JBoss vs. WebSphere

JBoss Enterprise Application Platform

A TCO Analysis

WebSphere Application Server

Virtuant™
Although this document was commissioned by Red Hat, the platform staging, testing and collection and analysis of results was completely under the supervision of Virtuant Corp. and its collaborators. Virtuant Corp. maintains all rights therefore to content and ownership of this study.

Virtuant Corp
4200 Loch Highland Pkwy
Roswell, GA  30075

IBM and WebSphere are registered trademarks of IBM.

Linux is a registered trademark of Linus Torvalds. Red Hat, Red Hat Enterprise Linux and the Red Hat "Shadowman" logo are registered trademarks of Red Hat, Inc. in the United States and other countries.

Microsoft and Windows are U.S. registered trademarks of Microsoft Corporation.

UNIX is a registered trademark of The Open Group.

Intel, the Intel logo, Xeon and Itanium are registered trademarks of Intel Corporation or its subsidiaries in the United States and other countries.

All other trademarks referenced herein are the property of their respective owners.

© 2011 by Virtuant Corp. This material may be distributed only subject to the terms and conditions set forth in the Open Publication License, V1.0 or later (the latest version is presently available at http://www.opencontent.org/openpub/).

The information contained herein is subject to change without notice. Virtuant shall not be liable for technical or editorial errors or omissions contained herein.

Distribution of modified versions of this document is prohibited without the explicit permission of Virtuant.

Distribution of this work or derivative of this work in any standard (paper) book form for commercial purposes is prohibited unless prior permission is obtained from Virtuant Corp.
CONTENTS

EXECUTIVE SUMMARY ........................................................................................................ 4
HOW TO READ THIS REPORT .......................................................................................... 5
SUMMARY OF FINDINGS ................................................................................................. 5
SECTION ONE: APPLICATION SERVERS ........................................................................... 6
  Historical Background ..................................................................................................... 6
  IBM WebSphere Application Server ................................................................................. 6
  Red Hat’s JBoss Enterprise Application Platform ............................................................ 7
  Comparing Apples to Apples: What is JBoss Enterprise Middleware? ......................... 7
SECTION TWO: PERFORMANCE ANALYSIS ................................................................. 8
  The Benchmarks .............................................................................................................. 8
  The Messaging Application ............................................................................................ 8
  The Web Transaction Application .................................................................................. 9
  Other Considerations ..................................................................................................... 10
  Virtualization ................................................................................................................ 17
  Development Costs ....................................................................................................... 18
SECTION THREE: TOTAL COST OF OWNERSHIP ....................................................... 19
  Hardware ....................................................................................................................... 19
  License Fees ................................................................................................................ 19
  Personnel Costs ........................................................................................................... 20
  TCO Analysis Results ................................................................................................... 21
  Vendor Support ............................................................................................................ 24
CONCLUSION ................................................................................................................... 25
APPENDIX A. TEST METHODOLOGY .............................................................................. 26
APPENDIX B. TEST ENVIRONMENT ................................................................................ 27
APPENDIX C. REFERENCES ............................................................................................... 28
**EXECUTIVE SUMMARY**

We present this benchmarking report to provide IT decision makers with a fair comparison of the actual Total Cost of Ownership (TCO) of two of the most popular application server products on the market today: IBM® WebSphere® Application Server Network Deployment and Red Hat’s open source offering, JBoss™ Enterprise Application Platform.

Information technology changed forever when companies began offering open source solutions in a formerly license-only marketplace.

Since then, several software companies deliver value by providing services rather than by charging for code distributions. Consequently, commercial vendors, whose businesses rely on license fees, are under pressure to justify their prices.

While “free software” seems compelling, IT managers have long recognized that real cost is more than just license fees. Several recent reports have cautioned decision-makers not to jump to the conclusion that a commercial product will always carry a higher TCO and have tried to prove the case that commercial options are indeed more cost-effective over time. And while TCO analysis is one meaningful way for customers to assess and compare solutions offered through very different business models, critical built-in assumptions can skew their results. In this report, we make our assumptions explicit, focus on a few key factors, and determine, based on straightforward testing, whether there is indeed a clear winner.

In this report, we compare two contemporaneous application server versions: WebSphere Application Server Network Deployment version 7 (WebSphere AS ND) and JBoss Enterprise Application Platform version 5 (EAP). We set up equivalent systems environments, installed the application servers and two common applications (a basic messaging application and a version of the well-known DayTrader application), simulated workload using industry-standard tools, and ran diagnostics. To eliminate knowledge bias, we engaged the services of experienced professionals well versed in the respective application server technologies.

As they consider adopting or migrating to one of these technologies, IT organizations have the good fortune to be able to choose between two strong vendors with impressive track records. IBM is one of the oldest and most recognized information technology vendors, while Red Hat is an 18-year old software vendor recognized as the world’s leading provider of open source solutions to the enterprise. This study emphasizes each offering’s enterprise capabilities as Java™ Platform, Enterprise Edition (Java EE)-compliant application servers.

While detailed findings are presented in the pages that follow, we have concluded that over a three-year period, JBoss EAP would be 50.91 percent less expensive than WebSphere AS ND.
HOW TO READ THIS REPORT

This report summarizes our findings and then presents our analysis in three sections.

- Discusses reasons to use these solutions
- Offers technical comparisons between the two solutions
- Draws conclusions about the TCO based on the detailed results of the technical analysis.

SUMMARY OF FINDINGS

Our analysis yielded the following results:

- As capable, modern Java EE application servers, both solutions can accomplish the objective of providing enterprise system services, a development test bed, serving dynamic web pages, clustering, fail-over, and load balancing. Both solutions offer robust, world-class containers for enterprise Java application deployment. Both solutions have many bundled services that are designed to support most resource-hungry applications.

- Both solutions have compelling features that help to facilitate management and monitoring in medium to large-scale enterprise deployments. In some cases, these are add-on components, especially on WebSphere AS ND, which increase the overall price of the system.

- Both solutions are capable of handling expected volumes of transaction throughput.

