes from the set of available servers (i.e. shares HTTP session state for each user). It only allows defining the number of partners to replicate across session, and JBoss AS clustering support selects which nodes will host which applications.
- JBoss 4 used to support farm deployments. However, this functionality has been removed in JBoss 5. This means that configuration steps must be either manually repeated on each cluster node or custom scripted to automate steps for cluster configuration changes. For example, setting up a database or LDAP server connection in WAS ND is done once through the administrative console for the whole cluster. With JBoss AS, the XML files must be updated on each server.
- Deploying a new node to a cluster of servers under security is simple in WAS ND but not possible in JBoss AS. The test to verify intra-cluster communication security could not be completed for JBoss AS because it is not supported by the JGroups. This failure occurred despite working toward the configuration of ENCRYPT and AUTH JGroups extensions for several days with assistance provided directly by JBoss developers through JBoss AS forums.
- Configuring an Apache server as a DMZ proxy server took longer for JBoss AS than WAS ND because of the need to manually download, compile, install, configure and test the Apache mod_jk connector module.

Security Administration

TCO Advantage: IBM WebSphere Application Server

This test case focused on the configuration of various security aspects of the administrative interfaces. The environment was configured to use Active Directory as an LDAP server. The test case required use of LDAP administrative groups and password management to control access to selected administrative functions. For JBoss AS, the JMX console, JBoss AS Console and JON Monitoring access were configured to be LDAP authentication protected. The following list outlines key factors and findings:

- Securing administrative access in JBoss was more difficult and not as thorough compared to WAS and WAS ND. This is due to additional functionality provided by WAS and WAS ND which is not supported in JBoss.
- Administrative security is possible in JBoss but requires a significant understanding of JBoss's inner workings because very little documentation exists for this topic in either JBoss 4 or JBoss 5.
- No cluster configuration security in JBoss AS
  Many security breaches originate from within a company. These breaches range from minor issues all the way through loss of data and complete network failure. Some forms of breaches can also result in the external exposure of private network resources. JBoss does not provide the capability of encrypted communications within the cluster and allows unauthorized instances of servers to be added to the cluster so long as an intruder knows the IP multicast address of the cluster. By comparison, the model employed by WAS ND can be configured to require secure communications between nodes in a cluster and the deployment manager. New nodes cannot be added to the cluster except by the express consent of the administrator when using the federation tools provided with WAS ND. This also means that individuals authorized to add new nodes can be restricted to a known set.
- Default JBoss AS configuration can lead to security vulnerabilities
  While JBoss has eliminated the farm deployment mechanism from JBoss 5, it has failed to address other basic security concerns when operating in a multi-instance environment. The JBoss clustering wiki discusses the need for developers and administrators to take corrective action to prevent instances, including developer workstations, from accidentally forming a cluster. Depending on how certain routing protocols have been configured it may also be possible for these *ad hoc* clusters to span networking subnets. As a general security rule, systems should be configured with the minimal number of harmful settings enabled by default; a conscious act should be required to weaken security. Combining JBoss's promiscuous nature and insufficient security regarding documentation it becomes difficult to fully understand how to configure JBoss to its most secure state.
- No administrative roles for security in JBoss
  WAS and WAS ND provides an administrative security model that allows role-based access control to administrative functions in both the web-based administrative interface and scripting language. JBoss security is exclusively reliant upon operating system-managed, XML file-based security. WAS and WAS ND uses several administrative security roles for performing different administrative tasks (Operator, Administrator, Monitor, Security Auditor, Deployer, Configurator, etc.) and also allows to limit access to resources for different administrators (Peter can manage Application A, but not Application B). This feature enables system administrators to more finely control who can perform different functions within the realm of application server administration. JBoss does not have a similar concept. Providing similar functionality in JBoss would require a significant level of effort to plan, script, and test this configuration approach.
- An experienced JBoss administrator can secure the administrative console in approximately the same amount of time as an inexperienced WebSphere administrator can secure WAS and WAS ND. However, in our test scenarios, an inexperienced JBoss administrator took approximately six times as long to configure LDAP authentication for JBoss. According to forum postings, many deployments of JBoss 4 remove administrative console applications to avoid this issue; this appears to be a good practice in JBoss 5 as well.
- JON is currently not supported on JBoss 5 and therefore could not be configured for LDAP integration.

- WAS and WAS ND offers auditing functions that provide logging and reporting of administrative operations. Because JBoss administration is based upon file edits, a system for capturing changes would have to be developed and tested.

JBoss does not support security for intra-cluster server communications and cross-cluster administration. WAS ND provides secured administrative access to the deployment manager, and communications between cluster peers is secured.

Secure Audit of Administrative Actions

TCO Advantage: IBM WebSphere Application Server

In many organizations, an audit trail of all system configuration changes must be maintained. Because JBoss keeps all of its configuration information in a series of XML configuration files it is easy to make configuration changes outside the view of change tracking software. Changes must be manually recorded or reviewed, or a process would have to be developed to ensure that changes are being tracked.

Conversely, WAS can be configured to log all configuration changes made which can then be shipped to auditors for review. There is a special “Auditor” role in WAS that allows this capability. Users granted this role can view and modify the configuration settings for the security auditing subsystem. For example, a user with the auditor role can enable and disable the security auditing subsystem, set the audit policy, define which security events are to be audited. The auditor has full authority to read and modify properties of the security auditing subsystem. The administrator can view, but can not modify auditing settings.

