Migrating JEE Applications to SpringSource tc Server

A Business Perspective

WHITE PAPER

vmware®

Table of Contents

1. Migrating JEE Applications to SpringSource tc Server™

In this whitepaper, we will be discussing an important trend in IT development and deployment architectures: the evolution from JEE Application Servers to lighter weight JAVA containers. As evidenced by the growing popularity of Tomcat server – according to Evans Research, Tomcat is in use at 68 percent of all organizations (50 percent of which are built in Spring) making it the most widely deployed application server on the planet, many IT organizations have been re-thinking their commitment to commercial JEE Application Servers. This is due to challenging business environments that drive the need for more cost effective application architectures and, just as importantly, the trend towards the use of more lightweight and efficient mechanisms for application development. This also becomes even more important as organizations begin to consider building applications for deployment into the cloud future. When IT organizations talk about "migrating" their applications, they generally are focusing on one or more of three distinct situations. These are:

- Moving existing applications (or portions of applications) off of JEE servers and onto lightweight, modular, horizontally scalable container infrastructures
- Expanding access to existing JEE applications by adding services layers built in lightweight containers.
- Transitioning new development away from JEE application servers and focusing on light weight containers

We will be focusing primarily on the migration of existing JEE applications to the SpringSource tc Server™, which is an "enterprise ready" web application server built on top of standard Apache Tomcat and optimized for the Spring Framework. There are many excellent reasons to consider moving applications off of commercial JEE servers. While we are focusing on the JEE application migration process, many of the business and technical decision factors apply equally well to the second and third situations as well.

The IT Manager considering migrating applications from their commercial JEE servers has a number of important things to determine before embarking on the process. It is extremely important to clearly understand the objectives of the migration in order to properly assess the benefits, and costs, of such a program.

In this whitepaper, we focus on the migration decision processes for determining whether to migrate your applications or portions of your applications from commercial JEE servers to SpringSource tc Server. Many of these decisions should be based on business factors, although there are also significant technical opportunities and challenges to address.

2. Why Consider Migrating a Working Application?

Perhaps the most important question to resolve is "why would you even consider migrating a working, successful, application?" With all of the time and budget demands on today's IT organizations, "if it works, don't mess with it" surely applies.

While there are certainly valid reasons to leave well enough alone, there are also a number of reasons that IT organizations are deciding to migrate applications from one of the commercial JEE application servers to the "enterprise ready" SpringSource tc Server™. In this context, the term "application" includes complete applications, portions of an application, or functional modules/layers ("services") which make up an application. In this whitepaper we will discuss those reasons, processes to determine a migration strategy, and how to select viable migration targets.

In most cases, migration of the organizations JEE applications is not an "all or nothing" process, so a standardized way to select specific migration targets will be a valuable decision tool. In many cases migration decisions will be motivated by long term strategic objectives, such as architecture and operating cost reduction plans, but in some cases there is an immediate tactical challenge that can best be met by careful migration.

In one recent case, a financial services application running on a JEE Application Server was becoming seriously overloaded at critical times of the day. Detailed performance analysis showed that almost 85 percent of the processor at that time was being consumed by a single JAVA application component. By abstracting that component, converting it to a service, and migrating it to a highly parallel Tomcat deployment, the company avoided purchasing multiple new licenses, saving hundreds of thousands of dollars in equipment, licensing, and operating (mostly maintenance contracts) costs. In this case, the application was no longer "working", so something had to be done, but the brute force approach of purchasing additional costly JEE application server instances was avoided.

In another recent case, investigation of a large scale application determined that it was almost 100 percent servlet code, with some utilization of data persistence, and was hosted in a cluster configuration. Most of the code transported seamlessly, plus adding OpenSource Hibernate technology. This resulted in an annual maintenance contract savings of almost $200,000, an excellent return on the three man months the migration project took.

3. What's an "Application Server"?

There are many definitions (a Google search returns almost a million references) for "Application Server" floating around the industry, in all too many cases promulgated by vendor marketing departments with the primary objective being "we are and they aren't". For our purposes, we are specifically focusing on the world of JAVA, even though the most widely used "application server" in the world is arguably the Microsoft Windows OS coupled with any of the Microsoft Visual IDE's.

Simply stated, an "application server" is a software framework dedicated to the efficient execution of procedures (scripts, routines, programs, ...) for supporting the construction of business applications. The term was probably created in the context of web applications. In these, the application server acts as a set of components accessible to the software developer through an API defined by the platform itself. These components are usually performed in the same machine where the web server is running, and their main job is to support the construction of dynamic pages.

Commercial application servers vary considerably in the range and quality of the services they offer to the IT function, a portion of which of which are based on "standards" (JEE, CORBA, etc). Commercial Application Servers also include many application services which provide valuable functionality but are implemented in highly proprietary ways. In this white paper we're assuming that you are starting with mature JEE standard application servers from one of the major vendors. These products are highly sophisticated, feature rich, and correspondingly expensive/complex. Additionally, much of the functionality in these products is outside of the JEE specifications, thus making each of them proprietary to some degree and particularly so in the areas of application management/administration. Other JEE application servers, also meet the JEE standard, but generally provide a much less broad suite of application services and are less sophisticated in their operations/management tooling.

