IBM WebSphere® Application Server V8.5

vs.

JBoss® Enterprise Application Platform

V6

TCO Analysis

Authors:
Larry Finch, Lee Irwin, Steven O'Neill, Hiranmayi Bellumkonda,
Stephen DeVito, James (Ox) Gibson

Prolifics
114 W 47th Street
New York, NY 10036

December 14, 2012

Prolifics
A SemanticSpace Company

Table of Contents

Executive Summary

The JEE Application Server markets continue to expand and adapt to multiple market factors including increased demands for uptime, lower total cost of ownership, ease of administration, enhancements and innovations within the Java runtimes, and stronger competition between commercial and open source vendors. These factors are all playing their part in driving providers like IBM, Red Hat and others to push product advances to the market quickly while also balancing the other demands.

Today, there is a hybrid software market. Open source originally meant software developed and maintained by users for their own benefit. Today, however, many open source products are actually "owned" by corporations who provide support and development services. This paper focuses on the IBM WebSphere Application Server Network Deployment (WAS ND) version 8.5 and the Red Hat JBoss Enterprise Application Platform (JBoss EAP) version 6.

A Prolifics team undertook comparing these Application Server products from IBM and Red Hat. Prolifics has been active in the Java sphere for over 12 years and is the leading specialist in migrating applications between application server environments and upgrading existing applications to newer standards.

Prolifics first compared JBoss EAP 6 and WAS ND 8.5 for feature differences. We found that Red Hat has focused on improving performance and manageability with EAP 6 and has made significant strides in these areas. JBoss EAP version 6 is now faster and easier to manage compared to version 5, however, it has been a considerable change to the JBoss code base and still carries forward some issues of the previous version. IBM, on the other hand, who already had a high-performing and easily manageable product family, has focused on adding powerful new features to its stable and mature code base. A brief comparison:

- WAS ND 8.5 now includes Intelligent Management, a unique feature set that includes Dynamic Clustering, Intelligent Routing, Health Management and Application Edition Management. JBoss EAP does not provide similar capabilities.
- IBM has released the Liberty Profile for WAS, a lightweight framework that installs in seconds and significantly improves developer productivity and provides seamless transition to production WAS.
- WebSphere has always provided role-based administrative security. JBoss now has administrative security enabled by default, but it is still not role-based.
- WAS has comprehensive tooling to support audit of all administrative actions and configuration changes, and to restrict who may set the auditing rules. JBoss has no audit capability. This makes JBoss unsuitable for environments that require regulatory or industry compliance for information security.
- The XA (2 phase commit) transactional integrity bug that was never fixed in EAP 5 still exists in EAP 6, making JBoss unsuitable for any application that must guarantee integrity of simultaneous updates to 2 or more data sources.
- WAS has always supported code consistency across all members of the family, from WAS Express to WAS Base, WAS ND and WAS for z Series. The code base is also consistent across releases, and when migration is necessary IBM provides migration tooling. JBoss is a significant rewrite with EAP 6, meaning that administrators will have to relearn the product and rebuild their shell scripts used to manage JBoss.
- IBM bundles JDK, LDAP, HTTP server, Cache, DB2, WLM servers and application virtualization (dynamic cluster management) with WAS ND at no additional charge while EAP customers must purchase the license or subscription for most of this software separately[1] (unless there is already existing LDAP, WLM and other software in place that can be reused). Note that JBoss EAP6 does

[1] EAP 6 includes some features of the open source Infinispan cache, but its capabilities are limited.

not offer application virtualization.

- WebSphere tops the SPECjEnterprise 2010 performance benchmarks; Red Hat has never submitted a benchmark for JBoss EAP.
- WebSphere outperforms JBoss on Web Service hosting by a factor of almost two to one, and still outperforms JBoss on web applications.
- JBoss EAP 6 is a major rewrite, and has all of the risks associated with a "dot zero" version. As of the date of publication of this paper there are no enterprise applications deployed on EAP 6 that we could find.

Thus, on features, WAS ND is still ahead of JBoss EAP 6. We next address Total Cost of Ownership (TCO). When considering open source, in most cases it is not the ability to modify source code that is attractive to customers, but the perceived cost savings that open source can offer. While the easiest and most obvious costs to calculate are acquisition costs, often overlooked are the far-reaching effects that your choice of application server can have on other costs such as management, administration, training, hardware and opportunity costs due to unexpected downtime.

According to a number of independent industry studies, the cost of software system maintenance in the past two decades grew to be over 90% (see Appendix F). Your choice of application server should not be based on acquisition costs alone, but the Total Cost of Ownership (TCO) associated with a given product. To quote Gartner research note G00165072, March 2009, "Products available for free (such as open source), or those that are a "same cost" swap out, can cost more during three- to five-year period than a first-time commercial purchase…"

A June 2012 Forrester white paper, *The Total Economic Impact To IBM WebSphere Application Server Migrating From An Open Source Environment*, analyzed several scenarios of businesses that migrated a portion of existing development, test and production environments from open source platforms to IBM WebSphere. A federal agency reported that they experienced a 44% savings in ROI during a payback period of 24 months, resulting in a benefit of over $3.9 M in a three-year adjusted ROI calculation. The benefits came from:

- Reduction in support costs
- IT Operational savings from reduced support incidents
- End user operational savings due to higher application availability
- Cross platform IT development team savings

The savings came in spite of additional costs that included license and maintenance fees, migration from the OSS product to WAS, training, implementation labor and administration and support labor.

As another example, a Prolifics client, a mid-sized financial services company, deployed WebSphere Virtual Enterprise (WVE)[2] in 2011 (the same functionality is included in WebSphere ND 8.5's Intelligent Management). At IBM's Impact 2012 conference the client's Lead DBA/Middleware Administrator gave a presentation on the benefits of WVE. After only 6 months of use he reported:

- An immediate displacement of 79 physical servers, generating $348K in hardware savings over 5 years
- A new WAS build time reduced to minutes instead of weeks
- A displacement of 158 virtual Windows instances
- A license cost savings of $301K
- A total predicted TCO savings of $4.4M over 5 years.

[2] WVE is an application server virtualization technology that allows WAS ND clusters to be dynamic, expanding and contracting to meet current workloads. It includes advanced monitoring capabilities, application edition management, autonomic error recovery and Health and Service policy definition and management.

This client had already considered and rejected migrating from WebSphere to JBoss based on a separate internal study.

In the application server market, open source technology has grown in maturity and reliability and is becoming an increasingly viable solution for many companies. With the latest release, JBoss Enterprise Application Platform v6 (EAP 6), Red Hat claims that JBoss is now equivalent to the mainstream Java Enterprise Application Servers. Prolifics set out to determine if the claim is justified by extensively testing JBoss EAP 6 and IBM WebSphere Application Server Base and Network Deployment V8.5 (WAS ND).

This study reports the results of that testing, measuring and quantifying the various costs associated with each product and provides a detailed comparison of their respective Total Costs of Ownership and technology robustness. Furthermore, this document provides a transparent and comprehensive presentation of the extensive testing that Prolifics performed for its analysis. The study is a revisit of a paper published by Summa Technologies in 2010[3] on earlier versions of JBoss EAP and WAS ND. Its purpose is to update the previous analysis with evaluations of WebSphere Application Server v8.5 and JBoss EAP v6.

It is not easy to compare JEE environments given all the factors involved in a complex product like a JEE Application Server, the needs of individual enterprises, and the nature of different applications. Looking at any one dimension in isolation may give a variety of results. We included tests that covered a wide variety of conditions and factors and normalized the comparison using the same hardware, OS and patch levels, and vendor agnostic applications to ensure no vendor had an unfair advantage.

The conclusion reached in this study is that in most environments, while acquisition costs for JBoss EAP are lower, IBM WAS ND provides lower overall TCO due to its advantages in stability, high availability, manageability, documentation and performance. We also identified scenarios where long-standing defects in JBoss that still carry into EAP 6 would absolutely preclude its use when coordination of updates between multiple data sources is required.

In large configurations[4] over a five-year period considered in this study **WAS ND had a 35% total cost of ownership advantage over EAP 6**.

TCO Category	IBM	Red Hat	Red Hat as % of IBM
Hardware	$2,060,934	$3,114,308	151%
Training	$84,375	$171,998	204%
Software License	$2,623,920	-	0%
Software Support	$2,008,815	$1,821,316	91%
Application Management	$759,492	$2,570,500	338%
Infrastructure Management	$1,533,834	$2,301,566	150%
Risk and Downtime	-	$2,268,548	n/a
Development	-	-	0%
Total	**$9,071,370**	**$12,248,235**	**135%**

The key reason for these differences is that the initial acquisition cost of a JEE environment is a relatively minor portion of the overall cost of operating a suite of enterprise Java applications. Less sophisticated products such as JBoss, Glassfish, Apache Tomcat or Geronimo may suffice and be cost effective for undemanding applications, but a full-featured, fully supported JEE environment is critical to success for mission critical applications or suites with dozens or hundreds of applications.

[3] *IBM WebSphere® Application Server V7 vs. JBoss® Enterprise Application Platform V5 TCO Analysis*, Jason Armstrong, Jeff Howell, Alex Wang (Summa Technologies), December 29, 2010

[4] We defined a large configuration as one running 30 enterprise applications, in maintenance mode (no new development), with 18 clustered servers and 6 standalone servers. Details of the configuration available upon request.

This is not to say that WAS is not cost effective for less demanding applications. WAS Base and WAS Express are the same application servers as in WAS ND and are completely code-level interchangeable.

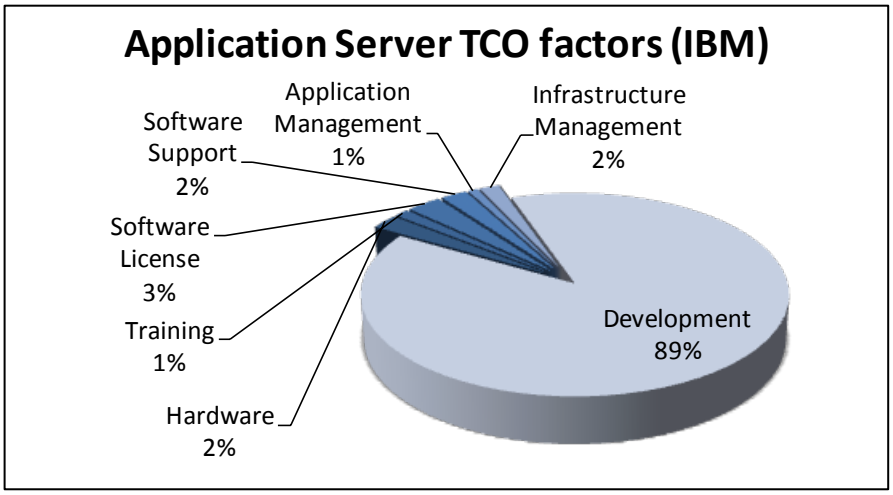
- WAS Express is well suited and cost-effective for small applications that are either stand-alone or require failover between two JVM instances. **The cost comparison for WAS Express showed a TCO advantage of 84% for WebSphere**.
- WAS Base is suitable for medium environments, and is again competitive with JBoss EAP, showing a **TCO advantage of 75% for WebSphere Base over JBoss EAP 6**.

Further, WAS Liberty Profile is a compelling choice for servlet based and Spring framework applications. It installs in seconds and is fully integrated into Eclipse and Rational development tooling. Applications developed on Liberty Profile can be deployed on WAS Express, WAS Base, or WAS ND. Liberty Profile is free when used with any version of WebSphere.

WAS Express, Base and Liberty Profile are also free to developers, so no investment is required to get started with WAS.

Infrastructure Cost relative to total IT expenditure

The comparison above assumed that there was no new or ongoing development, only maintenance of existing applications. This was done to give a fair comparison of WAS vs. JBoss. However, this is an artificial situation; in most IT departments existing applications are updated on a regular basis, and new applications are added periodically. If we update the same model to include active development, assuming 110 developers to maintain and enhance the 30 applications in our model, we see that the total cost of WebSphere licenses and support is only 5% of the IT budget, as shown below:



The equivalent distribution for JBoss is somewhat similar. Thus, for any enterprise that actively updates its applications the investment in IT infrastructure is dominated by development labor costs. It seems clear that choosing the best platform technologically should be the approach of choice as it is the lowest risk alternative. As we will show throughout our analysis, IBM WebSphere is more stable and lower risk than JBoss in almost every category.

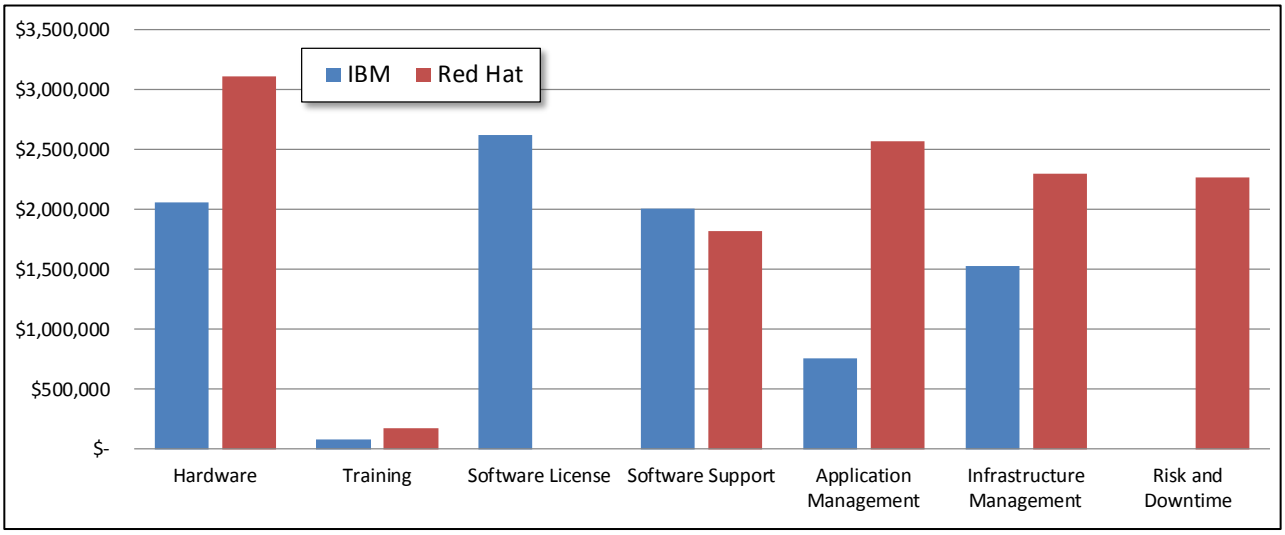
Key Findings

Overview

In 2010 Summa Technologies, Inc. published a paper comparing WAS ND v7 and Red Hat JBoss EAP v5. In preparing this current paper, Prolifics reviewed the 2010 paper and noted differences between the then-current versions of each product and the latest versions. In the summer of 2012 IBM released WAS ND v 8.5, and Red Hat released JBoss EAP 6. Both of these products evolved in the intervening 2 years, both adding features and improvements. On the JBoss side, EAP 6 is a major rewrite. The details will be discussed throughout the paper, but overall EAP 6 is more secure compared to EAP v5, and has an improved administrative interface. However, this has come at the expense of it being an essentially new product with a totally new file system structure, so most of what developers and administrators knew about previous versions must be relearned and configurations, administrative scripts and applications must be migrated. Neither of these is easy.

WAS ND 8.5 is more evolutionary than revolutionary, expanding on the solid base of WAS 8.0. Everything a developer or administrator knows about WebSphere still applies. Where migration is required, IBM provides automated tools for both configuration migration and application migration. While EAP 6 is for the most part an improvement in place, WAS ND 8.5 adds a host of new features, including Intelligent Management support for dynamic clustering, built in advanced health and service level monitoring with automated recovery, hot deploy of multiple application editions, request prioritization and throttling. WAS 8.5 also includes a new, lightweight Liberty Profile that installs and is ready to go in minutes. Liberty Profile and Intelligent Management are the “revolutionary” features in the WAS release 8.5.

JBoss has the obvious appeal from a software acquisition cost perspective but, as with many offerings that have a low barrier to entry, this is still a commercial offering and customers are simply paying in a different way, as covered below. Our results indicate that over the course of a five-year period the TCO clearly favors WAS and WAS ND. JBoss EAP 6 is also missing critical functionality, as discussed later in the paper. Projecting over five years, our study results demonstrate a 35% total cost advantage in favor of WAS ND for large environments (and even bigger advantages for small and medium environments), which can be broken down as follows:



For full disclosure of the recorded steps (Camtasia screen captures) and associated time measurements, server and licensing assumptions, as well as formulas used in this study, please

contact Prolifics or your IBM representative, or see the link to the Prolifics Technology blog [5]. Below is the description of some of the findings of the study:

- **Hardware Costs –** Based on the performance benchmarks we performed in the lab as well as the use of dynamic clustering, the hardware cost for JBoss EAP V6 is 151% of the cost of IBM WAS ND V8.5 This leads to a larger number of cores required to run JBoss and a more complex environment, which in turn requires more software support subscriptions and an increased cost of administration, as shown in the following sections. Moreover, looking at widely accepted industry benchmarks, IBM's 4Q2012 tops the charts on the SPECjEnterprise2010 benchmark, measuring performance for JEE applications in terms of performance (681.39 EjOPS /core). To date, Red Hat has never published results for JBoss to the SPECjEnterprise2010 benchmark nor to the previous industry benchmark, SPECjAppServer2004.

- **License, Subscription, and Support Costs –** An annual subscription must be purchased to obtain JBoss EAP software. Annual subscription costs for JBoss EAP 6 are 39% of IBM WAS ND combined license and support costs over a five-year period.