Scripted Deployment of Application to Cluster

TCO Advantage: IBM WebSphere Application Server

Secured and scripted control of application deployment is important in environments where operations and administrative staff maintain production environments separate from application developers and administrators. This test case focused on the tasks required to script the deployment of an application to a cluster, including resource definitions (JDBC and JMS). The following list outlines key factors and findings that affected the test case TCO outcome:

- All JBoss configuration information is maintained in an array of XML files. Some files may be hot-deployed; others require server restart. WAS ND also maintains server configuration as an array of XML files but exposes access to them through both web-UI and command line-based administrative tools. Some changes require a restart in both application servers. With WAS ND, the web administrative user interface indicates when a server restart is required for each change.
- With JBoss, the administrator must manually distribute cluster-wide configuration changes or develop and test scripts to handle the distribution automatically. With WAS ND, administrative commands may be applied to an individual server, all servers on a node or on a cluster-wide basis.
- WAS and WAS ND support JACL and Jython-based versions of the interactive wsadmin tool. Changes can be very fine grained (e.g., just changing the password for a JDBC data source). With the Command Assistance feature of the IBM WebSphere Admin Console, the appropriate Jython commands can be accessed based on actions executed manually in the Administrative Console. This can be used to generate scripts which can be used for additional environments or deployments.
- Scripting for JBoss is typically done with Ant and shell scripting used to assemble and move groups of files. The manner in which JBoss organizes information dictates granularity of how changes are secured and distributed. The organization will likely not be the same granularity required by change control procedures (e.g. passwords for resource access are stored in plain-text files associated with all other resource configuration information.)

When using JBoss, the implementation of audit trail generation is up to the user’s script development. JBoss AS environment files are typically backed with a source control system such as CVS or Subversion. With

WAS and WAS ND, changes that scripts or web console users perform are logged in the activity log file for the server. Wsadmin scripts can be managed as part of promotion procedures between environments.

Serviceability

TCO Advantage: IBM WebSphere Application Server

Logging, tracing and other diagnostic capabilities of both WAS ND and JBoss AS at times can be challenging, especially in distributed, clustered, and complex environments. JBoss AS does not provide a mechanism to access error information or log files outside of the file system, while WAS ND has log file access, trace analysis tools and error reports for administrative commands to the user upon failure. It was our experience in testing that the error reporting for WAS ND was more consistent, thorough and accessible than JBoss AS errors.

In JBoss 4, the help available in various online support forums and mailing lists were fairly mature. JBoss 5 was rewritten to use the new micro-container architecture. During our testing, JBoss 5's documentation appeared to be a copy of the JBoss 4 documentation and in some cases referred to functionality that no longer exists. When information was found on the JBoss support forums for JBoss 5, extra time was required to handle nuances arising from changes that occurred during the beta and candidate releases.

While attempting to ensure that the test application would deploy in JBoss 5, several days were lost attempting to troubleshoot the exceptions being thrown when deploying the application. In many cases, no answers were available via support forums.

When verifying that the JEE5 version of the application would successfully deploy in WebSphere, several issues were encountered early on, however, IBM's InfoCenter for WAS 7 was able to quickly provide answers. Overall, it took 12 hours to successfully deploy the application on WAS compared to over 100 hours for JBoss.

Server and Cluster Stability

TCO Advantage: IBM WebSphere Application Server

WAS ND executed a non-stop performance test for 48 hours with no degradation of the service, no server restarts and no other unhealthy conditions.

During our stress testing using the same application, JBoss servers had to be restarted once every 12 hours due to out of memory errors. Nodes were manually restarted and never resumed receiving HTTP session replication information, or requests from the web server. When the node was restarted, the other two nodes also began to experience communication issues, which required them to be shutdown as well. Our testing also found, that when this occurred, the only way to bring the cluster up so that all nodes were fully operational was to restart each server serially, starting with the primary node and waiting until each node had completely started before the next node could be started.

The nodes should be independent of each other and starting or stopping a node in the cluster should have no impact on the other nodes within the cluster. Restarting a failed node in the cluster should not require a full restart of the entire cluster. WAS ND supports all of these scenarios without need to restart nodes. In WAS ND nodes do not need to be started in any particular sequence and there is no need to wait until the other nodes are fully started to be able to start new nodes.

Monitoring and Management

TCO Advantage: IBM WebSphere Application Server

The monitoring tool provided with JBoss v5 is a JMX introspection of the server and does not allow real-time management or monitoring of JBoss. At the time of testing a version of JON compatible with JBoss 5 was unavailable and the JOPR open source project JON is based on is listed as experimental. Developers would have to manually implement monitoring from within their source code for applications deployed on JBoss v5. We expect this will change when Red Hat ships a refresh and supported version of JBoss V5 (expected by the end of 2009 or early 2010).

At present, modifying XML configurations by hand, or writing complicated scripts, are the only ways to manage JBoss servers. This process can be tedious and error prone resulting in lost productivity due to mistakes made in files or incorrectly copying files to the wrong location. Auditing changes in JBoss is difficult at best, since changes are managed by directly editing files on the file system.

In comparison, WAS and WAS ND provide Tivoli Performance Viewer for monitoring server performance in real-time. Configuration changes can be made at runtime via the administrative console to adjust how the server is running without requiring a restart.

Systems Integration

TCO Advantage: IBM WebSphere Application Server

Non-trivial web applications are typically required to integrate with one or more external systems, components, or applications through standard, non-standard proprietary and custom interface technologies. Within portions of the test cases that were performed for this study, we found and documented issues on the availability, ease of implementation and other differences that would impact the TCO by affecting administration and support costs for common integration scenarios with directory servers, databases and messaging products.

Within our testing, we found significant differences favoring WAS and WAS ND for ease of integration while establishing connectivity to Microsoft Active Directory as an LDAP server.

In the previous study, we evaluated several other forms of integration on both platforms. However, due to the unexpected issues when migrating the JBoss application we were unable to execute these tests. These tests were:

- Using DB2 as a backing store for JMS.
- The availability of various JCA adapters.

Costs and Benefits Not Covered in the Hands-on Study

Our detailed testing did not cover every aspect of TCO that an organization may encounter in its evaluation of application servers. While our study did perform tests that measured administration effort differences and qualitative issues related to areas such as security, transaction management, serviceability and availability, we did not quantify TCO impact for all of the areas that impact the service level capabilities of the products. The following list highlights selected findings (not a complete list) that may have an impact on some organizations' TCO and deserve additional consideration.