- Based on independent analysis (see the Fleming & Perry IDC study in “Development Costs” elsewhere), it has been found that it is approximately one-third quicker to develop with JBoss EAP as opposed to WebSphere AS ND. Since our TCO analysis did not incorporate significant internal application development, the actual JBoss EAP savings may be higher than stated in this analysis.

- As deployments increase, administrative functions tend to become more scripted, rendering graphical user interface (GUI)-based tools less effective. In this respect, JBoss EAP is preferable to WebSphere AS ND because in the JBoss EAP environment, administrators do not need to use a proprietary application programming interface API to accomplish repetitive tasks.

- Both solutions are backed by award winning support organizations. However, almost every new deployment in our testing scenarios required interaction between the WebSphere AS ND experts and IBM support. Whereas on the JBoss EAP side, the same tasks could be completed without contacting Red Hat support.
• We conclude that WebSphere AS ND encourages vendor “lock-in” to a greater extent than does JBoss EAP. Administrative tools in WebSphere AS ND shield the user from industry-standard nomenclature, depending instead on a completely IBM-only lexicon. This is true in almost every aspect of the product suite, and would also tend to raise the level of experience or training required for administrators and developers.

• We conclude from our analysis that over a three-year period, JBoss EAP would be 50.91 percent less expensive than the WebSphere AS ND.

SECTION ONE: APPLICATION SERVERS

Historical Background

Application server technology first evolved in the 1990s as a means of encapsulating services common to many applications, thereby insulating developers from differences among hardware platforms, operating systems, databases, and networks while standardizing runtime services. Services provided by the application server are accessed through standard APIs, an approach that maximizes the developer’s ability to focus on business logic.

Application servers have evolved and matured into comprehensive application platforms in parallel with the spread of web technology and Java. Most enterprises have come to adopt a web-based architecture for the majority of their critical business systems, and application servers now provide many of the runtime services used by these applications, providing many back-end capabilities in addition to a web server. Today multiple products are available and each offer a distinct set of services, though product capabilities can overlap substantially. The platforms discussed in this report are designed to ease the process of developing, deploying, and managing Java EE applications.

Because application requirements vary widely, many IT shops choose to implement only a small percentage of available capabilities. Some vendors deliver services via plug-ins to the basic application server core component; others take a holistic approach, and bundle all services into one monolithic system.

One of the most obvious differences between WebSphere AS ND and JBoss EAP is that JBoss EAP is known to use the plug-in approach, and WebSphere AS ND uses the single-system approach. In addition, the base executable codebase for JBoss EAP 5 is approximately twenty percent of the size of WebSphere AS ND 7. Because the JBoss EAP codebase is smaller to begin with, and because architects can choose the components they want to deploy, JBoss EAP can occupy a much smaller footprint than WebSphere AS ND in production. This smaller size has multiple advantages, including reduction in disk and memory consumption, shorter start-up times, and the ability for developers to host the deployment environment on their local machines for development and testing.

IBM WebSphere Application Server

Originally released as a servlet engine in 1998, IBM WebSphere AS ND is the flagship product within the WebSphere brand. WebSphere AS ND is built using open standards such as Java
EE, XML, and Web Services. It is supported on Windows, AIX, Linux, Solaris, i/OS, and z/OS. Beginning with version 6.1 and now into version 7, those open standard specifications are aligned and common across all the platforms. It works with a number of Web servers including Apache HTTP Server, Microsoft’s IIS, and web servers for several other platforms.

**Red Hat’s JBoss Enterprise Application Platform**

The open source JBoss product was launched in 1999 and performed remarkably well among a collection of Java-based open source middleware contenders for several years. JBoss continued steady growth throughout the early years and was acquired by Red Hat in 2006.

Now commercially in version 5, JBoss EAP continues to improve market share and Red Hat is considered the No. 1 provider of infrastructure software and support in the industry. The core product line has expanded to include a SOA platform, real-time data access services, business process management, and enterprise developer and management tools.

JBoss EAP is 100 percent pure Java. It is supported on any operating system, including Red Hat Enterprise Linux, Windows, and UNIX, as long as a supported JDK is used. (Supported JDKs include Sun JDK, Open JDK, and IBM JDK.) It also supports most of the popular JDBC-compliant databases.

**Comparing Apples to Apples: What is JBoss Enterprise Middleware?**

Evaluators should be careful to distinguish between 1) software from open source community projects, such as JBoss Application Server (JBoss AS), and 2) vendor-managed open source software products such as JBoss EAP.

Open source community projects experience continuous development by a potentially large volunteer developer community—an approach that results in beneficial innovations at the cost of stability and robust integration. To reduce the risks associated with using leading-edge community software in a production environment, Red Hat architects and developers engage in a well-defined process to harvest selected community innovations and integrate them into platforms that have been fully tested and verified to be enterprise ready—a process designed to deliver the best of both worlds for enterprise IT.

JBoss Enterprise Middleware products such as JBoss EAP are offered as part of a subscription arrangement that includes comprehensive Red Hat support, full access to the source code, a predictable update process, and proactive security management.

For more information about this process and the differences between community software and JBoss Enterprise software, see [http://www.jboss.com/products/community-enterprise](http://www.jboss.com/products/community-enterprise).
SECTION TWO: PERFORMANCE ANALYSIS

The Benchmarks
To maximize the practical value of our analysis, we attempt to mimic the real world and analyze
the performance of the products using two common enterprise application scenarios: a
messaging application and a web application. Using Apache’s JMeter¹, we simulated workload
against the two applications.

We selected this approach, rather than selecting a few simple metrics and running a generic
application, because, in our experience, most people are interested in actual measures rather
than academic ones, and application throughput is more meaningful than CPU cycles or SPECmarcs.

Each test ran in a controlled environment inside one or more Amazon EC2 Cloud instances. To
ensure a realistic comparison, we directed the testing teams to run each test on a “pristine”
deployment of the application server, in their “out-of-the-box” configuration. To the extent
possible, each environment was a clone of the other. JVM tuning was not performed, except to
assure conformity. Details of the test environment are listed in Appendix B.

The Messaging Application
For the first application, we set up several Java Messaging Services (JMS) message queues,
each of which has a Message-Driven Bean (MDB) attached to it. When a message is placed on
the queue, the MDB simply consumes the message.