- **Application Administration Costs –** The application specific administration labor cost for support of JBoss EAP 6 is 338% 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 larger size of the EAP 6 environment that must be managed. While EAP 6 has enhanced its management capabilities with the latest version of JON, our research found that the new functionality still contained quality and reliability issues. Meanwhile, WAS ND continues to have significant advantages in this area. The WAS ND Deployment Manager is more sophisticated and capable than JON, WAS ND 8.5 Intelligent Management adds automated administration capabilities. As a result, overall, initial software investment favors WAS, as its additional license costs are made up for with its substantially lower administrative labor costs.[6]

- **Infrastructure Administration Costs -** The infrastructure administration labor cost for EAP 6 support is 150% of the cost for IBM WAS ND, based on the difference in size of the infrastructures required for each system. This difference in infrastructure size between WAS ND and JBoss EAP 6 is further accentuated by WAS ND 8.5 support for Dynamic Clusters that reduces the overall software and hardware footprint to be managed. The production installation and management of the core application server is more difficult with EAP 6 and, similar to the application administration described above, requires higher skill level and more effort to maintain.

- **Relative Risk and Downtime Costs -** The study found that JBoss does not provide capabilities such as reliable XA transaction recovery, secure role separation between administrators, auditing capabilities, and other important enterprise qualities of services. The lack of these features introduces risks and potential downtime. The impact of risk and downtime can vary significantly based on the type of the business and application. In our study we made the assumption that the annual business value of the project is $30 M over 5 years. The approximate cost of additional risks and downtime for JBoss was conservatively calculated to be roughly $2.3 M higher than for WebSphere.

In addition to quantifying the total cost of ownership, we also evaluated several qualitative factors that influence the overall user experience for WAS and JBoss. These factors may have an indirect impact on the TCO for an organization and should be considered when evaluating either product. Significant findings are:

[5] http://expert-tech.blogspot.com
[6] The calculations in this study were done using US based labor costs.

- **Performance** – Our performance tests indicate a performance advantage favoring WAS over JBoss by a factor of between 1.1:1 and 1.9:1, with one scenario of 11:1. This difference in performance will increase hardware, software, and administrative costs for JBoss when attempting to achieve comparable performance results. We have conservatively used a performance factor of 129% (factor 100% would mean identical performance). We acknowledge that performance can be a factor of the application or the environment, so a common platform was used and a common vendor independent application widely used for this type of comparison was the benchmark. Still, individual application performance can vary widely.
- **WAS 8.5 Intelligent Management** – This new feature of WebSphere supports Dynamic Clusters, a capability that can improve the CPU utilization of servers running the application servers to 85% or higher. We applied a conservative 20% utilization advantage to hardware and license costs for WAS ND.
- **Documentation** – The EAP 6 documentation is geared primarily towards developers. It contains many errors and omissions and in many cases requires administrators to identify the correct settings by trial-and-error. This can be a costly process, especially if an issue is not encountered until after a system enters production.
- **Failover Support and Transactional Integrity** – JBoss has improved in areas of failover support such as HTTP session replication and JMS recovery; however, testing has shown that there are still transactional integrity issues with JBoss that were present in EAP 5, and that these are actually more serious in EAP6 than they were in EAP5, as under some conditions failures can go undetected.
- **Out-of-the-box Security** – JBoss EAP 6 has its management interfaces secured out-of-the-box. A user must be added to a local file-based security realm using a product provided script. The permissions on this realm must be set manually to 600 as a best practice. Further, the auto-deployments folder is active by default to speed time to value for developers. This should be turned off prior to going to a non-test environment. Also new, an authenticated user, representing a cluster member, is required to join a cluster. This resolves the issue reported in the previous version of this report on the security risks of the JBoss automatic clustering. We still observed that there is a security issue in that certain situations require direct editing of configuration files, which requires administrators to have login access to the JBoss directory structure and also permits changes to configurations that are not logged for audit purposes.
- **Role Based Access Control** – JBoss EAP 6 still does not have RBAC for administrative users, a potential security and compliance issue in a regulated environment.
- **Compatibility Issues Between Admin Tools** – The compatibility issues from the previous version of EAP seem to have been resolved. There is now a single configuration file and the read/write methods have been standardized across the access tools: embedded console, JON Console, and native CLI. However, we noted that not all admin operations are available across these tools. For example, the native command line interface is required for many configuration changes, as neither the JBoss console nor JON can perform all configuration management. Only the CLI has access to all configuration parameters. We also discovered that if a CLI command is not entered correctly the configuration files can become corrupted, necessitating manual editing of the file to remove the partial update. This is a security and compliance issue (see above), as such edits are not logged for audit.
- **Backwards Compatibility** – Testing indicated issues with backwards compatibility between JBoss EAP 6 and JBoss EAP 5. Additionally, we found backwards compatibility issues between the community-supported editions of JBoss and the commercially supported JBoss EAP 6. This latter issue is expected, as Red Hat does not support the community version of JBoss and makes no claims as to its suitability for production or compatibility with any of their supported software.

- **Upgrade Migration** – JBoss EAP 6 (and its core, JBoss AS 7) constitute a major rewrite of the Application Server. Most of the configuration files have moved, so migrating the environment from EAP 5 to EAP 6 is a major undertaking. This is documented in the 108 page *JBoss Enterprise Application Platform 6 Migration Guide*[7]. We found that locating parameters that have moved is very time consuming and error-prone. By contrast, IBM provides automatic migration tools for configuration migration from WebSphere 6.0, 6.1, 7 and 8 to version 8.5 rather than leaving this error-prone process to the administrator. IBM also provides no-charge code migration tools[8] that will upgrade code from earlier versions, as well as migrating code from other JEE platforms (JBoss, WebLogic, Tomcat, OAS, etc). In addition, IBM has a dedicated Worldwide Migration Team that can be engaged for no charge migration assessments and skill transfer. The team can complete representative version to version upgrades to help users accelerate their migrations and apply best practices.

Summary

When choosing an application server, organizations should carefully examine the costs and risks associated with a given product in the context of their staff capabilities, application and environment characteristics, and overall risk posture to determine which solution will have the lowest total costs in the long run. IT organizations must frequently launch software implementations and select products based on limited information. The assessment of cost is frequently based exclusively on initial product acquisition costs when the total costs to a business over the lifetime of an application are often difficult to measure and can include the many other factors listed below:

- Cost of selection of the vendor software
- Developer admin and end-user training cost
- Cost of integration with other systems
- Runtime price/performance analysis cost
- Product upgrades cost
- Risk cost and risk management
- Deployment cost
- Application enhancement cost
- Requirements analysis and capture cost
- Application design and development costs
- Quality, user acceptance and other vesting costs
- Networking and hardware costs
- Downtime and SLA costs (planned and unplanned)
- Removal and disposal costs
- Operational support cost
- Regression defect resolution cost

In addition to failing to account for these costs, other pitfalls in calculating costs include the fact that direct labor cost estimates typically fail to account for the actual skill levels required for a task, especially for advanced configuration and administration tasks. Managers also frequently fail to account for the development of additional non-functional support code needed to maintain deployments, along with the ongoing care and maintenance needed for complex systems.

[7] https://access.redhat.com/knowledge/docs/en-US/JBoss_Enterprise_Application_Platform/6/pdf/Migration_Guide/JBoss_Enterprise_Application_Platform-6-Migration_Guide-en-US.pdf

[8] http://www.ibm.com/developerworks/websphere/downloads/migtoolkit/

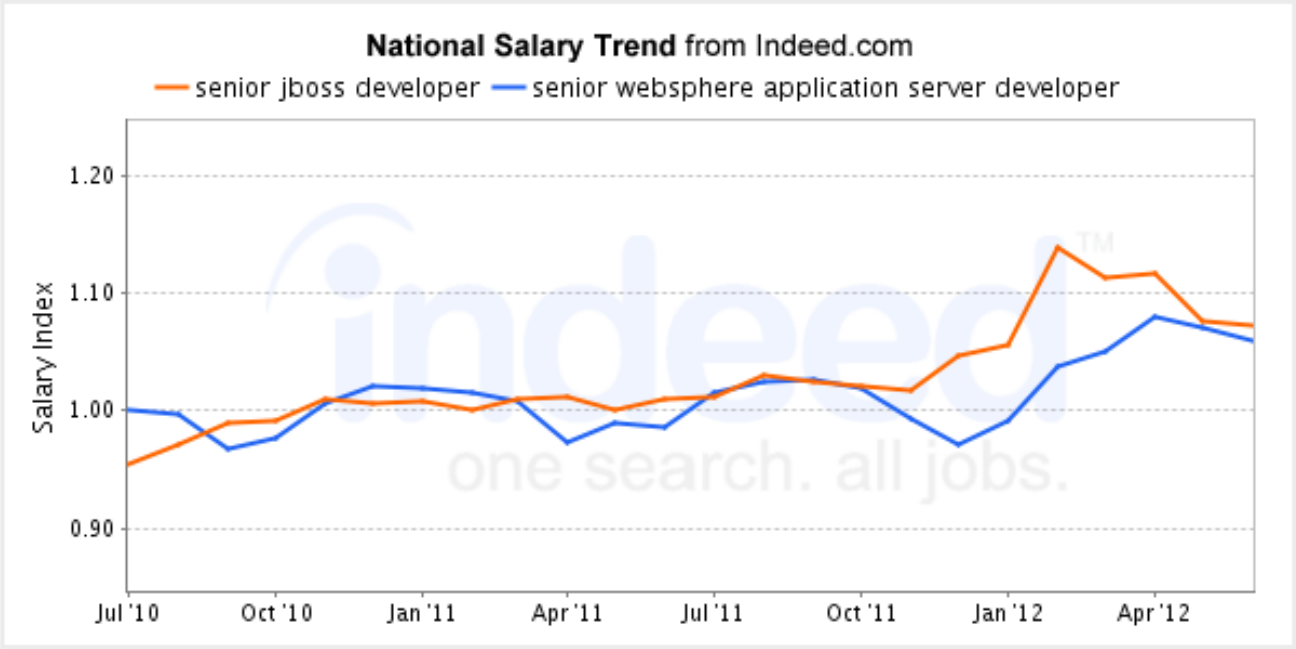
Cost Factors

The relative impact of TCO factors can vary widely between projects depending on the environment and type of application being deployed. The TCO factors considered in this study are described in the table below.

TCO Area	*Example Costs*
Software Licensing and Software Support	▪ Initial license and subscription acquisition costs for all elements of a solution, including application server, operating system, web server, load balancer, LDAP, HTTP server, JVM, etc. ▪ Ongoing product support and subscription costs
Hardware	▪ Server hardware and support costs ▪ Cooling and power costs
Application Management	▪ Installation and configuration costs associated with software development or implementation projects ▪ External system interface integration, support and testing costs ▪ Performance tuning and monitoring of application server environment ▪ Troubleshooting time for applications and development environments ▪ Learning the new, dramatically different user interface from previous versions
Infrastructure Management	▪ Deployment, monitoring and routine maintenance of application software ▪ Application of fix packs and patches to the OS and Application Server ▪ Upgrade of servers to newer versions ▪ Learning the changes to the management console and procedures ▪ Creation of new management scripts because of changes to the command line interface ▪ Backup and restoration of the OS and Application Server
Risk and Downtime	▪ Automated recovery from failures, including heuristic XA transactions ▪ Failover time of JMS queues and destinations, HTTPSessions and other stateful services, server restarts and memory leaks ▪ Consistency checks on configuration to prevent illegal settings ▪ Security tools for separation of administrative roles ▪ Auditing of administrative actions
Training	▪ Training/education for administrators and developers
Additional Considerations	▪ Costs to extend products to support service level agreements (SLAs) ▪ System integration costs for development and 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

We did not directly measure software development costs due to their high degree of variability based on an individual organization's software development practices, team sizes, and skill level. Assuming equivalent

developer productivity, however, development costs will be qualitatively lower for IBM WebSphere based on relative salaries of JBoss and WebSphere developers, as demonstrated in the chart below.



Analysis Findings

We executed a number of hands-on tests to calculate administrative costs and to gain a better understanding of the qualitative aspects of those costs for the products. The following is a list of the test scenarios executed:

Standalone Tests

- Install and Configure Application Server
- Upgrade Software from Previous Release
- Manual Application Deployment
- Configure LDAP for Administrative Security
- Scripted Application Deployment
- Apply Fix Packs and Hot Fixes
- Web Service Performance Tests
- JEE6 Performance Tests
- Heuristic Transaction Recovery Tests
- Extended stability tests (48 hours)

Clustered Tests

- Install and Configure Application Server
- Install JON in test configuration
- Upgrade Software from Previous Release
- Add Additional Server to the Cluster
- Configure XA and Transaction Logging
- Manual Application Deployment
- Configure LDAP for Administrative Security
- Scripted Application Deployment
- Apply Fix Packs and Hot Fixes
- HTTP Session Replication Failover Tests
- JMS Failover Tests

The TCO areas measured as a part of the study are discussed in the following sections.

Performance

TCO Advantage: IBM WebSphere Application Server

In this version of the study, we elected to perform several variations of standalone performance tests instead of long-running clustered performance tests. This shift in testing strategy was due to feedback from customers who indicated that very few of them were deploying JBoss in a clustered environment. Instead, they were deploying it for a purpose driven application under standalone conditions. We believe that by using the results for a standalone server we can extrapolate a best case scenario for clustered operations. Our performance testing strategy included the following test scenarios:

- 1 hour and 48 hours tests of WAS and JBoss using Apache DayTrader v 2.1.7
    - Single application server machine (with multiple JVMs if needed)
    - Single, remote web server machine
    - Remote database server running IBM DB2
    - Used JPA for the persistence tier
- 1-hour web service tests with small and large moderately complex payload (SOAP/HTTP, 15 kB and 110 kB in/out)
    - Single application server machine with multiple JVMs, if needed
    - Single, remote web server machine
    - No database
    - JAX-WS annotated POJO service

As part of the performance testing, we spent time tuning both JBoss and WAS for test cases. The software and hardware configurations used in the tests are described in the "Appendix C". Average server CPU utilization was greater than 90% with JBoss and WebSphere.

Prior to each of the performance tests being executed, we tuned the configurations for each environment (JBoss and WAS) to achieve optimal performance. There are a number of interesting discoveries that were made as part of the tuning process.

Tuning of WebSphere was a relatively straightforward process. WebSphere built-in Tivoli Performance Viewer was an important tool throughout the tuning and monitoring process because we were able to review relevant stats quickly and easily, including changes in values over time. We also found that the tuning advisor in WebSphere recommended more conservative settings and in our case were only used as a starting point for the tuning process. All in all, we saw significant improvements in WAS performance with only a few minutes spent on basic tuning.

The process of tuning JBoss was more complex, as we found various contradictory configuration and tuning advice. We found that some of the recommendations actually reduced performance over the out-of-the-box settings. We thus found no reliable tuning information, so we did the best we could with recommendations posted to forums. We feel that our results are consistent with what most users would see. As it was, we wasted a week just trying to find the optimal tuning parameter documentation.

The above factors identified during the course of performance tuning and testing will have an impact on TCO. We found that tuning JBoss is a more labor intensive task than tuning WebSphere and required a higher level of skilled resources. Several of the reasons are highlighted below:

- **Little Actionable Information in Documentation** - The JBoss EAP 6 website and customer portal (including documentation, wikis, and forum postings) yielded little actionable information. We had to use third-party websites to gain better understanding of what the various configuration options were and what they meant. (To use the customer portal we registered as developers and paid the standard support fee.)

- **Documentation Quality** – The EAP 6 documentation can be both bad and good. It gives a better plain English tone than the WebSphere Infocenter documentation; on the other hand, there appear to be more errors, omissions, and contradictions. The layout of the EAP 6 documentation also varies. Some sections will list Windows instructions first, while others contain only Linux steps. On occasion, there are notes for platforms other than Red Hat Linux, and there are a number of references along the lines of "consult so and so's documentation". We found that more often than not, we had to deviate from the official EAP 6 documentation and consult the documentation for the subprojects that get rolled into EAP 6. For example, if an issue comes up with messaging, you'd likely be better off going to the HornetQ project documentation, and only for the particular version that is included in EAP 6. Same thing for Hibernate, *mod_cluster*, JBoss AS, etc...

  By comparison, WebSphere infocenter documentation is notably constant with a black and white tone. Additionally, it doesn't have the same degree of errors and contradictions found in the EAP 6 documentation. In particular, the cross-platform documentation is more thorough and complete. The canonical documentation for all features and functions is located in one central location, which can reduce the time to troubleshoot and recover from issues.

  It is also sometimes difficult to determine the JBoss version that given documentation refers to; this leads to misinformation and chasing one's tail.

- **Best Documentation Only Available Through JBoss Training** – Red Hat provides training as a major profit center. Class attendees get advanced, higher quality information than is available to the average user, but at a very high cost. This is not included in TCO, but can be a major factor in adding to the cost of deploying JBoss. In contrast, all WAS documentation is available online in the InfoCenter and extensive Redbooks, as well as free or low cost self-paced training.

- **Incorrect Information in Documentation** – We found many examples of incorrect information in the JBoss EAP 6 documentation. Some these are mentioned in context in sections below. We also found much missing information, as will be described.

- **Issues with Performance Monitoring Tools** – Red Hat has eliminated the JMX console in EAP 6[9]. Red Hat recommends jconsole as a replacement. Monitoring metrics can be accessed via HTTP calls, which is the method employed by the embedded Administration Console as well as JON. JON provides a portal like environment where a monitoring workspace can be customized to display multiple metrics at once. Upon first use, the JON console is impressive and clean. However, as we got deeper into it, we experienced difficulty in navigation or creating dashboards as we had envisioned.

- ***mod_cluster* Scalability Issues** – The *mod_cluster* plugin scalability issues found in EAP 5 version appear to be resolved in EAP 6.

Application Simulation Performance

Extensive tests were run using the Apache DayTrader application, with a wide range of durations, settings and loads. We found that for the DayTrader application, WebSphere had a performance advantage over JBoss ranging between 3% to 17%. This represents a significant improvement in JBoss performance over the previous version, where WebSphere was typically 50% faster. Clearly, Red Hat focused on improving performance for web applications.

Web Service Performance

Web Service performance tests were run on both WAS and JBoss. The tests on WAS do not include the performance enhancements gained by using the included XML feature pack in WAS 8.5; however, internal

[9] https://access.redhat.com/knowledge/solutions/92223

tests at IBM labs show a significant benefit to using it beyond what we report below. Even without the XML feature pack, however, WAS 8.5 outperformed JBoss by a wide margin of 50% to 96% depending on message size and load.