If we were to assign approximate conservative cost to these additional factors, the overall TCO advantage of WAS over JBoss increases by additional **37%** over 5 years.

Return on Investment (ROI)

This study does not include an ROI comparison of WAS and JBoss. An ROI calculation quantifies both the costs and the expected benefits of a specific project over a specific timeframe. TCO, on the other hand, includes just costs. When someone asks about ROI, they are really asking: "What return do I get for my investment?" The potential business benefits (ROI) of implementing the same project with alternative solutions are often neglected and some decision makers have a misconception that tools from different IT vendors are "similar and good enough" to do the job, and as such are equal in capabilities.

While we found significant administration, migration, upgrade and other productivity differences between WAS, WAS ND, and JBoss and measured the costs, we did not measure the potential ROI impact of those differences on business. In the real world, however, it is equally important to measure the time delta between those tasks and calculate potential time to value differences between vendor solutions. For example, if you expect to generate $50,000 of revenue per day from your new application, then going into production with that application 5 days sooner with vendor A than you would with vendor B means additional $250,000 of revenue that you get by using vendor A solution. Similar discussion applies to the topic of high availability. If, for some reason, the above mentioned application deployed on vendor A runtime has 10 less days of downtime in a year compared to vendor B runtime, than you saved yourself $500,000 of revenue plus avoided penalties for SLA violations. Additionally, how do you measure the value of not being paged at night or not seeing your company name in the news?

Transaction Integrity

Transaction integrity guarantees users that the amount of funds taken from one account will not be duplicated on two accounts if the failure happened exactly in the middle of transaction, nor will it appear twice on both accounts. Think of the world's financial exchanges, credit card purchases, or on-line commerce. A failure in these systems could cost a company millions of dollars in lost revenue and lost customer satisfaction. As a result of this risk it is critical that applications handle these transactions correctly and ensure that enterprise data accessed during business transactions is maintained in a reliable and consistent state, regardless of any system or business failure that may occur.

IBM WAS has been rigorously tested in IBM labs and by thousands of customers on the subject of transaction integrity. On the other hands, JBoss V4 did not have the disk based transaction log and kept all transaction data in memory, so failure of the JVM would compromise the transaction integrity and make recovery impossible. RedHat claims that former Arjuna transaction manager has been integrated with JBoss V5. We have not tested this configuration, but since this is the first release of the functionality customers should be careful and investigate this subject in detail before running mission critical transactions on JBoss.

Application Development

Some might argue: "What about the development cost?" Software development is a large and important component of TCO. We did not consider the cost of building new application from scratch as part of this study. The only development cost included is related to the application migration and upgrade.

There are number of developer "camps". On one extreme there are proponents of visually driven development using high powered tools, for example Rational Application Developer from IBM. On another extreme there are proponents of Emacs and "pure" command line development. There is also a middle ground of folks who use Eclipse with few basic plugins. All of them argue that their environment is the most productive. While Rational Application Developer is IBM's preferred tool, WebSphere Application Server can support all three types of these environments – from high powered IDEs through Eclipse all the way to command line and Emacs. JBoss does not offer an advanced IDE, but supports development using Eclipse.

Frequency of Failures

Reliability of software is often one of the most critical selection factors. We did not include the cost of lost business in our calculations as it varies widely between businesses of different type. For example, consider payment processing company that clears transactions between consumers, retailers, banks and credit card processing centers. The cost of a single minute of downtime in this type of environment could cost millions of dollars and could cause significant impact on customer satisfaction and could result in lost future business.

How do you measure the failure rate of the product? How many nines of availability can you expect from WAS or WAS ND and from JBoss?

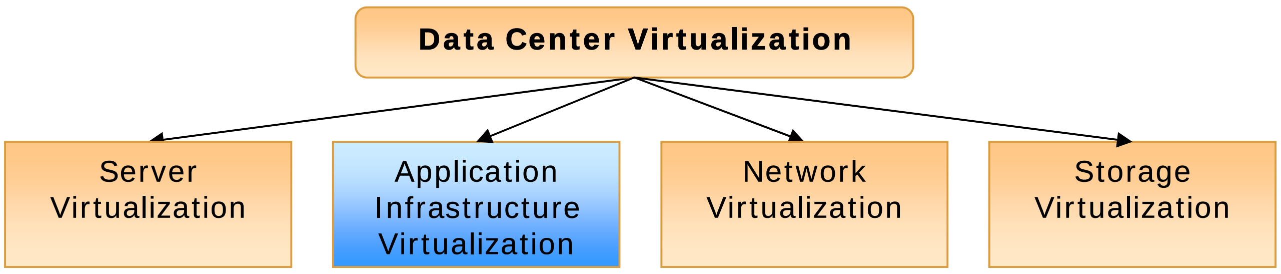
Failover and Speed of Recovery

Sooner or later things break. There are many types of failures: human errors, system overload, application defects, hardware failures, etc. The real question is not really how to avoid the failure, but how to quickly recover from it without significant loss of service. This means that application server must be able to transfer transactions logs, JMS queues and other critical information to backup machines in a cluster or perhaps to a completely different cluster. WAS provides very mature implementation of these capabilities via High Availability Manager and provides very fast data replication engine. For example, typical failover time for all JMS destinations of a server is under 15 seconds when using recommended HA configuration (see link above).