The test client starts by sending a message to six previously configured queues. Each message
gets consumed off of the queue by its respective MDB listener, which records the fact that it
consumed a message and then randomly places the message back into one of the other
queues. As you can imagine, this results in a matrix of increasing messages flowing through
the system. We wanted to know how the JMS subsystem in each platform would respond to this
intensifying flow of traffic.

Our study revealed that for JMS-based messaging delivery and processing, JBoss EAP
consistently performed about ten percent faster than WebSphere AS ND.

¹ Apache JMeter™, http://jmeter.apache.org
The Web Transaction Application

To test a transaction-based web application, the team began with the well-worn DayTrader application suite, a benchmark application built around the paradigm of an online stock trading system. In addition, some modifications were made to this application mainly around simple EJB 3.0 implementations conforming directly to the specifications, plus annotations to inject container resources into the runtime. The net effect was to modernize and simplify the application, and to rely more fully on the container’s native services than the previous versions of the DayTrader application.

In our tests, JBoss EAP demonstrated a repeatable ten percent throughput advantage over WebSphere AS ND.

Figure 1 - WebSphere DayTrader Performance

To execute the performance tests (see Figures 1 and 2), we set up a simple DayTrader servlet that logs in and performs a random trading function, which is a typical transaction scenario defined by the DayTrader application. Whether we ran 1,000 or 10,000 trades through the application, JBoss EAP maintained a ten percent throughput advantage.

The overall similarity between the two sets of results was somewhat surprising, considering the

---

2 *daytrader-a more complex application* [http://cwiki.apache.org/GMOxDOC22/daytrader-a-more-complex-application.html](http://cwiki.apache.org/GMOxDOC22/daytrader-a-more-complex-application.html) (September 2010)
findings of reports published by Red Hat competitors (see the *Summa Technologies* comparison whitepaper dated 2/26/2009). According to our findings, the two solutions deliver comparable performance when executing applications written to the Java EE specification.

*As a result of our tests, we have concluded that the frequently voiced assumption that JBoss EAP requires two to three times more hardware assets is entirely unsupportable. Despite the demonstrated modest JBoss EAP advantage, our ultimate compilation of TCO did not take this difference into account; instead it assumed that required hardware and associated costs were equivalent.*

**Figure 2 - JBoss DayTrader Performance**

**Other Considerations**

*Installation & Configuration*

As we prepared the test environments, we evaluated initial installation and configuration of the application servers. We compared a typical download, installation, and configuration experience for the two solutions, and also compared deployment on clustered and non-clustered environments.
Our findings are as follows:

- Installation and startup of JBoss EAP was clearly simpler and much faster than WebSphere AS ND. Installation of JBoss EAP was consistently an effort of less than fifteen minutes, including download. WebSphere AS ND always took at least two hours to install out of the box.

- Installation and configuration of WebSphere AS ND is very difficult for all but the most experienced administrators. The tools utilize a proprietary nomenclature that requires coaching and training.

- Given its component-based configuration, JBoss EAP is more flexible in installation and configuration of core components and extensions. This flexibility and modularity results in multiple configuration files and as a result does require knowledge of which file to specify for each of the configuration parameters.

- JBoss EAP provides a broad array of configuration and tuning parameters enabled via the command line on server startup with the "-D" option. The number of parameters grows with each release. While the addition of these system parameters is welcome, we found them difficult to discover due to a lack of centralized documentation.

- WebSphere AS ND offers feature packs to simplify the installation process, but as has been the case with WebSphere, they are difficult for new and moderately experienced administrators to manage.

- The total WebSphere AS ND installation process took several days, even with our experienced administrators performing the task.

- To configure JBoss EAP, the administrator must edit XML files located in several nested directories under different branches of the application server. As the product is upgraded, parameters can change in ways that are not compatible with prior releases, and this could prove to be frustrating. Red Hat reports significant improvements in this area in JBoss EAP version 6.

- Most of the WebSphere AS ND administration may be performed completely through a web-based graphical administrative interface. It is clear that the interface is intended to shield the administrator from typical Java EE nomenclature. Mastery of this interface is not a skill transferable to other environments, so it is likely to promote vendor lock-in for IBM.

- Some reports and marketing literature claim that as the number of deployed instances increases, WebSphere AS ND GUI-based administration offers advantages over the lower-level JBoss EAP administration process. Our experience contradicted this claim, demonstrating rather that seasoned WebSphere administrators will typically choose to perform those tasks with proprietary command-line tools for the sake of accuracy and efficiency.
Many initial adopters of application servers will perform most administrative tasks via the available GUI, if it is offered. However, when a shop grows, the need to automate administrative tasks becomes apparent, and many administrators take the time to learn the available deployment tools, and create their own scripts to perform otherwise tedious repetitive tasks.

As an example, imagine 10 instances of an enterprise-level Java application deployed across 12 different instances of WebSphere AS ND. We counted roughly 25 left-clicks for each application update; 120 multiplied by 25 equals 3,000 repetitive mouse clicks! And this is just for managing deployment of a single Java application!

As an alternative, the administrator will migrate naturally to command-line scripting. In WebSphere AS ND, the administrator uses the `wsadmin` tool to carry out instructions via the command-line tool. The first step is to connect to the app server instance to use this tool as such:

```
$ wsadmin.sh -host HOSTNAME -port SOAP_PORT -username USERNAME -password PASSWORD -lang jython
```

Then, to update the ear file:

```
wsadmin> AdminApp.update('ApplicationName', 'app', ['-contents Application.ear -preCompileJSPs -update.ignore.new -operation update -verbose'])
```

This needs to be executed for each deployable file. To save the application, execute the following:

```
wsadmin> AdminApp.list()
```

To restart the application, we must figure out some details about the running process. In order to stop and start the application, we need to set a variable with those details. Issue the following command, passing the cell name, node name and type.

```
wsadmin> appManager = AdminControl.queryNames('cell=perseusNode01Cell,node=perseusNode01,type=ApplicationManager,process=server1,*')
```

And, finally, to stop and restart the server:

```
wsadmin> AdminControl.invoke(appManager, 'stopApplication', 'ApplicationName')
wsadmin> AdminControl.invoke(appManager, 'startApplication','ApplicationName')
```

The `wsadmin` tool is quite powerful and can reach out across the network, if you know the destination name(s). The downside is that you now have an additional rather cryptic language to learn. Even if scripts are written to wrap around `wsadmin` tooling, each administrator that interacts with those scripts will be required to climb the learning curve to understand the API.