A test of web service response was run on "Out-of-the-box" untuned configurations for both JBoss and WebSphere. Untuned means the application servers were used as is with default settings. For JBoss EAP 6 the *standalone.xml* configuration file was used, and in WebSphere Application Server v8.5 the default template was used to create the application server via the WAS administration console. No changes were made to heap sizes, thread pools, etc. Results are:

Table 1 - Untuned Web Service Test Results, 1 hour test

Users	Message Size	WAS tps	WAS rt	JBoss tps	JBoss rt	WAS:JBoss
50	15K	2582.03	18.32	2032.13	24.06	127%
200	15K	2605.68	75.44	2069.07	95.14	126%
50	110K	160.75	310.42	125.52	397.75	128%
200	110K	157.83	1253.59	119.53	1655.42	132%

The tests were rerun after tuning. We noted that the tuning did little to improve JBoss, but represented a significant improvement in WebSphere.

Table 2 - Tuned Web Service Test Results, 1 hour test

Users	Message Size	WAS tps	WAS rt	JBoss tps	JBoss rt	WAS:JBoss tps
50	15K	3653.77	13.14	2042.28	23.94	179%
200	15K	3054.13	64.21	2013.53	97.78	152%
50	110K	208.54	239.17	106.17	470.33	196%
200	110K	192.7	1026.68	106.26	1861.55	181%

The tuning parameters used were:

Table 3 - Web Service Tuning Parameters for Tuned test

WAS	JBoss
-Xms1024m	-Xms1024m
-Xmx1024m	-Xmx1024m
-Xaggressive	-XX:MaxPermSize=256m
-Xlp	-XX:+UseLargePages
-Xdisableexplicitgc	-XX:+AggressiveOpts
-Xgcpolicy:gencon	-XX:+DisableExplicitGC
-Djava.net.preferIPv4Stack=true	-XX:+UseParallelOldGC
-Xnoloa	-Djava.net.preferIPv4Stack=true
-Xshareclasses:none	-Dorg.jboss.resolver.warning=true
-Xtrace:none	-Dsun.rmi.dgc.client.gcInterval=3600000
-Dsun.net.inetaddr.ttl=0	-Dsun.rmi.dgc.server.gcInterval=3600000

Distributed Transaction Performance

We encountered serious problems with distributed XA transactions (see Transactional Integrity on Page 16) that would make one think twice about using JBoss in any application where database synchronization is essential. We realize, however, that the probability of a failure in an XA transaction is small, and that some users are not concerned about a failure as much as about performance. So we ran a series of XA tests without forcing failures of a data source. These results reflect default settings for both JBoss and WAS and are the best runs of 4 samples:

Application Server Configuration		
Heap Size: 50M/256M DB Connections: 25/25 WAS: Thread Pool – Web Container: 10/100 JBoss: Max Web Connections: 100		
Test Parameters		
40 users 10,000 transactions		
Results		
Vendor	TPS	Errors
WAS	219.48	0
JBoss	18.83	12

These results show that WAS is 1170% faster than JBoss for distributed transactions. The XA performance tests were run after tuning the JBoss environment.

Summary of Performance Tests

Our performance tests described above indicate a performance advantage favoring WAS Base over JBoss by a factor of between 1.1:1 and 2:1 with the special case of 12 to 1 for distributed transactions. This difference in performance will increase hardware, software, and administrative costs for JBoss when attempting to achieve comparable performance results. We have made an assumption that the application will have a blended workload consisting of 80% of the transactions similar to what is simulated in DayTrader application and 20% web services workload. Overall we conservatively used a performance factor of 129% - a WebSphere advantage (factor 100% would mean identical performance).

For the complete documentation of the performance test, project files, load scripts and data files, please contact your IBM representative or Prolifics.

Transactional Integrity

TCO Advantage: IBM WebSphere Application Server (We do not recommend JBoss for any application that requires transactional integrity across 2 or more data sources)

Poor quality data will cost the banking industry 27 billion USD in operating costs[10]. The same study suggests that half of customers will give their banks only two times to make a mistake before considering a change to another bank. The diagram below illustrates a simplified version of a money transfer transaction. In most banks, the Core Banking System is a separate application from the Internet Banking, Credit Card System and several other systems not shown here. The “money transfer” operation must reliably update all of the related systems either as a single unit of work (XA) or long running transaction.

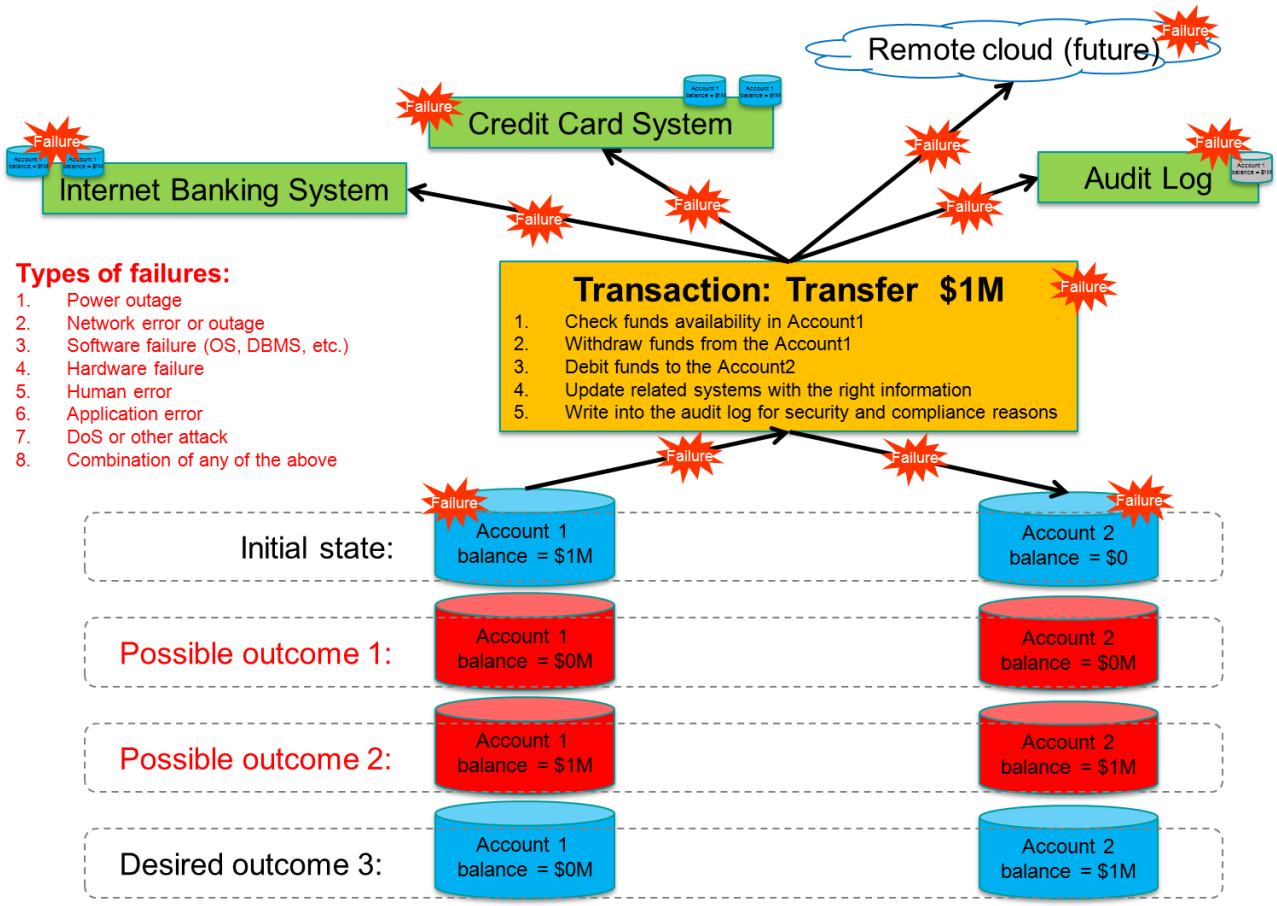
Transactional integrity is important when multiple related steps need to be completed for a successful transaction. For example, what would happen if a customer were to initiate a funds transfer between two accounts only to have a server crash in the middle of the transaction? Ideally, the system would be able to automatically detect the failure and take the appropriate action. Another potential solution would be to roll back the transaction and allow the user to try again. If not handled appropriately, though, valuable transaction data, including your customer’s money, could be lost or recorded incorrectly. A transaction failure in a critical system could cost a company millions of dollars in lost revenue and lost customer satisfaction. As a result of this risk, it is essential 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.

WebSphere Application Server version 8.5 provides a highly effective transaction engine, which was built on top of a solid foundation from the time of its introduction over ten years ago. It is designed to

[10] http://www.ibm.com/smarterplanet/us/en/banking_technology/ideas/?&re=spf

automatically detect and recover from transaction problems that occur during two-phase. This functionality is enabled by default and configuration is limited to how recovery behaves. WebSphere Application Server V8.5 also includes the ability to view transaction logs during run time. These capabilities help ensure transaction integrity for WebSphere Application Server users.

As reported in 2010, JBoss EAP 5 seemed to have a design flaw that prevented its transaction engine from correctly recovering in-doubt transactions. This problem has been documented by Red Hat in their Issue Tracking system as a defect (JBAS-7965[11] and JBAS-6244[12]). The defect describes a JBoss AS transactional integrity recovery issue, but does not specifically address how to fix the problem. This raises the question as to whether JBoss can automatically recover from all transaction integrity failures. Moreover, there appears to be no documentation on how to configure or fix JBoss to automatically recover from the transactional integrity issues when in-doubt distributed transactions fail. Instead, there are support forum postings showing the specific transaction recovery problem with the JBoss Application Server or discussions about the transaction recovery problems that do not address how to actually fix it. This issue was never corrected in EAP 5, as demonstrated by a continuing stream of posts in support forums.



IBM did a study[13] based on this public information in an attempt to determine whether the transactional integrity problem in JBoss could be recreated or if it has subsequently been fixed.[14] It appears that the problem still exists in JBoss EAP version 5.0 and the JIRA records are still in the state "Critical-Unresolved". This means that XA transactions with databases, messaging systems, adapters and all other resource managers cannot be reliably recovered by JBoss if the failure happened during the second phase of

[11] https://jira.jboss.org/browse/JBAS-6244
[12] https://jira.jboss.org/browse/JBAS-7965
[13] http://www.youtube.com/watch?v=UzWVXRoe15Y
[14] ftp://ftp.software.ibm.com/common/ssi/sa/wh/n/wsw14068usen/WSW14068USEN.PDF

the 2PC protocol. This leads to a loss of data or significant labor costs to manually analyze and resolve every failed instance. It shall be noted that manual recovery is time consuming, very costly and can cause additional errors - it is manual after all.

With the release of JBoss EAP 6 Prolifics undertook to determine if the problem persists in this latest release. Prolifics also ran the same test with WebSphere v8.5 for comparison purposes.

Test with JBoss EAP 6

The test was run with 3 VMware guests on an otherwise dedicated server. 1 VM was the app server, and the other 2 VMs had one DB2 database each representing application data. The test application simply transferred money from accounts in one database (TXBank1) to their associated accounts in the second database (TXBank2). The transaction was initiated prior to withdrawing money from an account in TXBank1 and was committed after depositing the money in the associated account in TXBank2.

Load was generated from a laptop to the application server, and during the test the network connection was disabled on the VM running the TXBank2 instance after processing 500 or so transactions. The load generator was stopped and the in-doubt transactions queried on the TXBank2 database. After verifying in-doubt transactions existed, the network connection was re-established to enable transaction recovery.

JBoss was run with the *standard-full.xml* configuration file, and all configuration changes required to enable transaction recovery were made to this file. It is important to note that transaction recovery on XA data sources is not enabled out-of-the-box in JBoss EAP 6. Specific configuration settings must exist in the XA data source definition in order to enable transaction recovery and these are only settable by using the new command line interface or manually editing the associated configuration XML file.[15]

Furthermore, missing configuration items cause unexpected results. The test was run multiple times. The first few times, in-doubt transactions were seen on TXBank2 and appeared to be recovered over a period of time once network connectivity was re-established. Once the recovery appeared to work, the databases were queried to see if transactions had recovered correctly. We found that transactions were not recovered correctly, leading to amounts being out of sync between associated accounts in each database, in-doubt transactions were cleared and locks released on associated database rows. Further updates to the out of sync accounts would only compound the errors and make it even more difficult to correct the original problem, if it was even possible to correct it.

Upon further investigation and referencing EAP documentation, it was discovered that the XA data sources were not configured correctly – the configuration was incomplete – to effectuate transaction recovery properly. Thus this led to the first finding, which is that incorrect or incomplete configuration of transaction recovery can lead to unexpected results and leave data in associated databases out of sync and open to further updating without any errors being reported. This is compounded by the observation that there is no indication that the server is incorrectly configured.

With proper configuration completed for transaction recovery for the XA data sources, the test was run again. This time, of the in-doubt transactions seen, only half were recovered and the other half were left in-doubt. Querying the database did show that only half the accounts remained out of sync – the same accounts that were part of the in-doubt transactions.

The database was restored to its original state and another test was run. This time the application server was restarted after the network connection was restored to the virtual machine hosting the TXBank2 database. After a period of a few minutes, JBoss did recover the in-doubt transactions properly and the accounts in both databases were in sync.

---

[15] See: https://access.redhat.com/knowledge/docs/en-US/JBoss_Enterprise_Application_Platform/6/html/Administration_and_Configuration_Guide/Example_IBM_DB2_XA_Datasource.html

The two results were consistent over multiple tests. Leaving the application server running after a database failure recovery recovered the in-doubt transactions (thus showing that recovery had been successful), but left the databases out of sync, meaning that the recovery was not successful.

Restarting the application server after the database failure recovery recovered the in-doubt transactions and left the two databases synchronized. As long as the application server was restarted after the database failure was recovered and before JBoss attempted to recover the transactions, JBoss XA recovery appears to work, however, it is clear that unless an administrator recognizes in real time that a problem occurred and restarts the servers data will be lost.

This leads to the second finding of the test– JBoss EAP 6 does not consistently recover two phase commit transactions that are in the HEURISTIC state (second phase of a 2 phase commit) properly without user intervention.

Test with WebSphere v8.5

The same test scenario described above was run against WebSphere Application Server v8.5. In this test, all in-doubt transactions were recovered and the databases were in sync after a failure and restoration on the TXBank2 database.

No configuration was required to enable transaction recovery for the XA data sources. Only minimal configuration changes were made to the Transaction Service at the Application Server level. Default values for the following items were changed to the values specified:

1. Total transaction lifetime timeout – 20 seconds
2. Async response timeout – 15 seconds
3. Client inactivity timeout – 20 seconds
4. Maximum transaction timeout – 20 seconds
5. Heuristic retry wait – 15 seconds

Conclusions

We see from this test that JBoss EAP 6 still has transactional integrity problems where more than one data source must be updated as an atomic transaction. We cannot assign a TCO value to this, as it would have to include the cost of failed transactions impact on the business, which could be in millions or even billions of dollars for large financial institutions.

We must therefore conclude that JBoss is unsuitable for any application that utilizes 2 phase commit transactions against 2 or more databases.

Hardware Costs

TCO Advantage: IBM WebSphere Application Server

We executed several performance tests for JBoss EAP and WebSphere Application Server in single server configurations using the Apache DayTrader test application and a simple web service implementation running under a simulated load. Details of the tests can be found in the performance testing section of this paper.

Based on the results of the performance testing, a JBoss-based solution will need between 10% and 60% more hardware to handle the same load as a WebSphere-based solution. For basic web applications WAS has roughly 10% better throughput than JBoss, for Web Services WAS throughput is 60% to 96% greater, and for 2 phase commit transactions it is 1170% faster. As each case will be different, the exact impact of these performance figures must be evaluated given the application workload. We excluded the XA (2 phase commit) performance, as currently JBoss cannot be recommended for applications that require synchronization between multiple data sources.

In addition to performance differences, WAS ND includes Intelligent Management of dynamic clusters. That is, a cluster that can automatically grow and shrink in size (vary the number of active JVMs) based on near real time performance demands. By vertical stacking of dynamic clusters across multiple servers, applications that experience peak loads at different times can share processing power. In the typical data center servers run 20% to 25% average utilization, allowing a reserve for peak times when additional processing power is needed. WAS ND dynamic clusters, on the other hand, can safely run servers at 85% or higher utilization when there is a spread of applications that experience peak demand at different times. This can reduce the hardware footprint (and software license costs) by a factor of 2.5 or greater (we have seen as high as 3.5).

Users of the predecessor of Intelligent Management, WebSphere Virtual Enterprise (WVE), have historically reported reduction in hardware footprint of 40% or better when deploying on WVE. Prolifics has one client who is running over 300 dynamic clusters on 1,000 physical servers, controlled from a single console.

In addition to improved TCO, the use of dynamic clusters results in higher availability, more robust failover, efficiencies around application-centric virtualization and policy-driven workload management.

Software License, Maintenance and Subscription Costs

TCO Advantage: Red Hat JBoss EAP 6

The mantra of commercial open source has long been, “If you don’t need support, don’t pay for support. If you need support, buy support.” For some open source products and use cases this may be true. However, our testing indicates that a JBoss EAP subscription is essential.

Through the course of our analysis we compared the community supported version of JBoss AS to that of the commercially supported JBoss EAP product. Of the 32 open source projects, which make up both products, we found that JBoss EAP uses newer versions for 14 of the components. To get the same version levels of these projects, one would need to manually integrate these projects. Manually integrating these 32 open source projects into an equivalent “free” package, similar to what CentOS has done for RHEL, would be very time consuming and costly.

For the TCO study we obtained list pricing for both JBoss EAP[16] and IBM WebSphere Application Server.[17] We decided to use Red Hat’s premium support pricing for TCO calculations since it provides the closest match to IBM’s basic 24x7 support. The hardware configuration used for our license cost analysis assumed 2 CPU sockets per machine with each socket having 8 cores, thus bringing total number of cores on a machine to 16.