In this study we have not measured the recovery time from different types of failures, but let's consider one example. In the case of a payment processing company one minute of downtime can cost between $1M to $20M or more. There are multiple components that need to be failed over to healthy servers and many types of data that need to be recovered, such as transaction logs, HTTP Sessions, Stateful Session Beans, different kind of singleton services. From all of these, let us consider just JMS part of the failover. WAS can provide configuration with under 15 second failover time (within the same datacenter). RedHat claims that JBoss V5 has brand new JMS implementation as it was one of the JBoss V4 weak points. Given that this is the 1.0 version of the JMS provider, what would be the quality of the code and failover capabilities? Since we did not perform these tests we could assume anything from 60 second automatic failover up to 30 minutes manual failover. This brings us to a significant difference in a quality of service. Depending on the cost of a minute of downtime the financial impact on the business can be very significant.

Application Virtualization and Large Data Center Installations

Application infrastructure virtualization complements server, storage and network virtualization. It is a fourth category of virtualization in the data center (see figure below) that can enable your business to push the boundaries of its IT infrastructure further for greater agility, cost savings, operational efficiency, economy and manageability.


Data Center Virtualization
Server Virtualization
Application Infrastructure Virtualization
Network Virtualization
Storage Virtualization

WebSphere Virtual Enterprise's (WVE) potential impact on TCO was not considered in hands-on tests of this study.

WVE provides application infrastructure virtualization capabilities that lower operational and energy costs to create, run, and manage your enterprise applications and service oriented architecture (SOA) environment. WVE allows data centers to run applications on any application server in a common resource pool. Furthermore, administrators can deploy and utilize resources quickly and seamlessly during peak periods and in response to the peaks and valleys of unforeseen demand in processing for various mission-critical applications. Last, but certainly not least, administrators can achieve application response times and service levels that meet service level agreements. WVE can help extend the benefits of server virtualization using an approach that can address potential issues and limitations. WebSphere Virtual Enterprise can be thought of as a hypervisor for application servers. In addition, it enables server virtualization and application infrastructure virtualization to be combined so that you can take full advantage of the strengths of both approaches.

JBoss does not have comparable functionality for supporting very large environments.

Flexible Management

WAS has a unique capability called Flexible Management that allows you to submit administrative jobs asynchronously for application servers registered to administrative agents and for deployment managers. Jobs can be submitted to one or more servers, including geographically dispersed servers. The administrative job manager can queue administrative jobs directed at the standalone application server nodes or clustered domains. The job manager can asynchronously administer job submissions and can complete the following tasks:

- Set the job submission to take effect or expire at a specified time.
- Specify that the job submission occur at a specified time interval.
- Notify the administrator through e-mail that the job has completed.

JBoss does not have comparable functionality. This functionality could reduce off-hours worked required by administrators or could be used to avoid potentially expensive site visits.

While this study was focused on the Medium and Large scenarios described above, here are several Very Large type scenarios and hypothetical company environments are examples of situations where a job manager is useful. Please note that we did not include this capability into our calculations of the cost, but for some companies this might be a significant factor:

- Branch office environment
  A business has a thousand stores geographically dispersed across the continent. Each store contains either a few application servers, or a small Network Deployment cell consisting of two or three machines. Each store is managed locally for daily operations. However, each store is also connected to the data center at the company headquarters, potentially thousands of miles away. Some connections to the headquarters are at modem speeds. The headquarters uses the job manager to periodically submit administrative jobs for the stores.
- Environment consisting of hundreds of application servers
  An administrator sets up hundreds of low-cost machines running identical clones of an application server. Each application server node, which is registered with an administrative agent, is registered with the job manager. The administrator uses the job manager to aggregate administration commands across all the application servers, for example, to create a new server, or to install or update an application.
- Environment consisting of dozens of deployment manager cells
  An administrator sets up hundreds of application servers, which are divided into thirty different groups. Each group is configured within a cell. The cells are geographically distributed over five regions, consisting of three to seven cells per region. Each cell is used to support one to fifteen member institutions, with a total of 230 institutions supported. Each cell contains approximately thirty applications, each running on a highly-available cluster of two for failover purposes, resulting in a total of 1800 application servers. The administrator uses the job manager to aggregate administration commands across all the cells, for example, to start and stop servers, or to install or update an application.

Per our earlier discussion, managing JBoss in these kinds of environments may become very difficult without building a custom home grown management framework similar to what is discussed above.

Quality and Availability of Vendor Support

Quality of vendor support has direct impact on the cost of operations. IBM provides one of the best software support offerings in the industry with support in local language in many countries in local hours as described in the IBM Software Support Handbook. Prior to the RedHat acquisition of JBoss customers had direct access to the JBoss development team when they needed software support. Tthis model does not scale. RedHat introduced tiers of support similar to other software vendors. Customers should evaluate quality of technical support, its availability in local language and hours of operations. Availability of training classes and onsite emergency support is also important. For example, JBoss V5 training is still not available as of this writing while IBM has well thought out WAS V7 training roadmap with classes delivered across multiple locations worldwide.

Testing and Certifications

Compared to WAS ND, JBoss AS does not provide explicit support statements or certifications around its interoperability with, or support for, other products and versions (e.g., relational databases, messaging systems, packaged applications). IBM provides well-defined, explicit statements of support for interoperability with databases, operating systems, Java standards, web services and other standards. While this is not a guarantee that all functionality will work, it does provide a useful tool for infrastructure planning, and mitigating risks early in a project's lifecycle.

As found within testing, some claims or evidence of support for certain products may be found or inferred from within either official JBoss AS documentation, wiki or forum posts indicating that selected products/versions have worked. However, firm statements to commit or rule out support are not documented.

During our tests, determining level of compatibility and resolving compatibility issues for JBoss AS were very time consuming.

Dual Support Costs

Some organizations may consider supporting both environments to reduce costs by hosting non-critical applications on top of JBoss AS and critical applications on a WAS ND server. However, despite common programming models and JEE capabilities, there could be additional costs to support both environments because of different skill set requirements and processes for server administration that we found in our study. There will also be additional costs associated with managing dual development, test and deployment environments and processes.