To perform similar functionality in JBoss EAP, the administrator uses more familiar tasks:
This difference has significant impact on TCO.

Our analysis concludes that training on WebSphere AS ND, with its subsequent learning curve, will need to be at least twice as frequently scheduled compared with JBoss EAP.

In conclusion, we looked at administration task costs from the standpoint of level of repetitive difficulty. We also assumed that over time, administrators will take advantage of scripting to automate repetitive tasks. Therefore, the use of a graphical administration environment, although convenient when managing a small departmental deployment, becomes less useful in the hands of a highly trained and experienced system administrator operating in a larger environment.

**Large Deployment Issues**

A recent IBM web posting says:

*If you have an existing WebSphere® Application Server Version 5.1.x or Version 6.x WebSphere Application Server, Network Deployment configuration with a significant number of large applications and you must meet a specific maintenance window for migration, you might have some difficulty if you use the standard migration scenario. In this case, you might want to copy the resources in the configuration tree from a Version 5.1.x or Version 6.x deployment manager configuration to a Version 7.0 deployment-manager management profile but defer adding applications to the version 7.0 profile so that you can continue managing the environment using the Version 5.1.x or Version 6.x deployment manager.*

Although the GUI can help with large deployments, at some point it becomes unwieldy. This now appears to be true of all the application management tools on the market.

**Upgrades & Migration Paths**

Typically, there would be at least one upgrade of the entire environment during a typical 3 year period as assumed in the TCO. While this is NOT incorporated in the TCO, we determined the following:

- WebSphere AS ND has an available migration tool that will export the settings from a previous version of WebSphere. We found the upgrade process for WebSphere AS ND to be more demanding than that for JBoss EAP, as the GUI version of the script

insulated us from understanding what was happening under the covers.

- Upgrading JBoss EAP is rather simple, done by installing the application server software from scratch in a new location and reconfiguring and deploying all applications again. If the user is moderately experienced, this process is not daunting, and can be automated in some environments. For most organizations, creation of simple command scripts will accomplish this goal. This is especially true with JBoss EAP 5 and later versions, as the directory structure has been simplified and similar resources are often found in one location.

- JBoss EAP does not provide a migration wizard; however we discovered that similar JDBC pool configuration file modifications had to be performed for most application deployments.

- JBoss ON can be utilized to manage several of these upgrade tasks.

- The process of upgrading a simple application was acceptable on both platforms, except for the task of re-configuring database connection pools, which was equally time-consuming on both solutions.

**Monitoring & Tuning**

JBoss offers a companion product to JBoss EAP called JBoss Operations Network (ON). JBoss ON’s costs are included in our TCO and the solution offers:

- **Provisioning** Definition and deployment of applications (or any content, e.g. JBoss EAP or JBoss Enterprise SOA Platform instances) into remote systems (content can also be updated and rolled back through this system)

- **Search** An enhanced search syntax and saved searches simplify finding resources

- **Alerts** A server-side system for alert notifications

- **Operations** Remote invocation of start-up and other processes

- **Custom Plug-ins** A server-side plug-in framework that allows creation of metrics, monitoring and alerts on a wide variety of subsystems

- **LDAP Support** Association of JBoss ON roles with LDAP groups for use with authorization to reduce the administration effort when setting up users in JBoss ON

- **Apache Web Server** Advanced configuration of Apache servers by leveraging an Augeas lens (Red Hat Enterprise Linux 5 only)

- **Database Support** Support for several back-end data stores

WebSphere AS ND builds on the functionality provided in past versions of the IBM Administrative Console. Administrators configure applications using the wizards built into the administrative console.

In addition, an application monitoring tool, the Tivoli® Composite Application Manager (ITCAM), is available for Application Diagnostics. Its costs are included in our TCO. This tool enables
users to view the health of web applications and servers and drill down to diagnostic information for specific application requests to identify the root cause of problems. ITCAM provides tools to:

- Analyze application performance through trending or historical analysis
- Provide key performance metrics for Transactions to sense and isolate potential problems
- View all Java EE in-flight transactions to uncover the root cause of bottlenecks and perform detailed memory analysis
- Correlate and profile transactions that span multiple subsystems such as IBM CICS and IMS
- Check software consistency to compare key system and JVM metrics on working and non-working systems to help isolate differences that may be causing problems
- Exchange information with IBM Rational Performance Tester so developers understand the performance of applications in test or production

When comparing the use of WebSphere AS ND tools or ITCAM to JBoss EAP with JBoss ON, it is apparent that there is quite a bit of overlap in functionality, yet a massive difference in cost. A premium support subscription to both JBoss EAP and JBoss ON costs $40,000 for 64 cores per year, whereas the IBM alternative is an initial license cost of $619,520 per 64 cores with annual renewal (5%) of $30,976 per 64 cores.

**Clustering and Load Balancing**

While you can take advantage of clustering and load balancing in both environments; each has a different strategy for providing these capabilities.

The architecture of WebSphere AS ND clustering centers on the *Deployment Manager* (DM), which is a process (running in a special WebSphere AS ND instance) responsible for managing the installation and maintenance of applications, connection pools, and other typical Java EE resources. It is also responsible for centralizing user repositories for both application and WebSphere authentication and authorization.

The DM communicates with the attached nodes through another special WebSphere process, the *Node Agent* (NA), which is the process responsible for spawning and killing server processes and also responsible for synchronizing configuration between the DM and clustered nodes.

WebSphere AS ND by itself does not perform load balancing for web requests between servers in a cluster. You need to install IBM HTTP Server (HIS) or WebSphere Edge Components for this purpose and they will determine how the request will be spread. Note that these add-on costs were not included in the TCO calculations nor were these add-ons included in the test.

The built-in clustering capabilities of JBoss EAP use peer-to-peer communications between nodes in a cluster. The administrator can sub-divide the cluster by using unique names for groups of nodes. This is similar to the hierarchical structure of WebSphere AS ND clusters,
servers, and nodes. JBoss EAP also supports the concept of “buddy clustering”, where small groups of nodes are collected into single references.