There are many factors that can affect software licensing costs. The following is a list containing some of the more important items to take into account when considering alternatives:

- Both Red Hat and IBM will discount pricing based on volume of business and application environment deployment, affecting individual cases very differently. The license and support cost analysis performed in the study is based on “street” prices assuming 35% discount off the list price for both IBM and Red Hat. We believe both of these are appropriate for the configuration of the size used in the study, though there is some evidence that suggests that Red Hat discounts very rarely go above 20%.
- JBoss EAP 6 pricing requires purchasing capacity in groups of 16 or 64 CPU cores. If someone needed a configuration with fewer than 16 total cores, Red Hat would still sell 16 cores configuration. IBM requires licenses for only the number of cores actually used. So an 8 core server would only require 8 units of license, while the same server running JBoss EAP 6 would be licensed for 16 cores.

[16] JBoss NA Channel SKUs, http://www.redhat.com/f/html/jboss_channel_skus.html.
[17] IBM WebSphere Application Server Pricing, http://www-01.ibm.com/software/webservers/appserv/was/appserv-was-pricing.html.

- Red Hat no longer provides JON subscriptions free of charge with JBoss support and now requires all customers to pay for JON separately (in the past JON was free for those clients who purchased over 32 or more CPUs of JBoss support).
- Both Red Hat and IBM have similar policies regarding sub-capacity licensing when using virtualization technologies such as VMware or other hypervisors.
- WAS ND v 8.5 now includes WebSphere Intelligent Management functions. The included features provide for dynamic clusters, comprehensive health monitoring, SLA policy management with autonomic cluster expansion and contraction of dynamic clusters, request prioritization, application edition deployment management and request queuing.
- WAS ND includes free licenses of the Tivoli® Directory Server, which can act as an LDAP server; JBoss EAP does not include a bundled LDAP server so one will be required. The cost of a Red Hat Directory Server was factored into the analysis for JBoss EAP 6.
- Since June 2009, IBM has offered IBM WebSphere Application Server for Developers Edition (WASDE) as either a no-cost community supported version or a paid support version. WASDE is the exact same code as WebSphere Application Server Base. Customers who have at least one full WebSphere license will also have access to the Rational Application Developer Assembly and Deployment Tool.[18] This tool is a subset of Rational Application Developer, which can be used to develop and package applications for deployment on WebSphere.

The TCO components for software licenses and support/subscription costs favor JBoss EAP 6 over the five-year period.

Application and Infrastructure Administration Costs

TCO Advantage: IBM WebSphere Application Server

As part of the TCO study we executed a number of time motion scenarios to understand the steps involved in various administrative tasks, the relative complexity of the tasks and issues that were encountered while carrying out these tasks. Specific scenarios evaluated include the installation, upgrade, migration, configuration, clustering and security hardening of an environment for a moderately sized application environment. For each test case, we calculated its impact on a five-year comparative TCO difference.

As part of the testing we also studied the effects of the learning curve associated with each of the products along with the skill level required for each product. The cost calculations use the best case results from each of the test cases to prevent over inflation of administrative costs associated with newer administrators coming up to speed.

With regard to the learning curve, our tests indicate that WAS ND has a lower overall learning curve than JBoss EAP. We found that execution of a given task with JBoss EAP can take substantially longer to complete than with WebSphere. We also found that on average, the skill-level required to complete tasks with JBoss EAP 6 was higher than the skill-level required with WebSphere.

We previously noted that JBoss EAP 6 is a major rewrite with new administrative tools and changes to existing tooling and administrative data structures. This essentially means that experienced administrators will need to learn EAP 6 as if it was a completely different product from EAP 5. While we found the newer interface and support utilities superior to the prior versions, they still need to be learned anew. This is reflected separately in our analysis for training, but it also means that administrators will be less productive initially after updating to EAP 6.

The remainder of this section describes the various scenarios tested, the results of each test, how the results correlate to TCO, and a discussion of notable findings encountered during the course of testing.

---

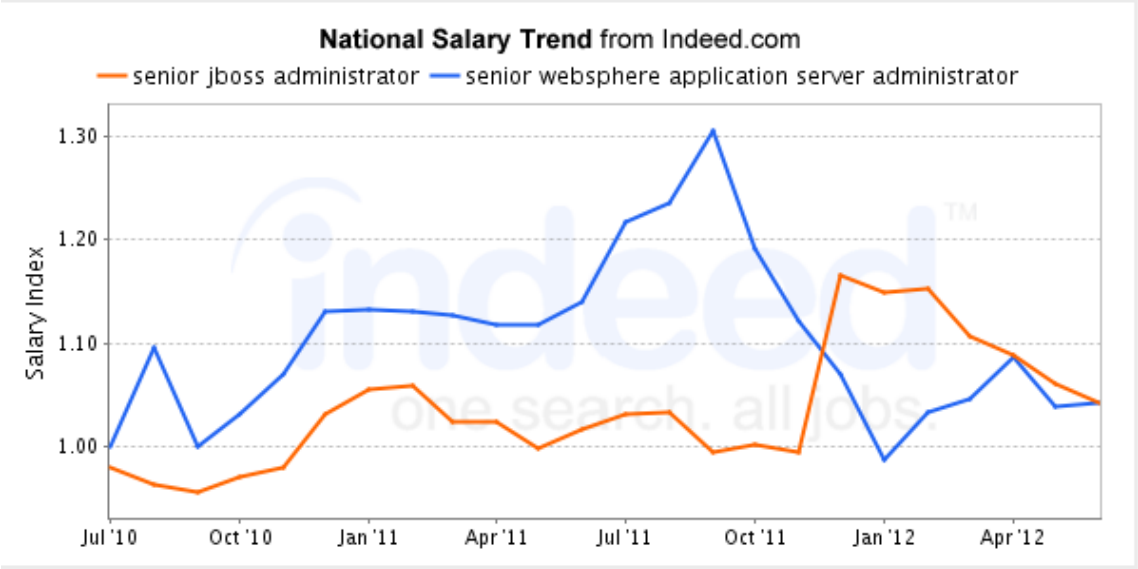
[18] IBM WebSphere Application Server for Developers FAQ, http://www.ibm.com/developerworks/downloads/ws/wasdevelopers/faq-wasdevelopers.html.

The study divides administration costs into two categories - application administration and infrastructure administration - to better reflect how many readers viewed administration costs.

Application administration costs are the costs most directly tied to managing an application during a software development project. These are the costs associated with standing up shared development environments, QA environments and the initial production environment. Many organizations treat this category as either capital or planned work projects.

Infrastructure administration costs are focused on the day-to-day operations of keeping a production environment up and running, applying fix packs or adding additional capacity to support increased application demand. Many organizations tend to categorize this work as expense work.

Note also that as JBoss has become more widely used, the labor cost for administration resources has gone up and administrator costs for WebSphere has gone down, as shown below. This must also be taken into consideration in calculating support costs. These two are likely to level out in the future, as the trend tends to show. We have thus assumed that the cost of labor for administrators is the same for both WAS and JBoss in our analysis.



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 and compared the initial product acquisition, download, installation and configuration experience for JBoss, EAP6, WAS and WAS ND. It also covered initial deployment (non-clustered) of the test application.

WebSphere Installation

IBM has greatly simplified installation of all of its distributed products using a universal IBM Installation Manager (IIM). We used IIM for the entire installation. Time to complete the installation was about 2 hours (including download time for the installer), and it worked the first time. It would be faster for additional installations (typically approximately10 minutes), as IIM can be scripted for silent installs.

It should be noted that this section deals only with the installation of the fully configured WebSphere v 8.5. WebSphere Liberty Profile, described on page 44, is more comparable to JBoss AS. Liberty Profile is installed by unzipping the under 50 MB installation package and is ready to go immediately, totaling an installation time of about 5 minutes.

WAS Server Preparation

We installed WAS ND v8.5 and associated products in Red Hat Enterprise Linux 6, 64-bit VMs. Our customer experiences indicated this was a widely adopted platform and was confirmed to be a common support platform across providers. Beyond the usual preparation necessary for a typical Linux environment, the combination of WAS and RHEL 6 requires installing or updating of libraries that may not be installed in a normal RHEL 6 install. These are documented here: http://pic.dhe.ibm.com/infocenter/wasinfo/v8r5/topic/com.ibm.websphere.installation.nd.doc/ae/tins_linuxsetup_rhel6.html. The main issue is supporting 32-bit applications on a 64-bit machine and missing libraries for rendering graphics. This is necessary to use IBM Installation Manager as well as WAS 32-bit, if so desired. The Installation Manager (IIM) greatly simplifies installation, patch management and configuration and is the recommended way to install WebSphere. It's just a matter of checking the items desired and telling it to go.

WAS Install Media

Install media for WAS 8.5 ND is large. The WAS package itself is 2,862 MB in size, not including the supplements (IBM HTTP Server, WebSphere Plugin, Edge Components, etc.) that are an additional 2,863 MB, nor the IBM Installation Manager itself (120 MB).

We are using the IBM HTTP Server and WebSphere Plugin in our tests, and in order to install them, you have to download the entire Supplements package (3 downloads in total).

Even though the download packages are large, they support multi-platforms (OS and architectures), so it is not necessary to root around on the Internet to find options that are needed.

WAS Installation Process

WAS and IBM SDK 7 Install

For installing WAS, the IBM Installation Manager was used on one server in record mode. This creates an XML response file that can then be used to install the product on other servers silently. This is especially useful for servers where there is no graphical interface enabled. We encountered no issues in running the installation silently on two other servers.

A helpful feature of the IBM Installation Manager is that you can install multiple products at the same time. Dependencies are checked and error messages are given which halt the install until such time as the dependency or problem is corrected.

We installed both WAS and the IBM SDK 7 at the same time. The default SDK that comes with WAS is v6. IIM will not allow you to not install IBM SDK 6 in order to save disk space. If you install WAS and IBM SDK 7 at the same time you have to install the IBM SDK v6, and there is no way in the installation process to indicate which SDK version to use. As such, an administrator has to go in after the environment is set up and choose the SDK via the WAS Console or by using the *managesdk* command.

IHS and Plugin Install

IBM HTTP Server and the Plugin use IIM as well for their installation. In WAS versions before v8 the GUI install process for the IBM HTTP Server would walk you through installing both IHS and the WAS Plug-ins and do some initial configuration of an both the default *httpd.conf* file and the configuration file for the IHS administration process. This left very little configuration to be done after the installation process.

With IIM, that configuration piece has been removed and is left to the administrator as a post install step. Items left to be competed included adding the load module for the WAS plug-in to the *httpd.conf* file, running the *setupadm* command to create a user ID to run the IHS admin process under and manually modifying the *admin.conf* file to add a port the admin server instance will listen on.

Evaluation

WAS Pros

IBM has consolidated their installation strategy across products and settled on the IBM Installation Manager. Now all WAS products use this tool, as well as Rational Products. It breeds familiarity and consistency in the installation experience.

It further aids in the update process of the products, as all updates, modifications (add/remove features), and uninstalls are done from IIM. If the servers in your environment have Internet access, it is very easy to find out what updates are available and apply them. No more having to keep up with versions of Update Installer, downloading fix packs or hot fixes and applying them using Update Installer.

The installation process can be scripted using response files and, furthermore, can be centrally managed and run from the WAS Job Manager. Job Manager allows for installation of WAS products across multiple platforms simultaneously to help reduce the total time to install on a given environment. This is particularly useful for large environments.

WAS Cons

The size of the installation media is large, and some people may balk at having to install IIM to install the WAS products. This is not an unzip-and-go install.

The advantage for updating software automatically cited above requires that the servers in your environment have Internet connectivity. If no Internet connection is available on the target computer (for example, if information security requirements dictate that a Production environment be isolated), an extra step is required; the updates are downloaded separately, installed in a local repository, then installed with IIM.

Learning how to use the silent install requires some experience, but the process is well documented in the WAS InfoCenter as well as the IIM InfoCenter, which also provides some sample response file for installing the various products and features of WAS.

JBoss EAP Installation

JBoss Server Preparation

We installed JBoss EAP 6 and associated products in RHEL 6.2 64-bit VMs. We tried to add 64 bit OpenJDK 1.7 for running the app server per the install guide; however, the command to install openJDK 1.7 was incorrect, and the correct one had to be determined. Note that JBoss EAP 6 does not have an installation manager. The correct components must be identified, downloaded separately, installed, the configured to work together. We installed the following components:

- JBoss EAP 6 (115 MB)
- OpenJDK 1.7 (92 MB)
- HTTP server (4 MB)
- JON 3.1 (230 MB)
- PostgreSQL (4 MB)

JON requires a database, and supports only Postgres and Oracle. We chose Postgres as it is open source and would be the likely choice for most users.

JBoss EAP 6 Install Media

The total size of the install media for all components is about 450 MB for our configuration. The smaller size, relative to WebSphere, is a reflection of limited functionality and OS platform support. The IBM installation package includes all supported platforms and languages; for JBoss the user must choose the correct packages to download and install.

JBOSS EAP 6 Installation Process

App Server

We installed a standalone App Server on VMware guest *wpapp1* using the most common method – extract the zipped install package. This was simple and straightforward. More servers can be created by repeating this process or copying an already customized install directory. To be able to access the Administration Console of a running server, a trivial configuration step was required after unzipping.

HTTP Server and HTTP Connector

We installed the http server package provided with EAP 6, the Enterprise Web Server, on VMware guest *wpweb1*. This is essentially the apache web server. We used the same process that was used for the App Server, extracting a zip file. For the connection back to the App Server, we used the http connector recommended in the documentation (*mod_cluster*).

JBoss Operations Network (JON)

JON is a management platform for JBoss EAP. It combines some of the functions of IBM Installation Manager, Deployment Manager, and Job Manager, but does not provide as robust a capability as the equivalent WebSphere management console. There is some overlap in functionality as far as management of App Servers and Web Servers. There are some specific things that you can do with the WAS Administration Console that cannot be done with JON, but there are many features of JON that fall outside the scope of this paper.

We installed JON on our administrative server *wputil* by unzipping it. This was straightforward and quick. It did not run initially, as it requires PostgreSQL to be running. We set PostgreSQL to autostart on system reboot; however, there is a potential race condition that must be addressed to assure that the database is running before JON starts.

We found contradictory information in the JON install guide as to whether JDK 6 or JDK 7 is supported or preferred.

We had some other issues with PostgreSQL and RHEL memory settings that took some work to resolve, as documentation on configuring the database and JON is lacking, and the race condition mentioned above must be dealt with.

Evaluation

EAP 6 Pros

- The unzip method is quick and simple for laying down EAP 6.
- Nearly all configuration operations can be done via JON or the management CLI.

EAP 6 Cons

- The quick download and unzip feature of EAP 6 components is offset by the fact that a JBoss EAP 6 installation requires downloading, installing and configuring 5 different products (after first identifying the required version of each), then configuring them to work together. This complicates the installation process and was actually much more time-consuming than the WAS installation, which was simply a matter of checking the desired options, then letting the automated process run to completion.
- Before the EAP 6 Administration Console can be accessed, a configuration step is required to secure the console (it is not optional in EAP 6). WebSphere, by comparison, can access the Administration Console immediately after installation and is secure out of the box.
- Time needs to be dedicated to understanding the difference with all the configuration file options. For example, out of the box, there are four standalone configuration files. The most light-weight config file is used by default. Documentation is weak on the differences between all four standalone configuration files and how they relate to each other. An experienced JBoss

administrator would be able to deal with this, but someone new to the product could easily make configuration errors that would have profound downstream impacts.

- It was discovered that to integrate with the recommended http connector, *mod_cluster*, only two of the four out-of-the-box configuration files are valid and can be used. Considerable time was spent researching the correct configuration.
- In some cases, we determined that manual editing of config files is required, a security and traceability issue.
- PostgreSQL must be installed, configured, tuned and secured and schemas created for JON to start working. Also HA configuration and often dedicated hardware for PostgreSQL will be required.

Summary – Installation and Configuration

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

- Production Server Installation Time
    - Applying patches and hot fixes to JBoss is a more labor intensive process than WAS. With WebSphere, the Installation Manager finds and installs patches; with JBoss, the administrator must manually search the support site for available updates.
    - The JBoss administrative console is not that useful for configuration of production instances; JON or the CLI is necessary.
- Server Configuration
    - Security hardening of JBoss administrative components for production use is more time consuming and difficult than with WAS, but is improved over EAP 5:
        - JBoss now has admin security enabled out of the box – a significant improvement.
        - Configuring the server for LDAP integration requires manually editing various files on the file system.
        - Securing JMS communications can be accomplished using JON, the CLI interface or configuring various files on the file system (the JBoss Administration Console is not suitable for configuring a production environment). However, the documentation is incorrect on how to effectively accomplish this, and there are many conflicting sources of information on JBoss discussion forums, wiki pages, and blogs.
    - Security hardening of WebSphere administrative components is simple and straightforward.
        - During the installation process, the administrator is prompted to select a username and password for the Administration Console.
        - Once initial installation is complete, the administrator can use a wizard in the administration console to connect the server to a variety of LDAP servers.
        - The IBM InfoCenter provided accurate steps for configuring access to our Microsoft Active Directory.
- Prebuilt Configurations
    - JBoss includes a number of pre-configured server configurations based on usage type. However, the differences between the configurations are poorly documented. It is also unclear what is required to create your own combinations of servers. Switching from one configuration to another requires maintaining the application binaries and relevant configuration information or third-party libraries within multiple directories on the file

system. Choosing the wrong configuration may not be evident until much later, and can even result in loss of transactional integrity.

- Based on the applications deployed to a server, WebSphere will determine which components, such as web container, EJB container, JMS, etc., should be loaded at runtime. This approach does not require administrators to take any action, or configure a server any differently to optimize the server's runtime footprint. For an organization with multiple servers, this greatly simplifies the configuration, deployment, and optimization process for servers.
- WebSphere provides an option for optimizing a server for development or production use. We found that in the default production mode, WAS will enable features that increase startup time but improve overall efficiency when the application server is running for extended periods of time. The development mode decreases the server startup time by deferring initialization of these components until they are first used. Switching between production and deployment mode is simply a matter of clicking a checkbox and restarting the server; applications do not need to be redeployed or reconfigured.

