JBOSS MIGRATION EXPERIENCE

CASE STUDIES OF WORK PERFORMED BY RED HAT

The information provided in this document details Red Hat's professional services offering for the JBoss platform and represents examples of work Red Hat has completed to aid clients in their migrations to the JBoss platform. Red Hat has assisted clients through a variety of capacities ranging from fully outsourced migrations to augmented subject matter expertise, mentoring, and product customization to fulfill unique client requirements.

Professional Services Overview

Red Hat provides a full-service consulting offering through its middleware consulting arm, Amentra. Amentra is a wholly owned subsidiary of Red Hat. Amentra has been working with the JBoss platform for nearly 10 years, and for the past 3 years Amentra has become the foremost provider of JBoss services in the industry since its 2008 acquisition by Red Hat. Amentra is widely recognized for its expertise across multiple middleware platforms, its signature mentoring approach for integrating knowledge transfer with project activities, and its history with legacy modernization. Amentra services provide a holistic offering to assist Red Hat clients with the adoption of JBoss, as highlighted by the following key differentiators:

AMENTRA
a Red Hat company

- **Straight From the Source –** Amentra's teams have direct lines of communication into the JBoss Support and Product Development organizations.
- **Broad Technical Experience –** Amentra maintains deep expertise on the WebSphere, WebLogic and JBoss platforms. Amentra's knowledge related to WebSphere and WebLogic and associated migration techniques enables a holistic view of a client's past and future platforms.
- **Mentoring Based Approach –** Amentra's widely recognized mentoring model will provide the right approach to jumpstart a client's move to JBoss in a safe and efficient manner.

Migration Experience

A significant driver behind the widespread adoption of the JBoss platform has been Red Hat's ability to aid its clients during application migrations. While many applications can be migrated easily from proprietary platforms such as WebLogic and WebSphere to the JBoss platform through minimal configuration changes, it is also common that applications require integration with third party platforms, alternative libraries, and code changes in order to properly deploy on JBoss. The following stories provide insight into the types of situations where Red Hat has helped clients quickly and smoothly move their applications to JBoss. The case studies listed below are representative of the types of migrations Red Hat has performed and do not amount to the entire portfolio of projects. In many cases, Red Hat has performed each of the types of engagements with many clients.

- At a large regional healthcare affiliate, WebSphere MQ was widely used throughout the organization. In order to make JBoss EAP a viable option, it was critical for JBoss to communicate with WebSphere MQ while still providing the same level of functionality, guaranteeing the same (or better) performance throughput and failover redundancy. To accomplish this Red Hat used the JCA adapter and WebSphere MQ Extended Client provided by IBM. The extended client was required to support distributed transactions, which are not available with the standard client. Once the libraries were in place, the queues and connection factories were configured using JBoss data source descriptors allowing any applications deployed on the JBoss platform to leverage WebSphere MQ.

- At a large retail client, Red Hat migrated an application running on WebSphere to JBoss 5. The application had several proprietary dependencies on the WebSphere platform as well as a custom caching piece, which was to be migrated to JBoss Cache. All of the libraries had equivalent, standard JEE libraries, which allowed for an easy transition away from the proprietary libraries. In addition to a straightforward code change required to leverage JBoss Cache, a new JBoss Cache configuration file was created, tested, and deployed. There were also classloading issues that required special attention with JBoss specific deployment descriptors.

- At a product development company specializing in the manufacturing industry, Red Hat encountered an environment utilizing WebLogic 8.1 to run all internal and external web applications. The client made the decision to migrate from WebLogic to JBoss 5 due to poor performance and the need to bring more nodes online for failover redundancy (??). The code utilized the automatic import of java.util into all JSPs and classes since code deployed to JBoss needs to include imports for classes used from this package. To resolve this issue Red Hat created a script to incorporate import statements throughout the code. In addition, because development was still being done in Java 1.4, developers used variables that were introduced as keywords in Java 5. To get around this issue, each JSP and class had to be updated to use different variable names. The final major hurdle in the code base was removal of WebLogic specific lookups (e.g. - JNDI, Constants, etc.), which were all replaced with JEE equivalents. After developing, testing, and deploying the newly migrated application, the desired performance and failover benchmarks were achieved or exceeded.

- At a large financial services provider, Red Hat uncovered an overly convoluted build process that was hampering the client's ability to migrate from WebSphere to JBoss. Red Hat performed extensive configuration of the IDE and the Ant build environment to get working code. Significant improvements were made during the migration by ensuring that builds were only performed using Ant and Ivy to achieve a standard build and manage dependencies. The introduction of a proper build process, dependency management, and automation to deploy code greatly enhanced the client's ability to move away from binary code management and for once be able to reconstruct their releases from source. Prior to working with Red Hat, the client had operated by each developer having to check in multiple copies and versions of each library due to build errors and build through Eclipse. This had resulted in no knowledge whatsoever regarding which libraries were actually being used by the classloader at runtime. Investing in this area greatly improve the quality of the code by taking the complexity of the build environment out of each individual developer's hands.