Conclusions

Comparisons of WAS ND and JBoss AS frequently focus on an application development point of view alone – only looking at feature sets and initial acquisition costs. Too often free initial cost leads developers to favor JBoss. Unfortunately, some operation managers do not know about this choice until it is too late. Our experience and the analysis from this study reveal that there are many more factors for organizations to consider in their selection process.

Our projections indicate that as the size and complexity of a deployment and the supporting organization grows (even moderately) in breadth, the bulk of costs shifts from product acquisition to administration and other operational activities. In addition to higher skill level requirements for JBoss AS administration there proved to be significant functional shortcomings for its use in a production enterprise application environment – both in its lack of important product capabilities and in its difficulty of administration.

JBoss AS 5 is lacking critical functionality in many areas including security, administration, backwards compatibility, and cluster reliability to name a few. When we examined factors such as system administration, software maintenance, ease of integration and development costs we found that the products take very different approaches that significantly impact TCO. JBoss 5 has a dearth of available support that hinders development and system administration. On the other hand, WAS 7 appears to be much more stable and has continued to build upon the lessons learned from previous versions of WAS.

Paid support for JBoss v5 is not currently available, and in our experience available documentation was unreliable. Red Hat’s JON administration tool is not currently supported for JBoss 5 and no release date has been identified. There is no upgrade path from JBoss 4 to JBoss 5 and in some cases applications that work on JBoss 4 will not work when deployed to JBoss 5.

In the simplest possible terms, JBoss 5, in its current state, is not ready for production use. Customers can deploy WebSphere Application Server V7 today or wait until improved and supported version of JBoss V5 is available to evaluate whether or not it fits their requirements.

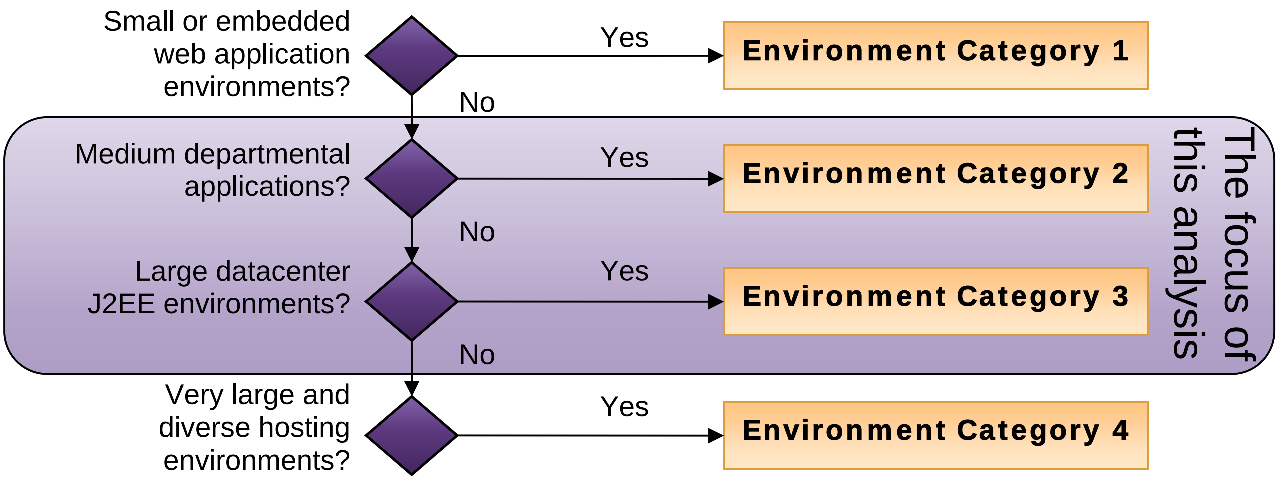
The bottom line is that an open source application server is not free; although it may appear so on the surface. Organizations should carefully examine the full cost and risks based on their staff capabilities, applications and environments.

About this Study

The study was commissioned by IBM and performed by Summa Technologies. All findings of this study are fully backed by the solid reputation of Summa. Since 1996, Summa has been architecting and implementing commercial-grade applications for organizations of all sizes, including Fortune 100 companies. Summa's IT consulting services help companies evaluate and implement modernization strategies for n-tier distributed systems. Summa helps their customers make the right technology investments by first assessing the current environment, and then recommending and implementing the technology solutions that improve customer satisfaction, employee productivity and partner relationships. Authors of this report each have more than 15 years of experience building and supporting large-scale systems.

Appendix A: Analysis Methodology

This section provides an overview of the methodology used to perform the analysis for this report. The TCO analysis and model provided both quantitative and qualitative comparison of JBoss AS and WAS ND based on a combination of test experiences, product research and the execution of a series of planned validation tests. Application server middleware is used to support a broad range of environments and application types. To perform our analysis, we first looked at characteristics of a set of environment classifications and organized the analysis into four categories.


Small or embedded web application environments?
Yes
Environment Category 1
No
Medium departmental applications?
Yes
Environment Category 2
No
Large datacenter J2EE environments?
Yes
Environment Category 3
No
The focus of this analysis
Very large and diverse hosting environments?
Yes
Environment Category 4

Category one - Small environments may host small non-critical applications, or packaged software products that require an embedded web application server. These applications may require less security, lower availability and a frequency of administration that is small compared to other categories. Features such as clustering and session replication for load balancing or fail-over are not required. These applications are typically used in very small organizations.

Category two - Medium is focused on departmental applications in medium and large sized companies. The applications are smaller in size compared to categories three and four, but may require some combination of security, clustering, or more frequent administration.