- Web Server Configuration
  - Our testing indicated that IBM HTTP Server (a derivative of Apache) is not supported by JBoss's *mod_cluster* plug-in for web server integration. We had to configure *mod_cluster* by adding configuration information to Apache HTTPD or by installing the JBoss Enterprise Web Server as base. We chose the latter.
  - EAP 6 clustering via *mod_cluster* requires two components out of the box, IP multicasting and security authentication. This is an improvement over the out of the box single requirement of IP multicasting in the previous version. *mod_cluster* also supports a TCP stack if multicasting is not desired.
  - An extra setup step is required for https out of the box – JBoss EAP 6 comes with no keystore/keys created.[19]
  - In our JBoss EAP 6 testing, we found the use of the *mod_cluster*-manager component is essential for verifying, diagnosing, and monitoring the web server configuration. This component must be configured using Apache's <Location> directive, which means that administrators should also be sure to secure the URL properly (i.e. restrict access to the URL from outside of the DMZ). The *mod_cluster* module still relies on the <Location> directive. Administrators need to manage the security of the associated URL.
  - WebSphere provides a plugin for a variety of web servers including Microsoft™ IIS, Apache HTTP server and IBM HTTP Server (IHS). We used IHS for our testing as, based on our experience, shops using WebSphere tend to also run IHS. We could also use the IBM installer to automatically install the web server and plug-in simultaneously to our server.
  - Configuration of the web server for use with WAS and WAS ND was straightforward and the initial integration of the web server with the app server was accomplished through the use of a script automatically generated by the installer.
  - The one issue with verifying that the application was deployed correctly was confusion surrounding WebSphere virtual hosts and ensuring that host names and web container ports are configured correctly in the Administration Console.

[19] https://access.redhat.com/knowledge/solutions/229963

Customers who have at least one WAS ND license can take advantage of the Job Manager functionality. This provides administrators with the ability to remotely install new installations or apply fix packs to a server via the Administration Console.

Using Job Manager, an administrator can install the deployment manager and at the same time create an initial repository. The administrator can then identify target servers on which to have WAS installed, either to be a federated standalone server or a clustered server. The administrator specifies the host name, OS type, user name, and password to use for the installation process. The administrator can also add additional WAS installation media for supported operating systems to the repository. For example, a repository installed on Windows can also contain installation media for Linux and Solaris. Once the administrator has provided the connection information and the appropriate responses to be used during the installation process, the process can be kicked off from the Administration Console and periodically checked for installation progress. Multiple nodes can be configured at the same time, using this functionality.

WebSphere fixpacks can also be deployed in a similar manner. An administrator first installs the fixpacks on the deployment manager, adds the update installer and fixpacks to the repository (or optionally downloads them using the Administration Console) and then selects which servers to install the fixpacks on. Job Manager automatically makes sure the target server has the correct version of the Update Installer installed and then proceeds to update the server without interaction or intervention required by the administrator. Our administrative scenarios for WAS include the use of Job Manager for managing WAS ND based environments.

JBoss EAP 6 does not have equivalent functionality for automatically installing servers in a remote or distributed manner.

Overall, we found that administrative tasks with WebSphere were simpler to execute and did not require highly skilled administrators to be successful. WebSphere's out-of-the-box settings are more consistent with the principles of least permission and least surprise. WebSphere's administrative console has been consistent between the last several releases with minor changes to the navigational structure of the console; experienced WebSphere administrators will be able to quickly adjust to these changes. Using publicly available support information, such as IBM InfoCenter and the built-in help for the Administration Console, administrators should be able to quickly install and configure standalone or clustered environments.

JBoss EAP 6 administration requires highly skilled administrators with a deep understanding of JBoss internals. Even experienced administrators have difficulty moving from version to version, as Red Hat has changed the configuration repository file structure significantly with each update.

Red Hat delivered a new administrative strategy with EAP6. Red Hat claims that all administrative functions can be performed either through the embedded Administration Console or the native management CLI. In our testing, we discovered this really means that the native management CLI is the fully functional client, while the embedded administrative console provides a basic subset of functions. The embedded console was useful for basic core functions and monitoring, but advanced configuration options needed to be set with the CLI. Additionally, we ran into an issue whereby incomplete content was mistakenly added with the CLI but could not subsequently be removed with the CLI. This required manual editing of the configuration file. JON will detect changes made by the embedded console, but there may be a time delay, resulting in a potential race condition.

Between the embedded JBoss console, JON, and the management CLI there is overlap of functionality. It is never clear which to use, and if a change is made in one place it isn't clear that it will show up in the others in a timely manner.

Incorrect and conflicting data was found in the published documentation as of the publish date of this paper. For example, the db2jcc4.jar driver is listed as the officially supported driver for DB2 9.7; however, the example for configuring a DB2 XA datasource uses db2jcc.jar, a supported driver. Furthermore, the DB2 XA datasource example in the documentation contained syntax errors and did not work. It does not

appear that Red Hat has tested JBoss with the DB2 database. We worked around issues like these by experimentation, consulting discussion forums, blogs, and wikis.

To summarize, WebSphere production environments are easier to install and configure than JBoss EAP environments, even though the WebSphere installation footprint is larger, while JBoss is easier and faster to install on a developer desktop.

Cluster Administration

TCO Advantage: IBM WebSphere Application Server

Our testing focused on analyzing the following conditions related to clustered operations of WebSphere and JBoss. Specific cases being analyzed were:

- Differences required in moving from a standalone configuration to a clustered configuration
- Ability to securely configure and administer clusters
- Evaluate the effort to add a new node to a cluster
- Configuration and handling of various failover scenarios related to HTTP session failover and JMS recovery operations

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

- Required Skills
    - Application server cluster configurations for JBoss EAP 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. It also requires a deeper understanding of JBoss internals including *mod_cluster*, *jbossweb*, and *JGroups* caching.
    - To move from a standalone configuration to domain configuration in JBoss is not a documented procedure. A manual procedure to do so would have to be developed based on needs. It would likely involve creating a custom profile in domain.xml based on the contents of the standalone server's xml configuration.
    - Creating a clustered WebSphere configuration can be accomplished by reading the available information in the InfoCenter and executing the recipes. A basic understanding of WebSphere architecture is required and can be gained from available documentation.
- Application Deployment
    - With WebSphere Application Server, application deployment in a clustered environment is very similar to that of a standalone application. The application is centrally deployed through the administration console, and the cluster is selected as the target instead of a standalone server. The changes are then automatically synchronized to nodes.
    - With JBoss WAP 6 for the farm-style cluster (which has no central deployment management), 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.
    - For the domain-style clusters in JBoss configuration changes are pushed down from the domain controller to the host-controller. This is similar to the WAS ND Deployment Manager pushing changes to App Servers via a Node Agent.
- Cluster Creation

  - For a farm-style cluster on EAP 6, starting up two copies of the 'ha' flavored config files will result in a cluster communications since they both will use default multicast settings. Some of the default clustered services, such as JMS, do require security information to be able to fully communicate in a cluster.
  - For a domain-style cluster, security information needs to be set for the host controller to receive a config file from the domain-controller. In this way, the domain-style cluster is more secure than the farm-style cluster.
  - Creating a cluster using WAS ND is a more deliberate process. In order for nodes to join a cluster they must explicitly be added to the cluster by an administrator. Unlike JBoss clusters, there is no guesswork or accidental creation of a cluster with an out-of-the-box installation.
  - WAS ND 8.5 also supports the creation of dynamic clusters that can expand or contract based on demand and service levels. There is no equivalent in EAP 6.

- Node Installation and Addition
  - We used WebSphere Installation Manager functionality for installing and patching our WebSphere cluster. More details about our experiences with IIM can be found later in this document. We found that this greatly reduced the amount of human time involved in the setup and configuration of application servers when using WAS ND (either clustered servers or federated standalone servers). This greatly simplified the cluster configuration process, because we had the option of adding the node to the server as part of the server installation process.
  - Adding a new node to a cluster in JBoss EAP 6 is the same as installing a new server. All steps must be manually completed via the command line. Most configuration files can be copied from one server to the next, but they may require modification to preserve each node's unique identifier.
  - Using JON in an EAP 6 installation allows central control of deployment and, in theory, patching, similar to IIM. This is new in EAP 6. However, The patches thus far available require manual installation outside of JON.
  - The process for a farm-style cluster is manual, but relatively simple, and could probably be scripted by an experienced administrator. On a co-located host, for example, a copy of an existing server is made, which can then be started up with a different port offset and unique node ID.
  - The process for a domain-style cluster is similarly manual and simple, and likely to be scripted by an experienced administrator.
  - Either process, unscripted, would likely take an experienced administrator, at most, a few minutes. In any environment other than development best practice would dictate a scripted solution, which would take considerably more time initially.

- HTTP Session Replication
  - Implementing session replication with JBoss was not possible with JON and required file access to change both the *web.xml* and the *jboss-web.xml* deployment descriptors. We also found that we had to have a more thorough understanding of the test application in order to select the appropriate synchronization parameters.
  - Session replication setup is relatively easy with the defaults, as only the <*distibutable/*> tag needs to be added to *web.xml*. However, changing values from the defaults for session replication can only be done by editing *jboss-web.xml*.

  - WebSphere supports several different flavors of HTTP session replication. These options range from session persistence in a database, to memory-to-memory using buddy groups, to memory-to-memory using active and passive failover. In addition, with the optional WebSphere eXtreme Scale (WXS), session replication can be managed across clusters and even across remote data centers. All of these, including WXS replication can be configured either through the administrative console or the command-line *wsadmin* tool and does not require any changes to deployment descriptors to enable session replication.
  - JBoss does not support HTTPSession replication via its own JBoss Data Grid product.
  - Results of our failure testing can be found later in this document including the situations encountered with JBoss that resulted in noticeable customer outages.

- JMS Failover with HornetQ
  - Both application servers require a senior level administrator to properly configure JMS for high availability operations.
  - The methods for message send and receive with HornetQ must be considered. The two options for HornetQ are in-vm and netty. When the client and server are co-located in the same JVM, the in-vm connector can be used. If the client and server run in different JVMs, then netty must be used.
  - WebSphere is less complicated to configure through the systems integration bus (SIBus) pages in the Administration Console.
  - JBoss required modifications to an XML file to properly configure clustered JMS. Direct configuration file editing can violate compliance standards.
  - By default, JBoss uses a multicasting approach for locating JMS servers and requires some care to properly separate nodes. In contrast, WebSphere uses explicit configuration to identify which servers should be part of an SI Bus.
  - JBoss Red Hat EAP 6 documentation on JMS failover is limited in the admin guide to a single sentence: High-availability failover is available with either automatic client failover or application-level failover. JMS clustering on/off switch is controlled via a property in the messaging subsystem and is enabled by default in the out-of-the-box 'ha' style config files. The public JBoss AS documentation, HornetQ documentation, and community sites had to be consulted to understand this better.
  - HornetQ does not have an option to persist queues to a database; thus there are scenarios where messages may be lost. WebSphere JMS has a persistent queue option.
  - Results of our failure testing can be found later in this document.

- Dynamic clustering
  - WAS ND 8.5 supports the concept of a dynamic cluster. This is a cluster that can be expanded at runtime by adding JVMs to the cluster or shutting down JVMs when demand for the application(s) running in the cluster have lower demand.
  - The expansion and contraction of the dynamic cluster is normally automatic, and is controlled by built-in load monitoring in accordance with user-defined Service Policies (e.g., response time must be less than 2 seconds), but can be overridden manually. The WAS ND On Demand Router will track performance against a set of SLAs on an application by application basis and adjust performance by starting or stopping instances of applications on servers
  - Intelligent Routing - not all requests are equal. If some applications are more critical (such as stock trading) they are given precedence over lower priority requests, such as Account

Management. Routing can also queue requests at peak times, rather than letting them time out.

- WAS Health Management monitors dynamic cluster health, and can take automatic or guided actions to restore health by restarting JVMs, holding requests, "storm drains" and other remedial actions that would normally require administrative recognition of the issue and manual intervention.
- Dynamic clusters support Edition Management, a process of rolling out application updates by taking offline only some of the JVMs in the cluster, deploying the update to those JVMs, then restarting them for either production or final test use. The new edition can be either moved to production or rolled back upon administrative command. Edition Management permits continuous availability and obviates the need for scheduled downtime.
- There is no similar capability to dynamic clustering in JBoss EAP 6.

In summary, Red Hat has simplified cluster configuration and corrected the propensity to create ad hoc clusters under default conditions; however, we still found the process of configuring JBoss to be manual and error prone, requiring significant labor to validate a proper configuration.

Our experience with WebSphere is that node installation and cluster creation is a relatively straightforward process. The skills needed to effectively manage standalone servers carry over to WAS ND through the use of a consistent administrative user interface. Additionally, out-of-the-box settings provide sensible defaults and eliminate guesswork when configuring environments of varying size and complexity. With dynamic clustering, IBM has an additional significant advantage for robust IT infrastructures.

Security Administration

TCO Advantage: IBM WebSphere Application Server

Securing Administrative Functions

This test case focused on the configuration of various security aspects of the administrative interfaces. WAS has always provided out-of-the-box administrative security. With JBoss it has been optional until the most recent release; EAP 6 now requires at installation that the server and administrative tools be secured with user ID and password. There is more to be considered, however:

- JBoss EAP 6 still does not include role-based access control (RBAC); WebSphere supports administrative roles and has for most previous versions.
- During installation, and under certain operational error conditions, EAP 6 may require an administrator to directly edit a configuration file. The administrator thus must have login privileges to the installation user (or to root) at the operating system level. WebSphere can be secured so that file level access is not required.

These are significant deficiencies for EAP 6, especially in regulated environments or those that require compliance with information security standards such as PCI, Sarbanes-Oxley or HIPAA.

LDAP Integration

Beyond standalone access, in some environments it is desirable to manage roles from a corporate LDAP repository. We therefore tested how this may be accomplished with WAS and with JBoss. 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 each platform. With WAS ND the process was straightforward and easily completed by an administrator. We had some difficulty configuring LDAP integration with JON, although we eventually did succeed. An experienced JBoss administrator could probably have configured LDAP about as quickly as we did for WAS, but an inexperienced administrator would have no difficulty configuring LDAP within WAS.

Cluster Configuration Security

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. Historically JBoss did not require security by default for cluster members, although it could optionally be incorporated. With EAP 6, clusters in JBoss now require credentials for a server to join. Some subsystems, such as messaging, require extra configuration steps to be clustered. However, not all those steps are security friendly. For example, there is unencrypted data in a cluster; the required password must be in plain text[20].

Security Patching

There are significant differences between JBoss and WAS managing security. In EAP 6, Red Hat offers *jar* replacements in which the administrator downloads the latest *jar* from JBoss and copies it into the environment. The customer has to keep track of the various jar and patch levels manually. In WAS, IBM requires the IBM Installation Manager (IIM) to be used in installing Security patches. IIM tracks and manages patch levels and can report on the patch level of each component. This helps tremendously when you have a problem and you are working with support trying to figure out what level of software you have installed.

It is yet to be determined how a full "fix pack" application process with work with EAP 6; as of this writing there have not been any released. This is thus an unknown risk.

General WAS Security Features

WAS ND has a number of additional security enhancements that are not available with JBoss EAP 6 or that can only be implemented with additional 3rd party tools:

- Full lifecycle management of key and certificate management covering all aspects from when the keys are generated, expired, or renewed.
- WAS can be configured to send emails to warn administrators their key/certs will be expiring.
- WAS is able to create both a WAS Personal Server and signing certificate.
- Support for multiple security configurations within the cell environment.
- Federation of LDAP servers and chaining multiple LDAP servers to present a virtual single view.
- Member Management capabilities.
- SPNEGO Support to provide single sign-on with Windows desktop.
- Kerberos Support.
- SAML Web SSO Post Binding Profile and SAML Web Services Token Profile Support.
- OAUTH support.

Secure Audit of Administrative Actions

TCO and Feature Advantage: IBM WebSphere Application Server

In many organizations, an audit trail of all system configuration changes must be maintained due to government, industry or securities regulations such as Sarbanes-Oxley, PCI, HIPAA etc.

Both JBoss and WAS keep their configurations in XML files and, as such, require access control and security auditing at the operating system level on the systems where they are installed. These controls do not track and audit changes made to the configuration files outside of the normal means of making

[20] https://access.redhat.com/knowledge/solutions/157913

configuration changes. They can show who modified a given file, but not what changes were made to the file. Thus, for proper security and auditability file level access for administrative actions should be restricted. WAS ND supports configuration management entirely through the Administrative Console and scripts, and provides comprehensive auditing tools for all administrative actions. JBoss, however, requires some configuration changes to be made directly to the XML configuration files, thus circumventing strict audit rules.

In addition, with JBoss there is no means to configure auditing of configuration changes made using the console or the command-line interface tool. Changes made using the console or command-line interface are stored in an XML history folder, but these must be compared to find out what changes were made and do not identify who made the change.

WebSphere Application Server can be configured to log all configuration changes made, which can then be shipped to auditors for review. WAS can audit all security relevant events, not only what has been changed in the Administrative Console.

There is a special "Auditor" role in WAS that allows these capabilities. 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 and 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 cannot modify auditing settings.

Deployment of Applications

TCO Advantage: IBM WebSphere Application Server

Secure 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 outcome:

- All JBoss configuration information is maintained in an array of XML files. Some files may be hot-deployed; others require a server restart. WAS and WAS ND also maintain server configuration as an array of XML files but expose access to them through both web-UI and command line-based administrative tools. Some changes require a restart of both application servers. With WAS ND, the web administrative user interface indicates when a server restart is required for each change.
- WAS and WAS ND support 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 Administration Console, the appropriate Jython commands can be accessed based on actions executed manually in the Administration Console. This can be used to generate scripts which can be used for additional environments or deployments.
- With EAP 6, a command line interface is now provided that can be used to manage all configuration changes. The CLI uses *bash*-style shell scripting, is easy to use and has prompting capability. However, if an error is made the configuration can be left in an indeterminate state. We made several inadvertent errors while configuring JBoss; in some cases it was necessary to edit XML files directly to recover from the problem introduced.