Load balancing in JBoss EAP is also achieved with additional components implemented in either software or hardware. JBoss EAP supports software load balancing out of the box using Apache’s HTTP server with either the mod_jk or mod_cluster modules at no additional cost. Administrators can also choose to take advantage of hardware-based load balancers if they are available.

**Clustering Tests**

Setting up paired application servers, we tested installation and configuration of clusters in each platform. Along with load balancing via Apache, we insured connectivity between peers, and set up session replication via a simple web application. We then validated failover of sessions and reliability during forced server interruptions.

Conclusions from our study:

- Application server cluster configuration for both platforms requires extensive coordination between diverse groups of personnel, including system administrators, network engineers, and developers.
- Instant clustering is achieved by JBoss EAP via simply starting a particular instance alongside similar instances on the same subnet.
- JBoss EAP clustering is very easy to start via a command-line flag, and auto-discovery of all cluster nodes within the same subnet is enabled by default.
- Administrators find that it is a much simpler task to cluster JBoss EAP servers than to add WebSphere AS ND servers together into a cluster.
- WebSphere AS ND provides some compelling tools for managing clusters in the enterprise, but the graphical administrative component for managing them is useful only in small departmental cases.
- As we found with other administration tasks, setting up clusters in WebSphere AS ND required interaction with IBM support, and took over a day of dedicated work to achieve, even by experienced WebSphere AS ND administrators. In contrast, JBoss EAP was automatically clustered within minutes.
- Unsecured cluster node addition is much easier in JBoss EAP than in WebSphere AS ND.
- For HTTP session replication, JBoss EAP now includes automatic start of “buddy” nodes from the set of available servers (i.e., shares HTTP session state for each user). JBoss EAP will automatically set the buddy configuration for you, and its clustering support selects which nodes will host which applications.
- JBoss EAP now supports both the “push” method for clustered deployments or the “pull” method. Deployments may be “farmed” out (the push model) to other servers from a single node or JBoss EAP’s configuration capabilities allow a central “deployment”
server to be created, from which all servers “pull” their configurations and deployment artifacts upon startup.

- Recent improvements in JBoss EAP version 5 have added even more robust network management protocols.

Our clustering tests on both platforms were eventually inconclusive. Unrepeatable problems were dealt with on both platforms, and failover worked in nearly 100 percent of our tests. Hence we concluded that clustering imposes a negligible TCO impact. Keep in mind that to get a cluster up and running on WebSphere AS ND, it took what we deemed to be an inordinate amount of time as stated above, even for the experienced team hired to shepherd the process. The JBoss EAP in contrast seemed to be naturally prepared to host clustered components out of the box. Note that these time differences were not factored in the TCO calculations as we tried to eliminate any subjective factors in our calculations.

**Stability**

We never encountered stability issues with JBoss EAP or WebSphere AS ND during application and stress testing. Both JBoss EAP and WebSphere AS ND were reliable over the several months that the servers were operational; during stress testing, in the cases where stability issues arose, we were always able to trace them to issues with the test application rather than the application servers themselves.

**Virtualization**

Both IBM and Red Hat offer extended deployment and management tools that are directed at virtual environments. These tools often offer application management features, but are delivered as add-on components, rather than included in the application server codebase. Therefore, they are considered outside the scope of this study.
Development Costs

In a 2010 IDC study of six large companies (totaling over 5,000 IT members), Fleming and Perry concluded:

*By migrating to JBoss, companies in our study changed their strategy for developing applications and as a result built a more efficient process. In addition to saving developer and IT support staffing resources, companies were able to reduce the costs for the infrastructure supporting application development. This savings includes annual costs for development servers and software as well as the infrastructure — space, power, and networking — directly tied to those servers. This savings means that these companies were able to reduce capital expenses while increasing the volume of application development by 51 percent. A company with 10,000 users could experience savings of $400,000.*

One of the drivers for migrating to JBoss EAP, according to that study, was companies’ desire to develop more custom applications in house for competitive advantage. Reduction of development costs prompted several companies in the Fleming and Perry study to expect an increase in the number of custom development projects. On average, companies in this study expected 30 percent growth in application development.

On average, this study found that IT shops gained more than $6 million a year (over $20,000 per 100 users) in financial benefits by moving from traditional proprietary application servers to JBoss EAP. These companies were able to realize the following key benefits:

- Time to develop custom applications was reduced by an average of 16 weeks (39%).
- Developer hours for each application were reduced by 24%.
- Infrastructure costs for supporting application development were reduced by 65%.
- Costs to manage applications following deployment were reduced by 89%.

Because of more efficient application development, companies were able to develop 51% more applications per year with the same staff.

### Application Development

<table>
<thead>
<tr>
<th></th>
<th>Traditional</th>
<th>JBoss</th>
<th>Savings</th>
<th>%</th>
</tr>
</thead>
<tbody>
<tr>
<td>FTEs per application — development</td>
<td>5.41</td>
<td>4.09</td>
<td>1.32</td>
<td>24</td>
</tr>
<tr>
<td>FTEs per application — ongoing support</td>
<td>0.05</td>
<td>0.02</td>
<td>0.03</td>
<td>69</td>
</tr>
<tr>
<td>Average development time per application (weeks)</td>
<td>41.55</td>
<td>25.40</td>
<td>16.15</td>
<td>39</td>
</tr>
<tr>
<td>Number of applications developed annually</td>
<td>21.00</td>
<td>31.67</td>
<td>10.67</td>
<td>51</td>
</tr>
</tbody>
</table>

---

**SECTION THREE: TOTAL COST OF OWNERSHIP**

We consider the TCO generally to include:

- Initial license fees for the application server (per core and/or resource)
- Initial investment or lease startup costs
- The additional costs of add-on software we judge to be essential, such as load balancers and databases
- Monthly lease, maintenance, and support costs
- Renewal license fee plus periodic upgrades
- System administration costs, including staff retraining and new hires
- Development team costs and burden
- Training

Evaluating two solutions that have completely different business models can be confusing. TCO analysis provides a more comprehensive view than simply looking at license and maintenance fees versus subscription fees.