- At a large commercial bank, Red Hat encountered a large code base that targeted older JVMs. Red Hat performed a full pass through the code to uncover errors such as the use of words that are now Java keywords and other minor compilation errors. In addition, due to the introduction of Generics and improvements in the Collections framework, most of the additional issues were warnings regarding type safety of the code. Red Hat corrected the code by leveraging the IDE and the new, targeted JDK. Beyond the initial code changes, the primary challenges were around what libraries were being used and how dependent the application was on the originating container. Proprietary/vendor libraries were changed to versions of common open source libraries. Applications that heavily used tools provided by the container (e.g. - Custom Tags, MVC layer, Web Services, etc.) were evaluated to determine the ability to port the libraries directly or find comparable replacements. While it is relatively easy to find the libraries in each application archive, Red Hat's experience in migrating applications to JBoss accelerated the process of completely mapping out the dependency graph as it related to proprietary libraries. Many of the dependency challenges were managed through classloader isolation, library replacement or upgrading to a later version and making the appropriate API tweaks back to the code.

- Working with a global credit card processor, Red Hat performed extensive customizations to the JBoss Enterprise Application Platform (EAP) in order to comply with the client's unique and rigid security requirements for its company-wide standard build. The EAP platform was integrated with IBM's TFIM (Tivoli Federated Identity Manager) using standard JAAS security and the PicketLink framework. Red Hat's services team worked closely with client stakeholders and Red Hat product engineering teams to ensure that any customizations made were congruent with long term plans for the platform and would be fully incorporated into future releases of JBoss EAP and supported by Red Hat. The incorporation of these customizations enabled the client to deploy a large set of applications to JBoss while not compromising on any of its core requirements.

- At a large online retail store, Red Hat migrated a large online store processing millions of transactions a day from ATG Dynamo to ATG running on JBoss Enterprise Application Platform (EAP). Red Hat performed a thorough infrastructure and application review to ensure that existing application requirements and infrastructure requirements could be met in the new JBoss environment. The application utilized various ATG specific libraries and configurations. These libraries were mapped over to the new JBoss environment, as well as code recommendations and industry best practices for application re-factoring. This resulted in the next generation of the platform, which performed better and was more maintainable through rigorous performance testing and tuning of the end-to-end platform. The new platform also required multiple versions of web service stacks deployed across various applications. These web service stacks were integrated with JBoss without removing the default JBoss WS stack to ensure portability and maintainability with future versions of the JBoss Enterprise Platform.

- At a major airline, Red Hat migrated multiple applications from WebLogic 9.x and WebLogic 10.x to JBoss Enterprise Platform 5. This also required integration with both TIBCO EMS as well as WebLogic JMS using JCA to ensure a standards-based integration supporting distributed transactions. The applications also utilized message driven beans and both stateless and stateful session requiring deployment descriptor changes. In addition to EJB's, the application also utilized various versions of Axis and Spring. These applications were standardized to use Snowdrop for Spring 2.x and Axis 2.0 to ensure maintainability and portability between application servers. These applications were also integrated with multiple mainframe applications using standard JCA connectors supporting distributed transactions and 99.999% of system uptime.

- At a major financial retailer, Red Hat led a major migration through proof-of-concept applications. These proof-of-concepts were modeled from the actual application stack consisting of Spring, Struts, Axis, JAX-WS, JSF, and Terracotta for a distributed cache. The application also required integration with Tivoli and IBM WebSphere MQ. This resulted in a successful migration and developer training from the proof of concept implementations. The proof of concept code and configurations were directly applied to the applications with minimal changes.

- At a major movie production company, an online game was migrated from Tomcat and TIBCO to JBoss EAP 5, while integrating the platform with Smart Fox servers utilizing Flash. The application was also integrated and performance tuned to support 99.999% of uptime and millions of concurrent players. The platform was integrated with the ControlTier platform to support automated deployments and provisioning of environments. This approach also addressed complex version control and automated patching along with custom scripts. The application was also refactored to use Hibernate 3.x from standard DAO's and JBoss cache for optimal performance to reduce database load on the MySQL database.

- At a large electronic medical device company, the primary device processor factory management application was migrated to JBoss EAP 4.X from a legacy, non-J2EE platform. With Red Hat's thorough review, implementation assistance, and mentoring on the application code, enterprise environment architecture, deployment procedures, and load testing strategy the response time of the management application was reduced 300% while the production environment also achieved a level of reliability, consistency, and automation never before encountered for the factory management application.