When using JBoss EAP 6, the implementation of audit trail generation is up to the user's script development. JBoss EAP environment files are typically backed with a source control system such as Subversion; wsadmin scripts can be managed as part of promotion procedures between environments. With WAS and WAS ND, changes resulting from script execution or actions of the Administration Console users are logged in the activity log file for the server.

Additional cost savings can be realized with the WebSphere Workload Deployer Appliance by automatically provisioning entire cluster configurations into the cloud of available resources managed by

hypervisors in a matter of minutes. These additional savings are not included in this study. There are no comparable offerings from the JBoss side.

Failover and Speed of Recovery

TCO Advantage: WAS ND and JBoss are equivalent

JBoss EAP 5 did not handle failover properly; it required administrative intervention to restart components. We ran tests to determine if this issue is resolved in EAP 6.

We co-located a JBoss vertical cluster, a web server and DB2 in a VM. The cluster was configured for messaging by default with the *standalone-full-ha.xml* configuration file; only the username and password for the cluster had to be added. We loaded the DayTrader application into this cluster, making sure the *<distributable/>* tag was set in *web.xml*, and validated its functionality.

We initiated a session by manually logging in as a user and doing some buy and sell tasks. We then brought down one member of the cluster via kill -9. When we went back to the application, we could continue to do buy/sell tasks. We then brought up the server that was down, and brought down the server that was up via kill -9. Back in the app, we could continue performing tasks in the same session. This seemed to validate that JBoss now handles failover correctly.

JMeter was then used to drive a constant throughput of 32 requests per second to the cluster. The first cluster member was then brought down. No errors were observed in JMeter and no unexpected errors were observed in either app server log file. The first cluster member was brought back up and the second cluster member was brought down. This time, in the logs, a duplicate order on the queue was detected and handled. Again, no errors were observed in the JMeter results.

In summary, JBoss's HTTP session appear to function as expected.

We repeated the tests with WebSphere and, as expected, HTTP session and JMS failover functioned correctly.

The HTTPSession and JMS failover tests confirm that when a JBoss app server in a cluster stops and then starts again, a corresponding restart of the web server is no longer required.

Server Monitoring and Administration

TCO Advantage: IBM WebSphere Application Server

JBoss EAP 6 has 3 separate tools for monitoring and administration. We encountered issues with all of them:

Embedded Console

- While the embedded console is very quick and responsive, its capabilities are also quite limited. There were a number of tasks we were unable to complete using the embedded console. For example, adding a new thread pool for use by the http subsystem, or creating an XA capable connection factory messaging destination.
- Deploying a known bad .ear file did not return control automatically to the user. Eventually, we noticed that a red colored error message would very briefly flash in the background. This message is very easy to miss, since the background enters into a 'faded' style look in comparison to the foreground (See below for more details).

JON

- We determined that we wanted to configure an executor for the out-of-the-box http connector of the web subsystem. We went through the process of adding a new ThreadPool child to the threads

subsystem, which reported back success. However, this never appeared in our configuration tree in JON, nor did it actually show up in the underlying configuration file. We ended up adding this manually – it could have also been added via the CLI. After adding it to the config file and restarting the server, the threads section of the JBoss configuration in JON remained empty even after a rediscovery of JON.

CLI

- In one instance, a component added using the CLI was unable to be modified or deleted, requiring manual intervention by the means of editing the files directly on the server to clean up the error.

As part of our standalone server testing we evaluated the embedded Administration Console, the JON console and the new CLI interface.

We assessed the deployment of an application via JON with both a working .ear file and a broken .ear file. The working .ear file deployed successfully with no unexpected results. The broken .ear file displayed signs of hanging and, after a few minutes of no response we inspected the log file. We discovered that the deployment had failed and rolled back in the log file. We tried to deploy a bad .ear file through 3 methods on JBoss: JON, embedded Administration Console, and CLI.

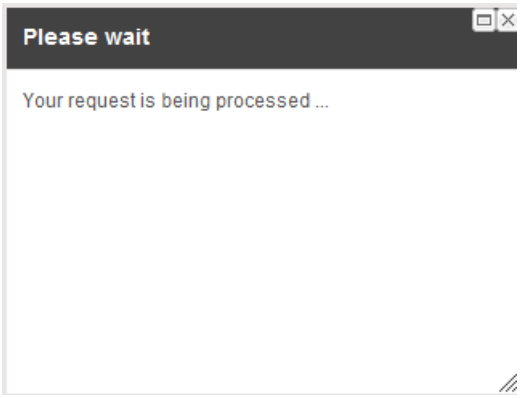
JON

1. Right click the EAP instance you want to deploy to -> Create Child -> Deployment
2. 3 step process: declare a package version for the resource, select one or more files to upload, give a runtime name and timeout value
3. Refresh workspace on left and deployment should appear; if not, run a manual discovery on the platform, which should turn it up
4. Right click the deployment -> operations -> enable, schedule the operation
5. Status will be returned for the operation, in this case - failure
6. Left clicking the status indicator icon displayed the reason for the failure

Embedded Console

Note that the embedded console cannot be used for an unmanaged cluster deployment, however, it can be used in a managed cluster.

1. In the runtime tab, select Manage Deployments on the left
2. Click the Add Content button, browse to the file to upload
3. Click the "Enable" label for that application
4. No indication of failure is ever given, just a *please wait* screen that hangs around until closed



CLI

The CLI can be used to manage and deploy to a cluster.

1. Login to the app server via jboss-cli
2. For an app that is already uploaded, run the following

```
read-operation-description   read-operation-names
[standalone@10.0.10.17:10099 /] /deployment=daytrader-jboss-bad.ear:deploy
```

3. This resulted in an indication of the operation failed that, like JON, prints the reason for the failure as reference

Thus, the results were different for each of the 3 means of deploying, and one did not work at all when deploying a defective .ear.

These issues do not exist when using WebSphere's Administration Console or administrative scripting client; there is only one tool that is used for deploying or any other management action. WebSphere does not exhibit the same behavior with failed application installations or updates. A failed installation is rolled back. If this is a new install, no traces of the failed installation were found. In the case of an application update, the failed installation was successfully rolled back, and the last successful deployment was still available and working properly.

Monitoring Capabilities

During performance tests, we used WebSphere's embedded Tivoli Performance Viewer and the WebSphere Performance Tuning Toolkit to evaluate app server health. For JBoss, we relied on basic statistics available in the embedded console and the monitoring features of JON.

- In WebSphere, the Tivoli Performance Viewer provided a way to quickly evaluate a number of PMI enabled statistics regarding pool usage and JVM heap size on the same screen. In JBoss, the embedded console provided some of the same statistics but across a number of different pages, which means significant additional time needs to be spent to navigate to all of those pages.
- The WebSphere Performance Tuning Toolkit connected to the app servers remotely via SOAP to get PMI statistics and display them in a friendlier manner than TPV. We preferred the layout of the eclipse based WPTT to the embedded TPV.
- Although JON has the capability to pull metrics from different sources into a dashboard style view, we found the resulting dashboard to be prone to different UI glitches depending on the browser used. For instance, with IE 9.0.8112.16421, the graphs did not display at all. With FF 16.0.1 and Chrome 22.0.1229.94, the real time graph updates had the timestamps re-draw on top of each other. If, say, a screenshot was needed to show someone, the timestamps on the x axis would be unreadable.

When the included On Demand Router (ODR) is installed with WAS ND 8.5, advanced monitoring capabilities are enabled, including automatic recovery from many errors. The ODR provides scriptable health monitoring and service level monitoring and recovery. JBoss, even with JON, has no equivalent capability.

Backwards Compatibility for Applications

TCO Advantage: IBM WebSphere Application Server

Beyond the server upgrade and its additional 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 the JBoss 6 release notes indicate multiple areas where backwards compatibility has been broken. When we installed the DayTrader 3.0 benchmark application it took several days and code changes to get the application first to install, then to run, in JBoss EAP 6. It ran almost immediately in WebSphere v 8.5, the only change required being a JNDI entry.

IBM provides a code migration plug-in for Eclipse and Rational Application Developer (RAD) that guides the developer through the process of identifying incompatibilities between earlier and current versions of the JEE standards. These are almost exclusively new reserved words with the retirement of deprecated functions in the Java standard. There is no similar tool for JBoss EAP 6.

Server Upgrade and Migration

TCO Advantage: IBM WebSphere Application Server

During the 5 year assessment, we assume one major version upgrade of the full environment. That is, at initiation of the timeline, WAS 7 and EAP 5 are installed. Then, in the 3rd year, those environments are incremented to WAS 8.5 and EAP 6, respectively.

We evaluated Red Hat's official 108 page migration guide document to consider a move from EAP 5 to EAP 6. This migration guide recommends two third-party tools to assist with migration – Tattletale and IronJacamar (which includes a migration tool). These automated tools help to partially offset the potential for making manual errors with required new xml configuration syntax. However, Red Hat does not offer any support for these tools. Though the two automated tools help to a degree, many manual xml changes are still required in the course of application migration.

We found that WAS, comparatively, results in a greater migration risk reduction, due to the inclusion of a supported full-function migration utility.

Furthermore, as the size of the environment increases, the JBoss manual migration path becomes more risk prone. In such cases, the more automated and supported IBM migration tool is clearly preferable to the more manual JBoss migration process.

One major issue when upgrading JBoss from EAP 5 to EAP 6 is the fact that the configuration data structure has been completely rewritten, so nothing is in the location it was with EAP 5. Thus, administrators will need to relearn the structure from scratch, requiring both time for enablement and time for incident correction when the inevitable errors creep in.

Also, admin and shell scripts written for prior versions of JBoss will no longer work and will need to be rebuilt. Time for this must also be added to the upgrade cost.

Patching

The process of applying a patch on either EAP 6 or WAS 8.5 should be similar, in that patches can be downloaded and applied using a separate tool to avoid issues with doing so manually. With EAP 6, JON has the ability to automatically apply patches to EAP 6 and other supported products when the patches become available in the Red Hat customer portal. This excludes patches that require a manual step. Unfortunately, so far, every patch available has required manual steps. This manual process was found to be error prone and cumbersome, as it sometimes required the merging of xml file settings.

By comparison, WAS 8.5 uses the Installation Manager tool to apply staged patches. If JON actually worked as advertised these approaches appear to be equivalent but, as of the date of publication of this paper, it does not appear to be the case.

Problem Determination and Troubleshooting

TCO Advantage: IBM WebSphere Application Server

Initial configuration of the test application took substantially longer on JBoss EAP 6. Time was initially spent trying to use the official EAP 6 documentation in which we found many inaccuracies. For example, the XA DB2 datasource example does not work as listed and uses a DB2 driver file not supported by Red

Hat. Eventually, we came to realize that better documentation existed on a sub-project basis for each component that goes into EAP 6. This decentralized nature makes it difficult to track down the right information. After we were able to correctly configure the application once, a fresh configuration took as long as configuring on WebSphere. In both cases, the configuration could be scripted.

JBoss datasources that are in an enabled state cannot be modified; they must be disabled first. To be able to disable a datasource, it must not be in use by any running applications. This adds extra time and effort that would be required to update a datasource's settings, as compared to WebSphere.

Action	Requires Restart?	
	WebSphere	JBoss
Change JDBC provider classpath	Y	Y
Adjusting JDBC data source pool size (cell-level)	Y	Y
Adjusting JDBC data source pool size (node or server-level)	N	Y
Adjusting JDBC data source settings (pool size, connection string, username, password, etc.)	N	Y
Adding a new JMS bus member or node	Y	Y
Changing JMS configuration	N	Y
Re-deploying an application after a failed deployment	N	N
Binding a web server to an application server	N	N

While trying to get the application configured correctly the first time for JBoss, tracing levels were modified to get more insight into what was happening. Every time a level needed to be changed, the app server needed to be restarted. This took more time than it would have with WebSphere, where trace levels can be changed dynamically at run time. This is important because in a critical troubleshooting situation, the ability to dynamically adjust trace levels at run time is less time consuming, and more feasible, than having to restart every time a minor trace adjustment is required. If there is a way to dynamically adjust the runtime trace levels for JBoss, it is not listed in the canonical Red Hat documentation.

WebSphere can do dynamic trace logging; we could find no way to do this with JBoss.

Serviceability and Documentation

TCO Advantage: IBM WebSphere Application Server

Logging, tracing and other diagnostics can at times be challenging, especially in complex environments. JBoss EAP does not provide a mechanism to access error information or log files outside of the file system. This is a potential security issue. WAS and WAS ND provide the ability to access log files without file system access. Administrative commands and the Admin Console provide trace analysis tools and error reports upon failure. It was our experience that the error reporting for WebSphere was more consistent, thorough and accessible than JBoss errors. Additionally WebSphere provides localized error messages in many different languages while JBoss does not.

Many customers told us that they really like the IBM JDK for its serviceability features. Here is an example comparing it to OpenJDK and HotSpot:

Capability	IBM J9	Hot Spot, OpenJDK
Fast garbage collection for large heap sizes (>4GB) - -Xgcpolicy:balanced	Yes	No
System class data sharing for reduced memory footprint and faster startup	Yes	Client only
Application class data sharing for smaller memory footprint and faster startup	Yes	No
Avoid JVM restarts due to the PermGen filling up	Yes	No

Capability	IBM J9	Hot Spot, OpenJDK
Compressed 64-bit references (for faster runtime and smaller memory)	Yes	Recent
Dump analyzer - hang, crash, memory management	Better	Good
Garbage Collection and Memory Visualizer - memory usage and performance	Better	No
Memory Analyzer – troubleshoot memory leaks and excessive heap consumption	Yes	Yes
Health Center – real time monitoring of running virtual machines	Yes	Partial*

*Jconsole provides some of these capabilities

Throughout our testing, we found many JBoss errors that would cause the server to fail to start. We also found that documentation became increasingly unreliable when being applied to complex configurations or production-level configuration and security hardening. In our own testing we noticed that performing root cause analysis for some JBoss issues was difficult due to the number of configuration changes required to solve issues due to the time and effort to painstakingly track and validate each change made.

We found that many JBoss forum postings and wiki articles provided conflicting information. In some cases, this was due to the long release cycle for JBoss EAP 6 and the number of defects that were opened and closed during the development process. Unfortunately, when verifying whether these proposed solutions would alleviate our issues, we generally either uncovered additional issues or found that the solution did not meet our needs.

A general trend we observed with the JBoss EAP 6 documentation was its slant towards developers versus administrators. For example, many of the quick start guides were geared at quickly setting up a single/local development environment as opposed to configuring servers for use in multi-server environments.

We were hampered by a number of outright errors in documentation. Examples include:

- The web page with the official supported configurations for EAP 6 lists this as the DB2 supported JDBC driver https://access.redhat.com/knowledge/articles/111663

  The following databases and database drivers were tested as part of the latest JBoss EAP 6 release.

Databases	JDBC Driver
IBM DB2 9.7	IBM DB2 JDBC Universal Driver Architecture 4

  However, as of JDBC 4.0 support, IBM no longer refers to it as "IBM DB2 JDBC Universal Driver Architecture", its now called "IBM Data Server Driver for JDBC and SQLJ" http://pic.dhe.ibm.com/infocenter/db2luw/v9r7/index.jsp?topic=%2Fcom.ibm.db2.luw.qb.upgrade.doc%2Fdoc%2Fc0023412.html

  This may be a minor consideration, as the "4" designation appears to indicate that db2jcc4.jar is the supported driver, and not the db2jcc.jar driver.

- Contradicting the supported configuration page, in the official documentation for configuring a db2 driver, db2jcc.jar is used for driver the configuration. https://access.redhat.com/knowledge/docs/en-US/JBoss_Enterprise_Application_Platform/6/html-single/Administration_and_Configuration_Guide/index.html#Example_IBM_DB2_Datasource

  This is just another example of the inconsistency of the "official" documentation. Trying to track down the supported XA jdbc configuration has been a challenge.

- Finally, on several occasions the Red Hat support site was unavailable at key times. Some of these were announced, but others occurred randomly. We find it interesting that some of the scheduled downtime for maintenance occurred during the business day, rather than after hours or on

weekends (one scheduled day was a Saturday, but another was a Wednesday). These outages delayed resolution of some issues that affected our testing.

Database Integration and Problem Diagnosis

TCO Advantage: IBM WebSphere Application Server

WAS has excellent database integration items that help in problem determination and failover. Examples include:

- Connection Annotation - trace back the database issue to the offending WAS application (e.g., bad SQL query). Supported with DB2 and Oracle
- Dynamic JDBC trace enablement and integration within WAS; ability to get traces for both WAS and database (integrated together in one file) and dynamically enable and disable the trace from WAS
- WebSphere integration with database failover solution
    - DB2 HADR further integration by taking advantage of the "Automatic Client Reroute ACR" from DB2
    - Ability to automatically or manually (without server restart) switch between datasources (supports all relational backend databases)
    - Oracle RAC integration: use RAC specific features such as ONS, FAF, through the enablement of Oracle Universal Pool in WAS
- Client Identity Propagation (DB2)
    - Complete accountability – you know who did what in the database from the client side of things
    - Improve database security – you will no longer need to give one user in the application server access to all the tables in the DB but instead, allow each user (or group of users) specific access
- Support for Heterogeneous pooling – configure one connection pool but allow it to contain different connection properties. Eliminates the proliferation of connection pools and hence reduces the database load/usage (CPU and memory) by allowing connection usage concentration.
- Support and ease of configuration and deployment for things like SQLj and PureQuery which support static SQL and can provide performance and security benefits

These features, for the most part, are not available with JBoss EAP 6.

Management of Large Topologies

TCO Advantage: IBM WebSphere Application Server

WAS has a unique capability called Flexible Management[21] 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

[21] http://publib.boulder.ibm.com/infocenter/wasinfo/v7r0/index.jsp?topic=/com.ibm.websphere.nd.doc/info/ae/ae/cagt_jobmanager.html

- Specify that the job submission occur at a specified time interval
- Notify the administrator through e-mail that the job has completed

JBoss EAP 6 does not have comparable functionality, although JON has some task and script scheduling capability. This functionality could reduce off-hours work required by administrators or be used to avoid potentially expensive site visits.

While this study focused on the Medium and Large scenarios described above, here are several Very Large scenarios and hypothetical company examples of situations where a Job Manager would be useful.