Category three - Large is focused on the large middle ground of web-application environments. These environments have a range of two to ten servers; many require clustering for reliability, scalability, availability and performance. These environments also demand more comprehensive treatment of security than category one because of application criticality and exposure. Common attributes of applications/deployments in this environment include:

- Hosting of multiple independent and integrated applications;
- A need for moderate to high levels of security for both infrastructure and applications;
- High-availability requirements driving the need for clustering of services;
- Division of responsibilities between developers and administrators for either regulatory compliance (e.g., Sarbanes-Oxley) reasons, efficiency or other organizational reasons;
- Solid change management policies to manage software and server configuration promotion from development to testing and production environments;
- Sample applications in this category include moderately sized web banking and other financial services systems, insurance claims processing systems, and manufacturing process management systems;

Category four – Very Large environments are classified as very large, clustered, and highly available deployments. Example applications in this category may include very high-volume consumer-facing web applications or very large integrated online financial applications. Due to the need to support complex change management monitoring requirements for diverse technologies, these environments benefit from virtualization technologies.

The bulk of the analysis in this paper is focused on factors that impact category two and three environments. This focus was chosen for several reasons. Environment category one is a reasonable target for OSS, yet depending on the level of support purchased from the vendor WAS Base might be more cost efficient than JBoss. Category four is clearly a target for highly scalable commercial software, and therefore the detailed analysis of TCO is not as valuable. The gap between TCO impact and the typical understanding of the full range and magnitude of its factors is most significant in environments in the second and third categories.

The outline below highlights key elements of the analysis approach so that readers may understand the choices made in the selection of TCO criteria and results.

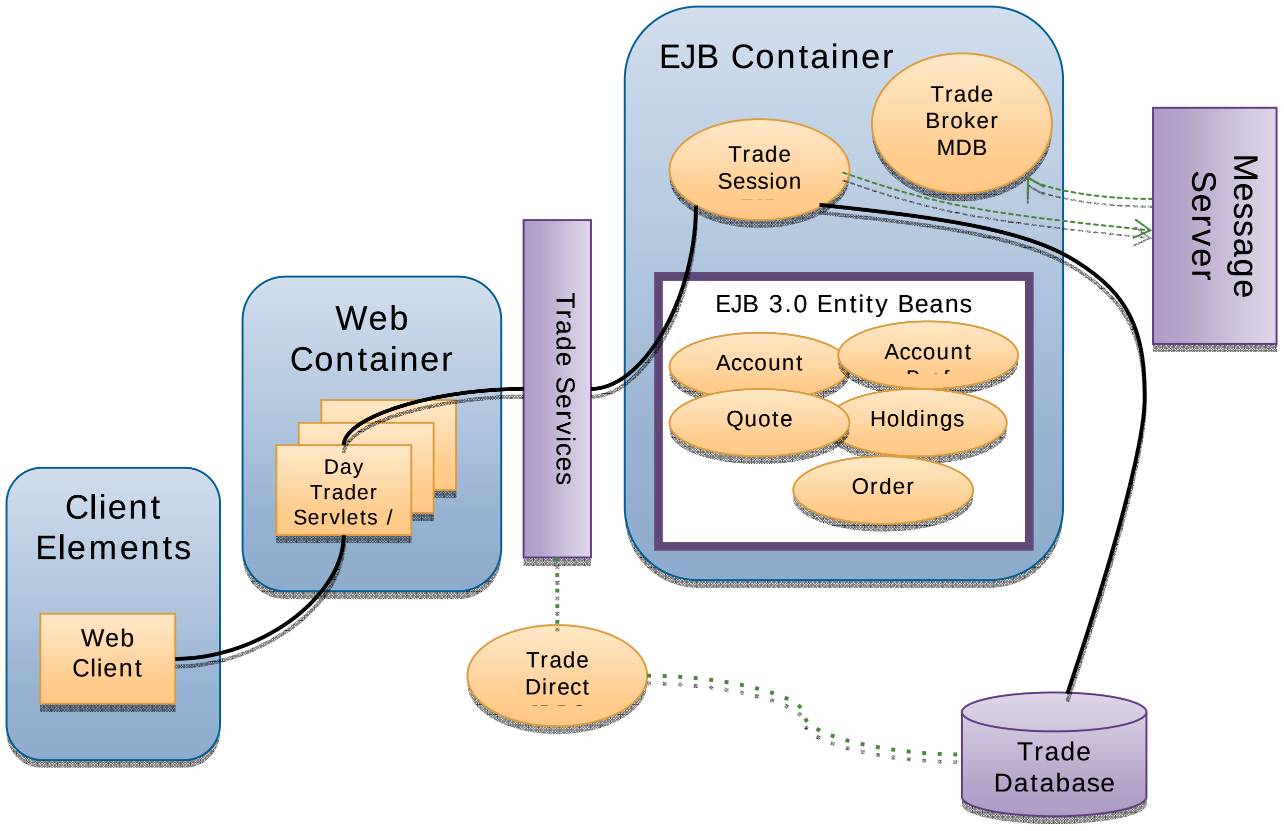
- A list of TCO factors for application server use was established against common architectural, deployment and operational requirements.
- From this input, the four high-level usage categories scenarios were established for comparison purposes. The categories segment analysis across environments that have very different weighting of TCO factors.
- Following the scenario and environment definitions, a set of test cases were developed and reviewed. The test cases defined configuration requirements, goals and scripts for the set of actions to be performed and measured against real-world issues and system configuration requirements for each product. The following list outlines the test case coverage:
    - Installation and Configuration
    - Administrative Security Configuration
    - LDAP Integration
    - Cluster Configuration
    - Clustering Security
    - Extending a Cluster
    - Transaction Logging and XA Configuration
    - Configuring JMS Persistence
    - Automated Deployment of an Application to a Cluster
    - Troubleshooting Clustered Application Issues
    - Application Server Memory Utilization
    - Long Running Stress Test
    - Managing Administration Audit Trail
- The tests were performed by timing operations on both application server environments. Activities included planning, implementation and validation of the execution steps for each test case. Details on the specific operations performed, issues encountered, issue resolution approach and failure cases were timed and documented. We separately measured and recorded time for:
    - Initial time to research and define options and approach
    - Time to perform the initial implementation
    - Time to perform subsequent operations
- In many cases, operations within the same test require different skill levels. For the evaluation, we tracked required skill levels against the Usenix SAGE special interest group defined system administration levels [SAGE 01]**.** Costs for administrators were associated with skill levels using supporting salary survey per skill level.
- The model derives TCO calculations based on the number of times operations are performed over the TCO analysis time. The values are based on environment size factors such as
    - Number of application servers, HTTP servers, LDAP servers, WLM servers, cache servers
    - Number of clusters
    - Number of applications and their complexity and frequency of updates
    - Number of administrators
    - Number of software developers