**Hardware**

As we discovered in our performance analyses, previous studies that assumed disparate hardware requirements for JBoss EAP and WebSphere AS ND were fundamentally flawed. While capacity utilization varies, such variations are not significant enough to impact the required number of servers and associated resources. Therefore, for purposes of our study, we will consider hardware costs simply as part of the TCO, and equivalent for the two solutions.

**License Fees**

We have discussed the fundamental differences between commercial and open source licensing scenarios. For our TCO analysis, we consider this a significant component of the result, but not the only influential metric that might compel an organization to adopt open source options.
Personnel Costs

As we have found with other commercial software, salaries are rising for WebSphere AS ND administrators and developers. Salaries for administrators and developers versed in open source solutions, however, are not increasing at quite the same rate.

This differential is typical of other open source software. The simple fact is that more people are available who are familiar with open source software, so there is greater competition for jobs.
The JBoss Community itself provides expertise that is naturally self-supporting, and adds value that the WebSphere AS ND customer cannot access.

We found the average JBoss EAP administrator costs to be approximately 13 percent lower than the costs of equivalent WebSphere AS ND personnel, and we expect this difference to continue. We also expect that, over time, open source personnel will continue to be more versatile, as their knowledge is based on industry-standard specifications rather than vendor-specific terminology and features.

**TCO Analysis Results**

TCO is calculated for a three-year period, with appropriate burden and taxation loads. The calculation breakdown in spreadsheet form is provided later in this section.

- **Overall** In summary, the TCO for WebSphere AS ND was $5.1MM while the TCO for JBoss EAP was $2.5MM. This difference results in a $2.6MM or 50.91 percent advantage to JBoss EAP.

  This JBoss EAP advantage would grow if 1) application development is higher (only 300 development hours were assumed in this analysis), or 2) the IT environment expands, or 3) the wage premiums for WebSphere-knowledgeable staff continue to increase relative to JBoss.

- **Hardware.** We assumed a nine server-class machine configuration with two quad core processors per server. Each server-class machine costs $15,900 for both solutions – a $0 TCO differential.

- **License Fee.** We assumed a $103,040 WebSphere AS ND one-time license fee for each server whereas JBoss EAP has no license fee, for a $927,360 ($103,040 x 9 servers) JBoss EAP advantage.

<table>
<thead>
<tr>
<th>Processor</th>
<th>Per Server</th>
<th>Cores/Processor</th>
<th>PVU’s/Core</th>
<th>$/PVU</th>
<th>Total/Server</th>
</tr>
</thead>
<tbody>
<tr>
<td>Intel Xeon E5640</td>
<td>2</td>
<td>4</td>
<td>70</td>
<td>$184</td>
<td>$103,040</td>
</tr>
</tbody>
</table>

- **Software Support.** For WebSphere AS ND we assumed a 17.5 percent annual maintenance fee and a 5 percent annual renewable support cost resulting in a $208,656 impact in the first year. We assumed JBoss EAP premium support at $40,000 for 64 cores and $11,250 for an additional 16 cores totaling $51,250 in annual fees -- an $157,406 annual JBoss EAP advantage growing to $472,218 over three years, or 17.97 percent of the total JBoss EAP advantage.

- **Software Monitoring & Management** The respective add-on tools in this category facilitate the development, testing, deployment, and monitoring of the application lifecycle. These tools greatly facilitate multiple server management in a dynamic environment. We assumed Tivoli (for WebSphere) at a $696,960 all-in first year cost and
a subscription cost of $34,848 annually thereafter. We also assumed JBoss ON with a $0 acquisition cost with a subscription cost of $0 annually since it is included in the combined JBoss EAP/ON subscription. This JBoss EAP advantage totals $801,504 for the three-year period, providing 30.49 percent of the JBoss EAP savings.

<table>
<thead>
<tr>
<th>Processors/Server</th>
<th>Cores/Processor</th>
<th>$/RVU (Core)</th>
<th>Total/Server</th>
</tr>
</thead>
<tbody>
<tr>
<td>2</td>
<td>4</td>
<td>$9,680</td>
<td>$77,440</td>
</tr>
</tbody>
</table>

- **Application Management Labor** We assumed five developers were required by both solutions, with WebSphere AS ND salaries at $102,000 annually vs. $92,000 for JBoss EAP. We assumed one deployment specialist for both solutions, both earning the same $65,000 salary. This initial $50,000 annual JBoss advantage grows to $150,000 over three years, providing 5.71 percent of the JBoss EAP savings.

- **Infrastructure Management Labor** We assumed two administrators for both solutions, with WebSphere AS ND salaries at $95,000 vs. $78,500 for JBoss EAP. This initial $33,000 annual advantage grows to $99,000 over three years, contributing 3.77 percent of JBoss EAP savings.

- **Training** We assumed two WebSphere AS ND training events annually at a cost of $3,400 each vs. one JBoss EAP event at a cost of $2,800, reflecting the existing level of community support, which is a "trainer" and the relative supply of training resources, which affect pricing. This initial $32,000 annual advantage grows to $96,000 over three years or 3.65 percent of total savings.

- **Application Development** We assumed three projects per year for both solutions, taking 450 WebSphere hours vs. 300 JBoss hours each, reflecting the relative ease of JBoss development (see the Fleming & Perry IDC study in “Development Costs” elsewhere). This initial $27,450 annual advantage grows to $82,350 over three years or 3.13 percent of total savings.

If one analyzes over the two labor differentials ($102,000 WebSphere developer vs. $92,000 JBoss developer and the $95,000 WebSphere administrator versus $78,500 JBoss administrator) one comes up with a $249,000 JBoss advantage, or 9.47 percent of the total $2.5MM JBoss advantage. Over time, we think that labor differentials will account for an even more significant impact on cost savings.