- 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.

Based on the findings in our hands-on testing, managing JBoss EAP 6 in these kinds of environments may become very difficult without building a custom home grown management framework similar to what is discussed above.

Training Costs

TCO Advantage: IBM WebSphere Application Server

The available training from IBM for WAS v8.5 and Red Hat for JBoss EAP 6 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 again half-way through the five-year period to account for turnover, growth or major upgrades that may occur.

The quantity and skill level of administrators is based on the results of the hands-on analysis. For the purpose of this study we used our study team size (three administrators). It is important to note that for larger JBoss environments, training costs could be higher due to the performance difference and the associated increase in the number of servers required.

One of our clients researched the availability of JBoss administrators, and discovered that experienced administrators (and developers) are harder to find than for WebSphere. They conducted a developer web

study that found that 65% said they use IBM development tools while only 1 person has used JBoss. Thus, we also need to include higher costs for training new and current staff on JBoss than on WebSphere.

In our study, the amount of training costs being factored into the TCO calculation over a five-year period is $84,375 for WAS ND training versus $171,998 for JBoss EAP, a difference of 204%. Note, however, that this is not a significant contribution to the overall TCO for the size of systems considered for this study.

Costs not Included in the Study

Our detailed testing described above attempts to cover the major quantifiable aspects of TCO that an organization may encounter in its evaluation of application servers. However, it does not cover all possible factors that affect TCO. Through our testing we uncovered a number of qualitative factors that may have some impact on TCO and the application server vendor selection process. How these factors are measured and affect the TCO for two different organizations may vary widely, therefore we did not include these findings below in the formulas used to calculate the TCO results. It is feasible that, for certain organizations, these additional considerations described below, when taken together, may well exceed the cost considerations discussed in the previous sections. Organizations that require high quality services from their infrastructure should do a cost/benefit analysis of the capabilities discussed below.

Software Development Costs

A commonly held belief is that open source solutions more quickly integrate new standards and functionality. Over the past three years of conducting this study, we have found that this is no longer the case. In fact, with regards to WebSphere and JBoss, this appears to have reversed. With WebSphere 6.1, IBM introduced the concept of a pluggable architecture via the use of free feature packs. WebSphere 8.5 expands this concept with the Liberty Profile, a lightweight package that can be set up and deployed in minutes.

In addition to the programming models recommended by IBM, WAS is flexible in the choice of programming models. For example, the Spring framework is supported on Liberty Profile as well as WAS base and WAS ND, and WAS is flexible to allow applications to use 3rd party libraries even if they are not certified with WAS. There are some instances where a classloader conflict might exist and special care might be taken but, with the introduction of the Liberty Profile, a complete isolation now exists to help in those cases.

WebSphere 8.5 Liberty Profile

WebSphere version 8.5 is available with a scaled back, incrementally expandable package that meets the need for quick installation, setup and launch. We evaluated the Liberty Profile from a developer's perspective.

The Liberty Profile is a new dynamic profile for WebSphere Application Server (WAS) v8.5, which enables rapid development and deployment of web applications in a simple and lightweight manner. The Liberty Profile provides fidelity to the WAS full profile, enabling applications to be developed and tested against the WAS V8.5 Liberty Profile and deployed in production against the full profile of WAS (including WAS 8.0) without any runtime behavioral differences. Applications may also be deployed to a WAS V8.5 Liberty Profile server for production with performance similar to that of the full profile WAS server.

The Liberty architecture is inherently composable; users can configure at a fine-grained level the components needed by their applications, so the server is very fast to start and the footprint remains minimal. Features that can be optionally configured include a number of Java Enterprise services (Servlet, JSP, JSF for example) as well as security services and OSGi application support.

We verified that Liberty Profile is lightweight and easy to install. At under 50 MB, it is smaller than the lightest version of JBoss AS, and it installs via an unzip. Alternatively, it takes only a few minutes to install from Eclipse, and the installation is automatic. Adding utility projects is also easy; just drop them into the server.

We found the start and stop times almost instantaneous (around 2.5 seconds), a great help during development and local test cycles. Configuration is also easy and quick using either Rational Application Developer (RAD) or the free WAS Developer Tools for Eclipse (WDT) plug-ins. This support includes UI

tools for configuring server settings to provide syntax assist or form based editing of configuration files, including:[22]

- Dynamic updates: changes to the server are picked up instantaneously and no server restart is required.; another plus for saving development time
- Easy deployment – just drop the application EAR in the dropins folder and it will be picked up by the server
- The server startup time and footprint for the TradeLite benchmark web app is slightly faster and smaller than JBoss (WAS = 2.3 seconds to start; 47MB footprint; JBoss 4 seconds to start, 110 MB footprint) with 20% higher throughput
- Right-sized app server based on the needs of the application at a much more fine-grained level than simply web profile versus full JEE; JBoss AS is not that composable
- Rich set of Eclipse-based tools available for free from Eclipse Marketplace or ibm.com, integrated with WAS, provides a subset of RAD capability for Java EE app development for no usage fee (support subscription available)
- Simplest possible server configuration - treat the server configuration as a development artefact just like the application source code (for version control, diffing, team development); it's simpler than JBoss AS
- Easy access for developers (free, lightweight download - no install program; unzip and go) - an area where WAS was previously at a disadvantage but has caught up on
- Dynamic Server profile determined by the app at a fine-grained level.
- Dynamic runtime – Add features and update configurations without having to restart the server
- Includes Mac as a development platform
- Supported by WAS ND Job Manager
- Fidelity to full profile WAS for easy migration from development to production
- Ability to create a packaged server of server+application+configuration for unzip deployment of a complete solution, something that is simple to integrate into third-party continuous delivery frameworks (just need zip/unzip) and has optional centralized installation of WAS ND Job Manager; there is nothing equivalent in JBoss EAP 6.

A useful link that we found for configuration and understanding of WAS V8.5 Liberty Profile:

- WAS V8.5 Liberty Profile Quick Start Guide

  http://www.redbooks.ibm.com/redpieces/abstracts/sg248076.html?Open

Programming model Extensions

The WebSphere Application Server Web 2.0 and Mobile Feature Pack helps developers create more interactive desktop and mobile applications. Standards-based technologies, including Ajax, REST Web services, and Dojo, are used to simplify and speed the addition of rich desktop and mobile user interfaces to WebSphere Application Server applications. A large collection of core services are also included, including new skins for a variety of mobile platforms (Blackberry, iPhone, Android, etc.), and innovative user interface widgets to improve the user experience.

- WebSphere Application Server includes "out of the box" support for SIP (Session Initiation Protocol).
- WebSphere Application Server also provides "out of the box" support for the Portlet API (JSR 268).

[22] https://www.ibm.com/developerworks/mydeveloperworks/blogs/wasdev/entry/getting_started_with_the_was_liberty_profile?lang=en

- In WAS v7 and v8 he WebSphere Application Server Feature Pack for Communications Enabled Applications (CEA) helps developers add co-browsing, and communications capabilities to their applications without requiring detailed knowledge of SIP. JBoss EAP 6 provides no similar capability.
- The WebSphere Application Server Feature Pack for Dynamic Scripting (which used to be a separate sMash product) can help enterprise IT to address situational application needs by providing a Web-oriented programming model focused on agility through the use of Web 2.0 and dynamic scripting technologies. The Feature Pack for Dynamic Scripting delivers the PHP and Groovy dynamic script languages and an innovative Web 2.0 programming model (AJAX, REST, Atom, JSON, and RSS). JBoss EAP 6 does not provide similar capability. Those customers who wish to have support for their scripting applications on JBoss will have to obtain and integrate 3rd party products. This was a feature of V7 and V8.
- Also in V7 and V8, WebSphere Application Server Feature Pack for XML delivers critical technology that enables adoption of key XML standards and principles, including XSLT 2.0, XPath 2.0, XQuery 1.0. These new and updated W3C XML standards offer application developers numerous advanced capabilities for building XML applications. Some of the benefits delivered in these standards include simpler XML application development and improved developer productivity, improved ability to query large amounts of data stored in XML outside of a database with XQuery 1.0, improved XML-application performance through new features, improved XML-application reliability with new support for XML schema-aware processing and validation. JBoss supports older version of the XSLT and XPath with all of the ease of use and performance issues associated with those older less mature and less flexible specifications.

Thus, IBM has enhanced the core WAS JEE6 runtime with a number of capabilities built into the WebSphere Application Server at no additional cost to its customers.

Mainframe Support

WebSphere 8.5 is available natively for IBM z/OS, as well as being supported on IBM zSeries under z/Linux. For more information see:

http://www-03.ibm.com/support/techdocs/atsmastr.nsf/WebIndex/WP102110

No other JEE application server will run natively on z/OS on a mainframe.

Quality and Availability of Vendor Support

Quality of vendor support can have a direct impact on the cost of operations.

- IBM provides local language support in many countries in local hours. IBM offers several levels of support ranging from 9x5 (local time) through 24x7 support. In addition to paid support, IBM offers the InfoCenter. Over the past few years the quality and accuracy of the documentation available in InfoCenter has improved. Finally, IBM has settled on a quarterly release schedule for WebSphere Application Server fix packs and interim and hot fixes are still made available on an as needed basis.
- Red Hat provides support options of either 12x5 or 24x7, but for many locations across the world there is no support in native language. Documentation is available online, however, as has been noted throughout this paper, it is greatly lacking. With regards to fix packs, it appears that Red Hat has not settled upon a fixed period for maintenance releases and hot fixes are released as needed.
- Red Hat limits the number of contacts who may request support, based on the number of cores purchased[23]. IBM has no such limitations.

[23] https://access.redhat.com/support/offerings/production/contacts.html

- We found that frequently the best support for JBoss came not from Red Hat directly, but from the user forums. Numbers of dedicated users were especially helpful in resolving issues for us, but this does not replace direct vendor support as users cannot fix defects and the response is not predictable when dealing with critical situations.
- IBM's DeveloperWorks user forums are useful in the same way, but IBM's dedicated support structure is generally more responsive than Red Hat's.
- By way of contrast, we had a Fortune 100 client recently whose public consumer portal went down late on a Friday evening. When routine restarts failed to resolve the problem IBM dispatched a team of 20 specialists (include 3 from Prolifics) over the weekend to first resolve the issue for the moment, then complete a root cause analysis so steps could be taken to prevent a recurrence.

Conclusions

Comparisons of WebSphere and open source solutions frequently focus on licensing costs and the application development point of view alone, only looking at feature sets and initial acquisition costs. Unfortunately, the long-term administration costs associated with JBoss or Tomcat are not factored into the equation. Too often, the "free" initial costs lead developers to favor JBoss. Our experience and the analysis from this study reveal that there are many more factors for organizations to consider in their selection process beyond initial acquisition costs.

The experience of our clients and our own experiences indicate that as the size and complexity of a deployment and the supporting organization grows, costs shift from product acquisition to administration and operational activities. It is only in smaller and less demanding applications that initial licensing costs are a significant part of the IT budget, and even in these cases there are fully supported products that can still keep acquisition costs reasonable. For example, WebSphere Developer Edition is free, and WebSphere Express and Base are fully supported products that are priced within any IT budget.

Our evaluation of WebSphere ND 8.5 versus JBoss EAP 6 shows significant differences since the last time a comparison was done in 2010. On the WebSphere side:

- IBM has standardized across all WebSphere products with the Installation Manager (IIM), which provides a common user interface and scripting capability. Using IIM installing any IBM product, including WAS and WAS ND, is simply a matter of checking off options in the user interface and clicking go. IIM will download all required components, then apply any fix packs or patches.
- IBM's J9 JVM sets new standards for performance and manageability.
- IBM now includes Intelligent Management in WAS ND, providing dynamic clustering, extensive real time monitoring of health and service levels, and application updates with no downtime.
- WAS 8.5 Liberty Profile is a lightweight fully WebSphere compatible development and production environment that can be installed in minutes and can have an application up and running immediately. Liberty Profile is a strong alternative to open source servers when paired with WAS Express edition. The simplicity of Liberty Profile can significantly reduce application development time for applications that are later to be deployed on the full profile WAS ND.

JBoss EAP 6 has improved in several areas, notably:

- The addition of a full-featured command line configuration tool that largely eliminates the need to edit configuration files (which is fortunate, because Red Hat has completely changed the file structures of the configuration directories).
- JON is much more complete and provides better control and patch management.
- Earlier problems with HTTP Server restarts and *mod_cluster* instability have been resolved.
- Application server security has been improved
- Installation in the development environment, which has always been a strong point of JBoss, is still simple and quick.

Despite these improvements, JBoss EAP 6 lacks critical functionality in areas such as administration, backwards compatibility, transactional integrity, security, auditability and cluster reliability. Further, with Intelligent Management, WAS ND has capabilities that are totally absent from JBoss EAP 6. When we examined other factors including system administration, software maintenance, ease of integration and development costs we found that the products take very different approaches that significantly impact TCO. JBoss EAP 6 also has significant shortcomings in the areas of available support and reliable documentation that hinder development and system administration. This was true for JBoss EAP 5 and has not improved significantly for EAP 6. Documentation is still incomplete, inaccurate and sometimes contradictory. On the

other hand, WAS 8.5 continues to be stable and build upon the lessons learned from previous versions of WAS.

One of the significant findings in 2010 was that there was a serious issue in EAP regarding transaction integrity when multiple data sources needed to be updated as an atomic unit (XA transaction). If there was a failure of an update to one data source others would still complete, leaving the data sources in an inconsistent state. This bug was well documented on the Red Hat support site and was never resolved for EAP 5. We were surprised to find that EAP 6 has a similar issue that in some situations is even more serious, as it leaves the impression that the failed update succeeded. Based on these tests and the fact that this issue is still not resolved we must advise against using JBoss in any application that depends on XA transactions to maintain data integrity.

In both EAP5 and now EAP6 JBoss has inadequate controls on auditing of administrative actions that put the product out of compliance with many government and industry standards.

An important area to address is the need to update or upgrade environments periodically. IBM has committed resources to ensuring a migration path from previous versions to the latest versions of WAS and providing migration tools to assist in the process. The WebSphere Configuration Migration Tool enables easy migration of configurations from earlier versions going back to WAS 6.0 to WAS 8.5. The Application Migration Tool will assist developers in upgrading code from earlier versions to the latest Java standards, as well as migrating applications from other application servers such as WebLogic, JBoss or Tomcat to WebSphere. Red Hat has no official statement regarding support for backwards compatibility and, in some cases, applications that worked in JBoss EAP 4 or 5 will no longer work in JBoss EAP 6. JBoss EAP 6, being a major rewrite, has issues migrating configurations from earlier versions to EAP 6. Red Hat's only action to address these issues is a 108 page configuration migration guide.

What we have demonstrated again is that while in some scenarios WebSphere Application Server may have higher initial acquisition costs, over time it makes up the difference by having lower administration costs, better performance, better reliability, and less overall risk. Further, there are instances where JBoss fails to meet security, compliance and data integrity standards. For these instances JBoss EAP 6 is not an option at any price.

About this Study

About this Study

The study was commissioned by IBM and performed by Prolifics, a SemanticSpace Company. All findings of this study are fully backed by the solid reputation of Prolifics. For over 34 years, Prolifics has had a single mission—solving customers' business problems with technology solutions by providing expert consulting services and high-value technology enablers. Founded in 1978 in New York City, Prolifics first specialized in building high-performance transactional systems for many of the biggest firms on Wall Street. Today with offices throughout North America and Europe, Prolifics utilizes its wealth of expertise to offer comprehensive business solutions, having worked with over 3000 companies, both domestically and internationally, across a broad range of industries including banking, insurance, government, telecommunications, manufacturing and brokerage. In 2008, Prolifics joined with SemanticSpace Technologies and Arsin Corporation to form SemanticSpace Group – a global organization offering clients a "blended-shore" delivery model with a componentization strategy for building custom applications.

One of the keys to our longevity is the solid relationships we have built with our partners and customers. Due to the completeness of their e-business vision and robust technology, our most strategic alliance today is with IBM. This alliance enables us to combine Prolifics' expertise and long history of delivery success with IBM's solid vision and market leadership in order to bring our customers real-world solutions for their business needs.

In fact, IBM selected Prolifics as 1 of only 3 WebSphere Service Providers to be retained to service its own key customers. Selected for its consulting excellence, Prolifics has a Level 5 ranking—IBM's highest on its technical expertise scale—and is the recipient of the IBM Award for Overall Technical Excellence. Prolifics has also been honored with the IBM Business Partner Leadership Award, the 5-Star Partner Award, the Best Portal Solution Award and the IMPACT SOA Solution Award.

Combining business flexibility and cost competitiveness with excellence in technical skill, professionalism and a commitment to our customers, Prolifics' solutions build agility into businesses around the world. Our depth of experience delivering secure, end-to-end SOA and Portal solutions can help you with all aspects of the full project lifecycle including software procurement, architectural advisement, analysis, design, development, testing, training, support and system administration.

.

About the team that wrote this paper

The work behind this paper was performed by a team of expert JEE developers and administrators at Prolifics. We pulled together resources across disciplines, including a JEE architect and application migration practice director (Larry Finch), an Infrastructure lead (Leland Irwin), Automation and Monitoring specialist (Steve O'Neill), development and migration specialist (Hiranmayi Bellumkonda), and two support personnel (Stephen DeVito and James Gibson). The team has combined experience of over 90 years of middleware development and management. We intentionally chose a team with a wide range of skills, from broad experience to relative newcomers, to provide a balanced analysis of the products and services reviewed.

Larry Finch is a Vice President of Prolifics and a Senior Enterprise Architect with over 20 years of experience specializing in project management, enterprise systems architecture, Service Oriented Architecture (SOA) and Enterprise Service Bus (ESB) patterns, business process analysis and reengineering, systems analysis, detail design, implementation, and ongoing support. His current responsibilities include managing teams of specialists in migrating applications from other platforms to IBM WebSphere family products. Under his direction, Prolifics teams have migrated over 1,000 Oracle, BEA, JBoss and other applications to IBM WebSphere for 140 different customers.