Appendix B: Test Application

In addition to establishing a representative server environment, a realistic test application was required. To evaluate the TCO, the freely available Apache Geronimo project DayTrader application was used. Reasons that the DayTrader application was selected include:

- Use of a significant set of JEE5 services, including JDBC, JMS, EJBs, Servlets/JSPs, JTS/JTA.
- Enterprise-grade services requirements including security, performance, clustering and transactions.
- Relative ease of configuration for running and testing the application.
- Support of multiple databases and JMS environments.
- Performance in a clustered application server environment.
- Easy-to-use performance load testing interface.
- Migration path from J2EE 1.4 application to JEE5.

The following diagram provides an overview of the DayTrader JEE application architecture.

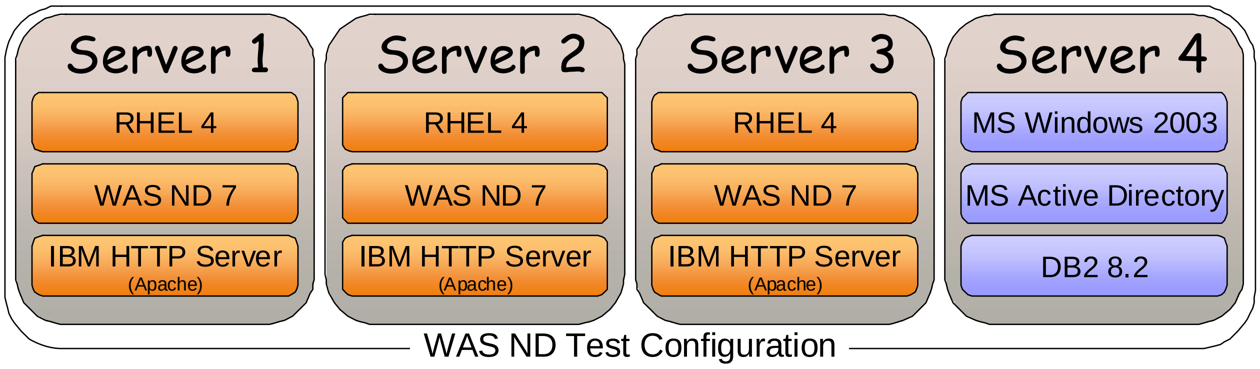


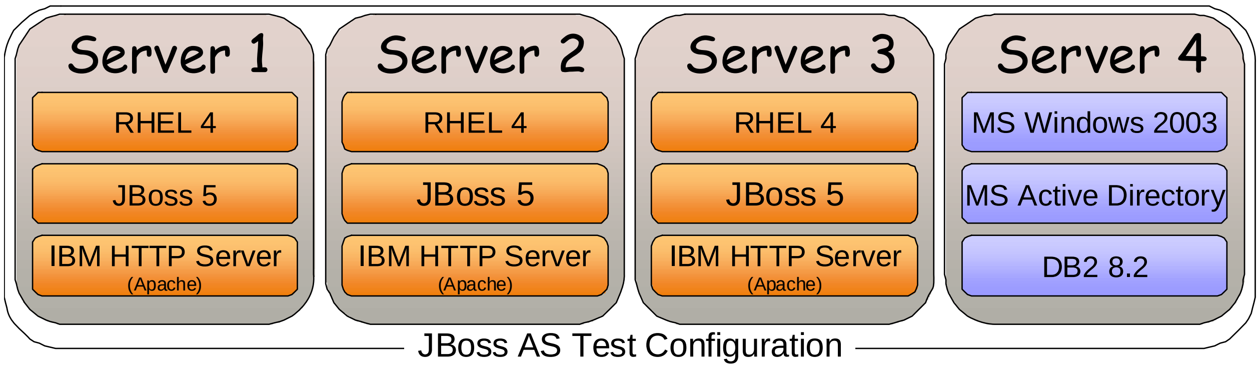
Appendix C: Test Environment

To perform the TCO analysis, a suitably diverse and realistic test environment was established for testing. The components of the test environment were selected based on common combinations encountered in real-world IT operating environments. The technical environments for testing used the same hardware for both environments to ensure that environmental differences were not a factor in TCO measurements. WAS, WAS ND, and JBoss versions were selected based on the most recent production and patch levels generally available as of January 2009. The components of the test environment were selected based on common combinations in real-world IT operating environments. Test environment software products and versions are listed below:

- Virtualization using VMware® Server was used for hosting all test environments. The three Red Hat® Enterprise Linux® environments were hosted on VMWare running on CentOS host servers, the Windows Server® 2003 environment was hosted on a Apple® OS X host server. The use of a set of virtualized VMWare allowed flexibility in configuration, replication, migration, tear down, roll back and retesting of various test cases.