Based on recent trends, we expect that the open source talent pool will grow significantly faster than the pool trained for closed source solutions, so it is reasonable to assume that the labor advantage would grow for JBoss, whereas specialized WebSphere skills will cost relatively more over time.
## Summary TCO Comparison: WebSphere AS ND and JBoss EAP

<table>
<thead>
<tr>
<th>CATEGORY</th>
<th>ITEM</th>
<th>WebSphere AS ND Yr 1</th>
<th>3 Year TCO</th>
<th>JBoss EAP Yr 1</th>
<th>3 Year TCO</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Hardware</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td># Servers</td>
<td>9</td>
<td>9</td>
<td>9</td>
<td>9</td>
</tr>
<tr>
<td></td>
<td>Avg Server Price</td>
<td>$15,900</td>
<td>$15,900</td>
<td>$15,900</td>
<td>$15,900</td>
</tr>
<tr>
<td></td>
<td>TOTAL</td>
<td>$143,100</td>
<td>$143,100</td>
<td>$143,100</td>
<td>$143,100</td>
</tr>
<tr>
<td><strong>License</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td># Servers</td>
<td>9</td>
<td>9</td>
<td>9</td>
<td>9</td>
</tr>
<tr>
<td></td>
<td>Avg License Price/Server</td>
<td>$103,040</td>
<td>$0</td>
<td>$0</td>
<td>$0</td>
</tr>
<tr>
<td></td>
<td>TOTAL</td>
<td>$927,360</td>
<td>$927,360</td>
<td>$0</td>
<td>$0</td>
</tr>
<tr>
<td><strong>Software Support</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>% of License Maintenance</td>
<td>17.5%</td>
<td>$0</td>
<td>$51,250</td>
<td>$51,250</td>
</tr>
<tr>
<td></td>
<td>% of License Subscription</td>
<td>5.0%</td>
<td></td>
<td>$51,250</td>
<td>$51,250</td>
</tr>
<tr>
<td></td>
<td>TOTAL</td>
<td>$208,656</td>
<td>$625,968</td>
<td>$51,250</td>
<td>$153,750</td>
</tr>
<tr>
<td><strong>Software Monitoring &amp; Management (Tivoli &amp; JON)</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>Acquisition</td>
<td>$696,960</td>
<td>$0</td>
<td>$0</td>
<td>$0</td>
</tr>
<tr>
<td></td>
<td>Subscription</td>
<td>$34,848</td>
<td>$0</td>
<td>$0</td>
<td>$0</td>
</tr>
<tr>
<td></td>
<td>TOTAL</td>
<td>$731,808</td>
<td>$801,504</td>
<td>$0</td>
<td>$0</td>
</tr>
<tr>
<td><strong>Application Management</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td># Developers</td>
<td>5</td>
<td>5</td>
<td>5</td>
<td>5</td>
</tr>
<tr>
<td></td>
<td>Avg Developer Rate/Yr</td>
<td>$102,000</td>
<td>$92,000</td>
<td>$102,000</td>
<td>$102,000</td>
</tr>
<tr>
<td></td>
<td># Deployment Specialists</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td></td>
<td>Avg Deploy Specialist Rate/Yr</td>
<td>$65,000</td>
<td>$65,000</td>
<td>$65,000</td>
<td>$65,000</td>
</tr>
<tr>
<td></td>
<td>TOTAL</td>
<td>$575,000</td>
<td>$1,725,000</td>
<td>$525,000</td>
<td>$1,575,000</td>
</tr>
<tr>
<td><strong>Infrastructure Management</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td># Administrators</td>
<td>2</td>
<td>2</td>
<td>2</td>
<td>2</td>
</tr>
<tr>
<td></td>
<td>Avg Admin Rate/Yr</td>
<td>$95,000</td>
<td>$78,500</td>
<td>$95,000</td>
<td>$78,500</td>
</tr>
<tr>
<td></td>
<td>TOTAL</td>
<td>$190,000</td>
<td>$570,000</td>
<td>$157,000</td>
<td>$471,000</td>
</tr>
<tr>
<td><strong>Training</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td># Personnel</td>
<td>8</td>
<td>8</td>
<td>8</td>
<td>8</td>
</tr>
<tr>
<td></td>
<td>Avg Trainings/Yr</td>
<td>2</td>
<td>1</td>
<td>2</td>
<td>1</td>
</tr>
<tr>
<td></td>
<td>Cost/Person/Training</td>
<td>$3,400</td>
<td>$2,800</td>
<td>$3,400</td>
<td>$2,800</td>
</tr>
<tr>
<td></td>
<td>TOTAL</td>
<td>$54,400</td>
<td>$163,200</td>
<td>$22,400</td>
<td>$67,200</td>
</tr>
<tr>
<td><strong>Application Development</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>Avg Developer Rate/Hr</td>
<td>$51</td>
<td>$46</td>
<td>$51</td>
<td>$46</td>
</tr>
<tr>
<td></td>
<td>Avg Projs/Yr</td>
<td>3</td>
<td>3</td>
<td>3</td>
<td>3</td>
</tr>
<tr>
<td></td>
<td>Avg Hours/Proj</td>
<td>450</td>
<td>300</td>
<td>450</td>
<td>300</td>
</tr>
<tr>
<td></td>
<td>TOTAL</td>
<td>$68,850</td>
<td>$205,550</td>
<td>$41,400</td>
<td>$124,200</td>
</tr>
<tr>
<td></td>
<td>GRAND TOTAL</td>
<td>$2,899,174</td>
<td>$5,162,682</td>
<td>$940,150</td>
<td>$2,534,250</td>
</tr>
<tr>
<td></td>
<td>$ Difference</td>
<td>($1,959,024)</td>
<td>($2,628,432)</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>% Difference</td>
<td>-67.57%</td>
<td>-50.91%</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Vendor Support

Some recent comparisons seem to have assumed that vendor support is inherently better in a commercial product than in its open source alternative. Nothing in the data appears to support this. In fact, an argument could be made for the opposite position.

Red Hat’s global support services team was named a top-three finalist in the HDI Team Excellence Awards in March 2011 in recognition of the support leadership Red Hat delivers to its enterprise customers. Sophie Klossner, ED, membership director at HDI, remarked that [Red Hat] “achieved high levels of excellence in the three categories of people, process and technology, making Red Hat a leading-edge company in the support industry.” HDI (formerly the Help Desk Institute) is a global membership, training, and certification association for 50,000 technical support professional members. HDI was founded in 1989 and has 68 chapters.