Leland Irwin is a senior consultant with more than 20 years experience in the IT industry with a breadth of experience spanning multiple technologies and platforms in a variety of roles. He is experienced in all aspects of the development life cycle, including estimating, analysis, design, development, testing, release management, and bug fixing. His current areas of expertise are administration and support of WebSphere Application Server environments, WebSphere Virtual Enterprise, and migration to WebSphere Application Server v8.5.

Steven O'Neill specializes in solving Application Performance Monitoring and Service Management business problems using IBM's suite of Tivoli products. He has a background in WebSphere Administration and Java EE development, over ten years of industry experience in the field of Software Systems, Middleware and Enterprise Integration Systems.

Hiranmayi Bellumkonda is a JEE Enterprise Application Developer and Designer with 10 years of experience. Her core skills include strong design capabilities and thorough understanding of JEE Design Patterns, specialization in developing backend business objects for enterprise applications, experience in all phases of software development (SDLC), and extensive experience migration applications between Java Application Servers and upgrading applications from earlier to later versions of products and JEE specification standards. She is a Java Certified Professional with in-depth knowledge of Object Oriented Analysis and Design of Software Systems employing Industry proven software engineering best practices.

James ("Ox") Gibson has over 17 years of professional technical experience in the high tech business sector, including 7 years as a Linux administrator in the United States Army. Ox's primary skills are in Linux administration, network management and application deployment and management.

Stephen DeVito is an associate consultant with Prolifics. His mathematical and analytical background grants him large insight into areas such as statistics, analytics, problem-solving and makes him a valuable resource in analyzing our test results.

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 EAP 6, WAS Express, WAS Base 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 three categories.

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 projects or 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, require more frequent administration, but do not need clustering.

Category three - Large is focused on the large middle ground of web-application environments. These environments have a range of ten to forty servers; often 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

Very Large environments are an extension of Category three and are always 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.

A summary of the parameters for Large Topology used in the TCO analysis is shown below:

Applications	
Years for TCO Analysis	5
Number of JEE applications	30
Number of new applications developed per year	6
Number of re-deployments per year per application	6
Number of datasources per application	2
Number of JMS topics/queues per application	2
Number of other integration points per application	2
Deployment configuration	
Number of J2EE cluster units in organization (cells)	4
Number of non-clustered standalone servers	6
Number of clustered JEE servers	18
Number of web servers (HTTP Servers)	5
Number of Dynamic and static cache servers	2
Number of LDAP servers	2
Number of IP sprayers (WLM)	2
Transaction rate growth per year	5%
Hardware configuration	
Hardware server for Application Server - baseline	$ 14,000
Other server options (memory, etc.)	$ 7,000
Cost per processor (sockets) per physical server	$ 1,900
Hardware Support Cost (percent of purchase)	18%
Number of processors (sockets) per physical server	2
Number of cores per socket	8
Number of cores per physical server	16
Number of WAS PVUs per core	70
Cost per server	$ 22,900
Non-application server cost (LDAP server, HTTP, etc.)	$ 6,000
Number of processors (sockets) per physical server	2
Power and cooling cost as % of the HW cost	70%
Staff costs (fully burdened cost)	
Yearly salary base	$ 90,000
Load factor	55%
Development	
For how many years the development team is involved	5
Number of Developers	2
Number of Lead Developers	1
Business costs	
Company revenue in a year ($M)	$ 600
Project impact on the company revenue	1.00%
Discounts	
IBM software discount	35%
JBoss software discount	35%

The bulk of the analysis in this paper is focused on factors that impact Medium and Large environments. This focus was chosen for several reasons. Environment category one is frequently assumed to be a target for OSS, yet depending on the level of support purchased from the vendor WAS Liberty Profile on WAS Express would likely be more cost effective than JBoss.

- A list of TCO factors for application server use was established against common architectural, deployment and operational requirements.
- From this input, the three high-level usage scenarios were established for comparison purposes.

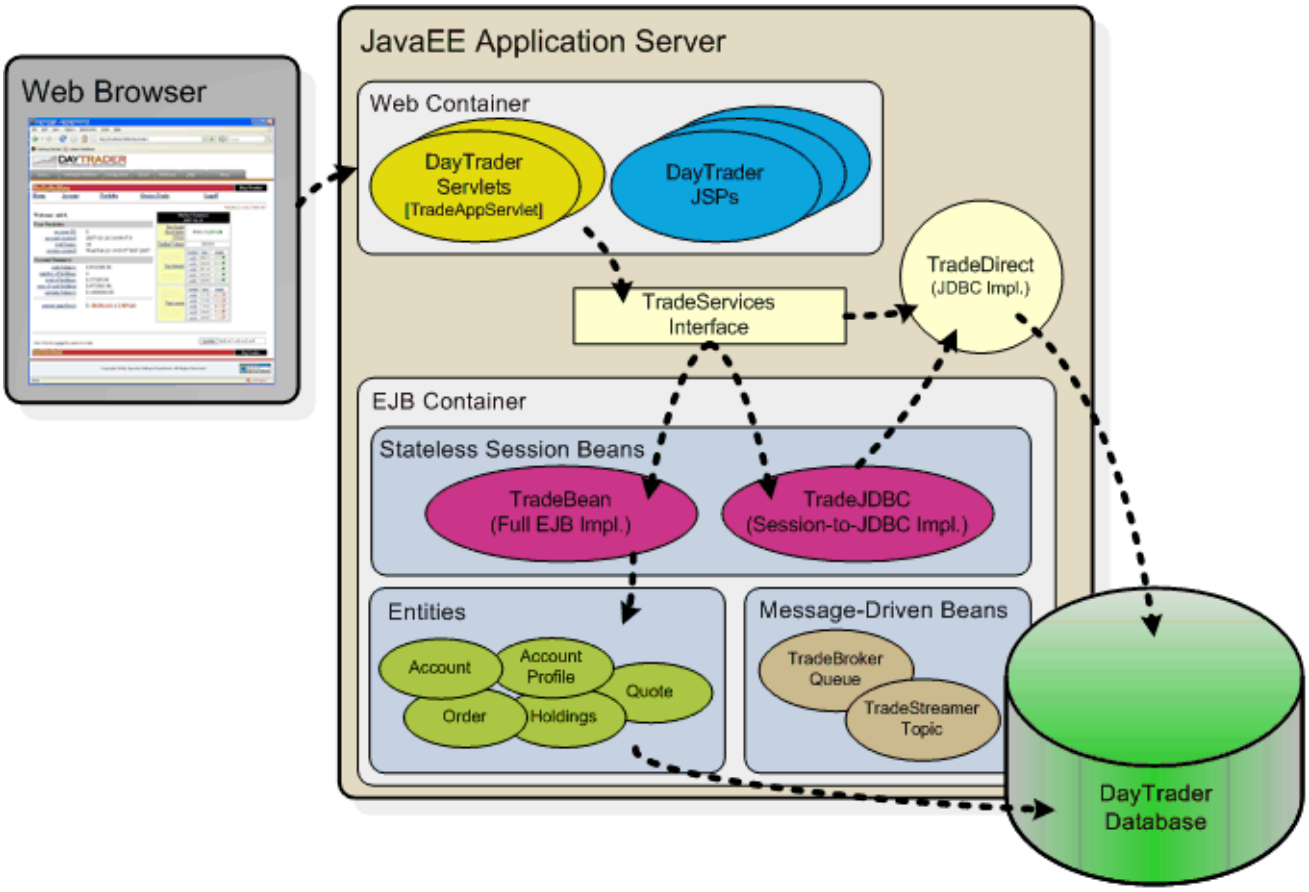
- Following the scenario and environment definitions, a set of test cases were developed and reviewed. The test cases defined configuration requirements, goals and scripts for the set of actions to be performed and measured against real-world issues and system configuration requirements for each product. 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
- 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[24] 2.1.7 application was used. Reasons that the DayTrader application was selected include:

- Use of a significant set of JEE6 services, including JDBC, JMS, EJBs, Servlets/JSPs, JTS/JTA
- Enterprise-grade services requirements including security, performance, clustering and transactions
- Support of multiple databases and JMS environments
- Performance in a clustered application server environment
- Migration path from earlier JEE application versions to JEE6

[24] https://cwiki.apache.org/GMOxDOC20/daytrader.html



DayTrader does not include web services capabilities. To test web services performance we developed our own application, which used moderately complex web service implemented as a JAX-WS 2.0 annotated POJO hosted by the application server. The development was done bottom up - first main POJO and all dependent POJO classes were developed and then main POJO class was annotated with @WebService. The business method used some of the input data to generate semi-random output. No backend services were called during the execution of this web service. The data classes used in the web service are representative of a real world application commonly found in financial sector. In total, there are 30 inter-related POJOs that together comprise the payload of the input and output logic for the web service. Combined, these 30 classes have over 150 primitive data fields - almost every primitive data type used in Java. Certain data types are arranged as arrays and appear in the input XML payload multiple times. The input workload for the web service performance test included SOAP/HTTP requests varied in sizes and included requests and responses of 15K and 110K.

Appendix C – Test Environment

To perform the analysis, a diverse and realistic environment was established for testing. The components of the test environment were selected based on common combinations encountered in real-world environments.

The technical environments for testing used the same hardware and operating system configuration for both environments to ensure that environmental differences were not a factor in TCO measurements. WAS ND, JBoss EAP and JON versions were initially selected based on the most recent production and patch levels generally available as of July 2012 and updated as late as September, 2012. The components of the test environment were selected based on common combinations in real-world IT operating environments.

We configured a set of five separate virtual machines (*wputil, wpapp1, wpapp2, wpweb1, wpdb1*) hosted under VMware® Server running on Red Hat® Enterprise Linux® environments. These were replicated on other servers as needed. One set of virtual machines was designated for performance testing, and all other environments were shut down during performance testing to prevent interference from other images. Each virtual machine had the following configuration:

- Red Hat Enterprise Linux host servers running RHEL 6.2 64 bit running:
  - X86 architecture
  - 4 GB RAM
  - 40 GB storage
  - IBM WebSphere Application Server Network Deployment v8.5 fixpack 1 (8.5.01)
  - JBoss EAP 6.0.0-4 (Sept 3, 2012)

Each of the server instances was configured differently:

- *wputil*
  - 4 cores
  - WebSphere 8.5 fixpack 1 (8.5.01)
  - Deployment Manager
  - JON Server
  - JON Agent
  - Other
    - Tivoli Directory Server (LDAP repository)
    - Testing utilities (JMeter, etc)
    - PostgreSQL (used by JON server)
- *wpweb1*
  - 4 cores
  - WebSphere 8.5 fixpack 1 (8.5.01)
    - IBM HTTP Server
    - On Demand Router (ODR)
  - JON Agent
  - JBoss Enterprise Web Server + mod_cluster
- *wpapp1*
  - 4 cores
  - WebSphere 8.5 fixpack 1 (8.5.01)
    - Node agent
  - JON Agent
  - JBoss EAP 6
    - Cluster Node 1
  - DB2 v 9.7 fixpack 6

- *wpapp2*
  - 4 cores
  - WebSphere 8.5 fixpack 1 (8.5.01)
    - Node Agent
  - JON Agent
  - JBoss EAP 6
    - Cluster Node 2
  - DB2 v 9.7 fixpack 6
- *wpdb1*
  - 8 cores
  - JON Agent
  - DB2 v 9.7 fixpack 6
    - http://www.ibm.com/developerworks/wikis/display/im/Red+Hat+Enterprise+Linux+(RHEL)+6+-+DB2+9.7

Packages:

- DB2 JDBC Driver: db2cc4.jar v 4.14.88
  [db2inst1@wpdb1 java]$ java -cp db2jcc4.jar com.ibm.db2.jcc.DB2Jcc -version
  IBM Data Server Driver for JDBC and SQLJ 4.14.88

Additionally, each team member had a local test and development workstation running:

- Oracle Java SE 7
- Apache Maven
- JBoss Developer Studio
- IBM Rational Application Developer (RAD) 8.5

Appendix D – JBoss JIRA unresolved issues

During the hands-on testing of JBoss, we ran into several issues with JBoss EAP and had to come up with workarounds. Some of the issues we discovered can also be found on the public JBoss JIRA bug tracking system. The link below is to the current list of known issues. As this will change over time, we have elected not to list specific issues we encounter here; rather, they are described in the sections where they were encountered.

https://access.redhat.com/knowledge/docs/en-US/JBoss_Enterprise_Application_Platform/6/html/Release_Notes/ar01s07.html

For a complete list of open JBoss issues, please search https://jira.jboss.org.

Appendix E – 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.

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

Capability ($ - indicates capabilities measured in the study)	*IBM WAS 8.5*	*JBoss EAP 6*
JEE6 support $	❺	❺
JDK 7 support $	❺	❺
Production support is available in local language and local hours $	❺	❸
Quality up-to-date documentation $	❺	❸
Completeness of the Administrative GUI $	❺	❹
Performance $	❺	❸
Scripted administration $	❺	❸
Administrative security roles separation (RBAC) $	❺	❶
Dynamic Clustering $	❺	❶
Intelligent Routing $	❺	❶
Application Edition Management $	❺	❶
Health Management $	❺	❶
Upgrade tools $	❺	❷
Problem determination tools (error messages, log analyzer, etc.) $	❺	❹
LDAP support and compatibility $	❺	❹
DBMS support and certifications $	❺	❹
Secure intra-cluster communications $	❺	❹
Command assist feature (audit script) $	❸	❶
Hot deploy capabilities $	❺	❹
Install footprint on HDD $	❸	❺
Install time on developer machine-Full server $	❸	❺
Install time on developer machine-Lite version $	❺	❺
Install and configuration time on production machine $	❺	❹
Server startup time $	❹	❺
HTTPSession failover	❺	❹
High availability and maximum up-time (avoid server restarts) $	❺	❹
Scalability	❺	❹
Integrated Development Environment	❺	❸
Support for latest WS* standards	❺	❹
XA transaction support and recovery $	❺	❶
OS support	❺	❹
Dynamic page fragment cache (Servlets, JSP, etc.) $	❺	❶
Performance tuning advisor $	❹	❷
Flexible management capability (scheduled management of remote servers) $	❺	❸
Eclipse based admin script development tool $	❺	❶
Secure audit of administrative actions $	❺	❸
DMZ hardened proxy	❺	❶
Ability to manage mixed version environment from one console	❺	❺
Ability to fully configure the server without access to its file system $	❺	❹
Portlet JSR 286, WSRP 2.0 support	❺	❶
SIP protocol support	❺	❶
SCA support	❺	❶
JMS Queue Persistence to database	❺	❶

Appendix F – Studies on Software Costs

According to several independent industry studies, the cost of software maintenance and its evolution has risen to over 90% of the total project cost.

The definition of the software maintenance costs in the table below is slightly different depending on the study at hand but, generally speaking, it includes system administration and management, software upgrades, software enhancements and refactoring, bug fixes, troubleshooting and other support related cost factors.

Year	Proportion of software maintenance costs	Definition	Reference
2007	Various	A Systematic Review of Software Development Cost Estimation Studies	Jørgensen and Shepperd (2007) IEEE TRANSACTIONS ON SOFTWARE ENGINEERING, VOL. 33, NO. 1, JANUARY 2007
2003	49%-75%	Software Maintenance As Part of the Software Life Cycle	Erdil (2003)
2000	>90%	Software cost devoted to system maintenance & evolution / total software costs	Erlikh (2000)
1993	75%	Software maintenance / information system budget (in Fortune 1000 companies)	Eastwood (1993)
1990	>90%	Software cost devoted to system maintenance & evolution / total software costs	Moad (1990)
1990	60-70%	Software maintenance / total management information systems (MIS) operating budgets	Huff (1990)
1988	60-70%	Software maintenance / total management information systems (MIS) operating budgets	Port (1988)
1984	65-75%	Effort spent on software maintenance / total available software engineering effort.	McKee (1984)
1981	>50%	Staff time spent on maintenance / total time (in 487 organizations)	Lientz & Swanson (1981)
1979	67%	Maintenance costs / total software costs	Zelkowitz et al. (1979)

Appendix G – References

- *IBM WebSphere® Application Server V7 vs. JBoss® Enterprise Application Platform V5 TCO Analysis*, Jason Armstrong, Jeff Howell, Alex Wang (Summa Technologies), December 29, 2010
- *The Total Economic Impact to IBM WebSphere Application Server Migrating From an Open Source Environment- Federal Client Analysis*, John Erickson, Forrester Research, Inc., June 2012
- *ftp://public.dhe.ibm.com/software/webservers/appserv/pdf/WAS_Liberty_Quick_Start_Guide.pdf*
- IBM WebSphere Application Server Library, http://www-01.ibm.com/software/webservers/appserv/was/library
- IBM WebSphere Application Server for Developers FAQ, http://www.ibm.com/developerworks/downloads/ws/wasdevelopers/faq-wasdevelopers.html
- WebSphere Training and Certification, http://www.ibm.com/developerworks/websphere/education
- IBM WebSphere Application Server Pricing, http://www-01.ibm.com/software/webservers/appserv/was/appserv-was-pricing.html.
- JBoss Documentation, http://www.redhat.com/docs
- JBoss Training, http://www.jboss.com/services/training
- JBoss issue tracking system, http://jira.jboss.org
- JBoss Pricing, http://www.redhat.com/f/html/jboss_channel_skus.html
- Geronimo DayTrader Example Application, http://cwiki.apache.org/GMOxDOC20/daytrader.html
- A Study on Reducing Labor Costs Through the Use of WebSphere Workload Deployer Appliance: http://www-01.ibm.com/software/webservers/workload-deployer/

Credits

- Authors would like to thank members of the JBoss community and Red Hat developers who responded to forum posts with solutions and troubleshooting tips.
- We would like to thank Roman Kharkovski (IBM SWG), and IBM Performance Lab employees for assistance in performance tuning and testing of WebSphere and JBoss.
- We would like to thank the colleagues at Prolifics too numerous to mention who gave us support, advice and encouragement, and especially to Eric Markowitz and Jeff Grossman who provided essential guidance that helped overcome some key issues.
- We would like to thank Jason Armstrong, Jeff Howell and Alex Wang of Summa Technologies for their pioneering work in this area of analysis and the seminal paper they published in 2010.

Legal and Trademarks