- A common shared database and directory server environment were established on a separate Microsoft Windows server to provide an opportunity to test interoperability of selected interfaces with the application servers. The database and LDAP directory server used:
  - Microsoft Windows Server® 2003 Enterprise
  - Microsoft Active Directory® (for interoperable LDAP directory services tests)
  - IBM DB2 8.2 (application and JMS backing database)
- The three primary test nodes ran Red Hat Enterprise Linux (RHEL) version 4.7 as the base operating system. Each of the test nodes was configured for both WAS and JBoss with only one environment run at a time so that all resources were available to the application server.
- On each of the JBoss application server cluster test nodes, the JBoss environment included:
  - Red Hat JBoss Application Server 5.0 (bundled with Messaging and Transaction support)
  - Sun JDK 1.5
  - **Note**: One server hosted Apache HTTP server for Load balancing the cluster.
- On the each of the WebSphere application server test nodes, the WebSphere environment included
  - IBM WebSphere Application Server 7.0
  - IBM WebSphere Application Server Network Deployment 7.0 (cluster manager) on one server
  - IBM HTTP Server 7.0 on the ND server for Load Balancing
- Additional test and development environment software included:
  - Sun Java 1.5, Ant, Maven and other build tools to build the sample applications
  - IBM Java 1.5, Ant, Maven and other build tools to build the sample applications
  - Eclipse 3.4 (JBoss) and Rational Application Developer 7 (WebSphere)
  - OpenSTA load testing tool (run on the Windows 2003 server node)





Appendix D: High Level Features Comparison

The table below shows a high level summary of features and capabilities, some of which we evaluated and measured in this study. There are many more capabilities provided by WAS that are not found in JBoss. While each feature is "interesting", our study has attempted to put a dollar value and measure business impact of having each of the capabilities listed below. After all, there is no reason to have a "feature" if it does not provide business value to its user.

Capability (\$ - indicates capabilities measured in the study)	***IBM WAS 7***	***JBoss 5***
JEE5 support \$	●●●●❺	●●●●❺
Release is production ready \$	●●●●❺	❶
Support is available from the vendor \$	●●●●❺	❶
Quality up-to-date documentation \$	●●●●❺	●●❸
Administrative GUI \$	●●●●❺	❶
Performance \$	●●●●❺	●❷
Scripted administration \$	●●●●❺	●●❸
Cluster administration \$	●●●●❺	●❷
Administrative security roles separation \$	●●●●❺	❶
Upgrade tools \$	●●●●❺	❶
Problem isolation and determination tools (log analyzer, etc.) \$	●●●❹	❶
Centralized application deployment in a cluster \$	●●●●❺	❶
LDAP support and compatibility \$	●●●●❺	●●❸
Mature JMS provider with fast failover capability and high performance \$	●●●●❺	●❷
DBMS support and certifications \$	●●●●❺	●●●❹
Secure intra-cluster communications \$	●●●●❺	❶
Command assist feature (audit script) \$	●●●●❺	❶
Avoid memory leaks and crashes during the stress load on the server \$	●●●●❺	❶
Hot deploy capabilities \$	●●●❹	●●●❹
Install footprint on HDD \$	●●❸	●●●●❺
Install time on developer machine \$	●●❸	●●●●❺
Install and configuration time on production machine \$	●●●❹	●●❸
Server startup time \$	●●●❹	●●❸
Scalability	●●●●❺	●●❸
Integrated Development Environment	●●●●❺	●●❸
Support for latest WS* standards	●●●●❺	●●❸
XA transaction support	●●●●❺	●●❸
OS support and certifications	●●●●❺	●●❸
Kerberos and SPNEGO support	●●●●❺	●❷
Automatic WAN backup cluster support	●●●●❺	❶
Dynamic page fragment cache (Servlets, JSP, etc.)	●●●●❺	❶
Monitoring tool and performance tuning advisor	●●●❹	❶
Transactional recovery tools	●●●●❺	❶
Flexible management capability (scheduled management of remote servers)	●●●●❺	❶
Eclipse based admin script development tool	●●●●❺	❶
Secure audit of administrative actions	●●●●❺	❶
DMZ hardened proxy	●●●●❺	❶
Ability to manage mixed version environment from one console	●●●❹	❶
Ability to manage and configure the server without access to its file system	●●●●❺	❶
Portlet JSR 286, WSRP 2.0 support	●●●●❺	❶
SIP protocol support	●●●●❺	❶
SCA support	●●●●❺	❶

❶ - not supported, ❷ - weak, ❸ - limited, ❹ - good, ❺ - excellent

Appendix E: References

[APACHE 01] Geronimo DayTrader Example Application
http://cwiki.apache.org/GMOxDOC20/daytrader.html

[IBM 01] IBM WebSphere Application Server Network Deployment InfoCenter
http://publib.boulder.ibm.com/infocenter/wasinfo/v7r0/index.jsp?topic=/com.ibm.websphere.nd.doc/info/welcome_nd.html

[IBM 02] IBM WebSphere Pricing
http://www-306.ibm.com/software/webservers/appserv/was/appserv-was-pricing.html

[IBM 03] WebSphere database requirements
http://publib.boulder.ibm.com/infocenter/wasinfo/v7r0/index.jsp?topic=/com.ibm.websphere.nd.doc/info/ae/ae/rdat_minreq.html

[IBM 04] WebSphere administration training
http://www-304.ibm.com/jct03001c/services/learning/ites.wss/us/en?pageType=course_description&includeNotScheduled=y&courseCode=WA361

[IBM 05] WebSphere development administration training
http://www-304.ibm.com/jct03001c/services/learning/ites.wss/us/en?pageType=course_description&courseCode=WF531

[JBOSS 01] JBoss AS Documentation
http://www.jboss.org/jbossas/docs/

[JBOSS 02] JBoss AS Wiki
http://wiki.jboss.org/

[JBOSS 03] JBoss Operations Network Documentation
https://network.jboss.com/confluence/display/DOC/Index

[JBOSS 04] JBoss Pricing
https://www.redhat.com/wapps/store/catalog.html

[JBOSS 05] JBoss Administration Training
https://www.redhat.com/training/jboss/courses/rh336.html

[JBOSS 06] JBoss Developer Training
https://www.redhat.com/training/jboss/courses/rhd261.html

[SAGE 01] Usenix System Administrators Special Interest Group System Administrator Job Descriptions
http://www.sage.org/field/jobs-descriptions.html

Appendix F: Legal and Trademarks