Recently one study found that when compared on the basis of understanding needs, meeting SLAs, professional services, knowledge transfer, customer loyalty, and support pricing, JBoss customers consistently showed more passion about support from Red Hat than that from other vendors. The study found that:

Objective evidence supporting Red Hat’s industry-leading performance may also be compelling enough to encourage a reexamination of current enterprise middleware investments and significantly impact current vendor evaluation processes.

IBM also has a lengthy history of accolades for its support organization. For years, it has been considered an industry leader for service, customer retention, and excellent products. However, given that its business model is based on license fees with annually recurring support costs, the simple assumption that its support is somehow automatically superior to its open source competitors is simply unsupportable.

During our deployment of test scenarios, the WebSphere team members, who were selected based on their single focus on and experience with WebSphere AS ND, found it quite difficult to deploy our test applications, and required several days of back-and-forth discussions with IBM support in order to get the applications deployed. On the other hand the JBoss team members, who were also selected based on their single focus on and experience with JBoss EAP, never required any contact with Red Hat support, as information was available online in forums, blog posts, and documentation.

Based on this experience, we found that the assumption that an open source contender will offer a reduced footprint of available online support is completely unfounded. In fact, if one considers that the open source vendor’s entire business model is built on support and training revenues, the argument could rationally be made in the other direction.

---

6 JBoss, by Red Hat: Technical Support Services Leader, Velociti Partners, 2010
CONCLUSION

While many factors may influence a company’s decision to adopt a particular IT infrastructure component, Total Cost of Ownership (TCO) is an important and popular metric. Looking beyond the simple costs of initial acquisition or support subscriptions, a TCO analysis attempts to assess what the process of owning a given piece of technology will actually cost the organization over time.

Given the persistence of many half-truths and downright misconceptions about the pros and cons of open source software, we wanted to get to the bottom of a key question: for IT decision-makers evaluating enterprise application servers, does IBM WebSphere Application Server Network Deployment or JBoss Enterprise Application Platform offer a more attractive TCO?

We compared these products as objectively as possible, utilizing people that were expert in each, and concluded that JBoss EAP is the clear winner over WebSphere AS ND with a 50.91% lower TCO, easier application development characteristics, noteworthy operational ease-of-use advantages, and a smaller footprint.
APPENDIX A. TEST METHODOLOGY

At the outset of this study, we were given the task to evaluate the comparative strengths and weaknesses of WebSphere Application Server Network Deployment and JBoss Enterprise Application Platform. With many different possible deployment scenarios from which to choose, we decided to focus on the medium deployment of 2-3 servers, as we felt it achieved our goal of comparing several factors including: installation, administration, deployment, and monitoring of deployed applications in a typical departmental shop.

We compared two contemporaneous application servers: WebSphere AS ND version 7 and JBoss EAP version 5. The test cases were a simple messaging application as well as the DayTrader application, a well-built application that implements a number of Java EE best practices, showcases many of the technologies used in the Java EE stack, and runs on many different application servers.

To eliminate knowledge issues, we fielded several expert teams composed of experienced individuals who perform similar work for VARs of these products. These teams installed the applications, simulated workload, and ran diagnostics. The DayTrader application was chosen in part because we wanted to validate the results in a similar Summa Technologies comparison whitepaper dated 2/26/2009. Our results clearly contradicted the Summa conclusions.

Our main goal was to test assumptions that the commercial contender is somehow inherently better in terms of performance. In other similar studies, this assumption had a major impact on the TCO outcomes reported. We felt that if, in our analysis, we discovered that this assumption was not valid, then the level playing field would reverse the earlier analyst’s TCO findings, which it did.

We also decided to eliminate some TCO factors including: specific administrator or developer work patterns, as experience in individual IT shops will vary. For that reason, our findings discussed earlier - such as differences in installation times and required support, were not included in the TCO analysis. We focused rather on overall use of tools and patterns found in real-world scenarios, and limited our attention to a few key factors that most affected TCO outcomes. The TCO analysis included the Tivoli ITCAM add-on for WebSphere AS ND and JBoss ON for JBoss EAP. However, the test case did not use these tools when validating the application servers.
APPENDIX B. TEST ENVIRONMENT

Hardware

<table>
<thead>
<tr>
<th>OS</th>
<th>Processor</th>
<th># Processors</th>
<th>Memory</th>
<th>Java</th>
</tr>
</thead>
<tbody>
<tr>
<td>Linux</td>
<td>Intel Xeon E5430</td>
<td>1</td>
<td>1.7GB</td>
<td>1.6</td>
</tr>
</tbody>
</table>

Software

<table>
<thead>
<tr>
<th>Name</th>
<th>Version</th>
</tr>
</thead>
<tbody>
<tr>
<td>IBM HTTP Server</td>
<td>7.0.0.15</td>
</tr>
<tr>
<td>IBM WebSphere Application Server Network Deployment</td>
<td>7.0.0.13</td>
</tr>
<tr>
<td>Apache HTTP Server</td>
<td>2.2.15</td>
</tr>
<tr>
<td>JBoss Enterprise Application Platform</td>
<td>5.1</td>
</tr>
</tbody>
</table>
APPENDIX C. REFERENCES

The Competitors

IBM WebSphere AS: www-01.ibm.com/software/webservers/appserv/was/


JBoss Enterprise Application Platform: jboss.com/products/platforms/application/

JBoss Operations Network: jboss.com/products/jbosson/

Pricing

IBM PVU Calculator: www-112.ibm.com/software/howtobuy/pasportadvantage/valueunitcalculator/


Infrastructure

Amazon Web Services EC2 Cloud: aws.amazon.com/ec2/

Testing Components

Geronimo DayTrader Application: cwiki.apache.org/GMOxDOC20/daytrader.html

Apache JMeter: jakarta.apache.org/jmeter/
APPENDIX D. ABOUT VIRTUANT

The people at Virtuant are dedicated to enhancing knowledge and practical application of leading-edge technology. We are a small group of well-worn technologists who have discovered that we can communicate technological subjects in consumable chunks of information. Our customers seem to like the fact that we can shorten the time frame from introduction to the effective use of new concepts, products and ideas.