Without doubt, the most widely utilized Java application server is Apache Tomcat, which for many years was the reference implementation of the Java servlet specification. The Apache Tomcat community continues to develop and support Tomcat, with regular releases that add sophisticated functionality, while maintaining 100 percent standard compliance and high reliability/performance. tc Server leverages this work, and adds both enterprise class management and monitoring and commercial support.

4. Reasons to Migrate from JEE to tc Server

Organizations that choose to migrate existing applications to a new application server are typically motivated by one or more of the following goals:

Costs
Infrastructure costs are frequently mentioned as a primary motivator for migration, and are certainly important. That said, these costs can be subtle, particularly since in most cases the license itself is a "sunk cost" and all the maintenance fees probably continue if you use any of your licenses (contract "non-retirement" provisions). Some of these cost considerations include:

- Capacity Expansion- The need to expand deployment of an application in a cost effective way frequently drives interest in alternative infrastructures. A frequent driver is also the need to provide additional access by a broader community of users.
- Application Replacement- When an application "wears out" and is being replaced entirely, there are opportunities to consider alternatives, particularly where the application simply doesn't require much of the power of a JEE Application Server.
- Vendor Replacement- While relatively rare, some IT organizations are choosing to replace their IT infrastructure vendors, for a variety of reasons. The cost advantages of replacing older architectures and equipment can be an important part of the cost analysis.

Equally important are all the costs associated with maintaining an application infrastructure and the infrastructure's effect on the cost of maintaining the applications themselves. Many studies have shown that maintenance costs are a much larger component of TCO than the original license acquisition.

Shifts in application development/deployment architectures
During the late 1990's and into the mid 2000's, most IT organizations (and application vendors) bought into the JEE vision. Without debating the reality of that vision, many IT Organizations have realized that today's JEE application servers have evolved into "do absolutely everything for absolutely anyone" behemoths and, not being modular, force everyone to carry the full weight of all those un-needed capabilities. Additionally the JEE vision was both difficult and costly to achieve in the first place for a large number of reasons and they have transitioned their JAVA application development in much more productive directions. These include a number of alternative languages and architectures (for example, widespread adoption of the Spring Framework), with the common characteristics being:

- Much simpler to develop and maintain
- More agile, allowing IT to better meet the rapid changes in business requirements
- Vastly lighter weight, suited to highly parallel, scalable, redundant, deployment architectures
- Order of magnitude lower acquisition and maintenance costs

Harmonize or realign standard deployment environments in the organization
In most large organizations, there are a variety of application infrastructures in use, typically resulting from divisionalized/departmentalized IT or M&A activities. The complexity of maintaining multiple infrastructures makes it difficult to create today's distributed application service environments. It also requires costly staff duplication to support differing technologies and release cycles. By shifting to a single infrastructure, with enough flexibility to support a wide range of application requirements, it becomes much easier to develop and maintain applications across the organization.

Related to this is the tendency for JAVA developers to use light weight containers (Tomcat, for example) on their desktops, while production deployment is on JEE Application Servers. This creates the need for "porting" each application from Tomcat to the JEE Application server, duplication of testing, etc. By harmonizing the development and deployment architectures, handoff between developers and production is significantly simplified and update costs reduced.

Virtual deployment environments
The reality is that today's datacenters are increasingly being virtualized. JEE servers have significant footprints that can reduce application density available in a virtual environment: the answer is a lightweight approach that is optimized for usage in a virtual environment. Add to that, in the application development process, QA organizations are likely testing the application inside a vitural machine, which extends that same challenge of scalability into the development environment.

4.1 Cost Considerations

Return-on-investment should drive the decision to migrate; ultimately, benefits must outweigh costs. Accurate quantification of both the benefits and the costs can be somewhat elusive, so it is important to take the process step by step and maintain careful records to build an experience base for future decision making. Carefully assess the costs (some of which may be fuzzy) of migrating against both the obvious costs (annual maintenance contracts, for example), as well as the "invisible" costs (difficulty of finding skilled IT operations staff, overly complex application support, etc) of not migrating:

- Consider acquisition costs for tc Server, and in particular for any third party technologies needed to supplement tc Server's functionality. In most migration situations, the JEE application server cost has already been written off, so license cost is only relevant in the capacity expansion situation. In a capacity expansion situation, the cost of acquiring new JEE licenses and the mega-hardware to support them should also be factored into the decision.
- Consider the costs of actually doing the migration, including the effect of committing scarce development resources to "doing it over", rather than meeting another business need. While migration should be much less effort than creating the application in the first place (assuming we've properly selected the migration target), in some cases migration can actually become a top to bottom re-write. In either case, you are developing and releasing a somewhat new application on a completely new infrastructure, which requires utilizing all standard development processes (design, code, test, document, release, etc).
- Consider the cost of ongoing infrastructure maintenance with both JEE and tc Server infrastructures. Include both vendor costs and identifiable internal costs, such as accepting and applying vendor maintenance releases (in some environments, the level of patch activity is so high that it swamps license costs in the first few months). The commercial JEE application server vendors share a common characteristic…they all make far more on their maintenance than they do on their license fees (see any annual report for verification), and their maintenance contracts are particularly rigid. On the tc Server side, consider the cost of potentially dealing with multiple open source communities or vendors, each of whom has their own release cycles and virtually none of which do integration testing with other vendors/communities technologies.
- Consider the cost of maintaining (or perhaps extending) the application itself. The application monolith architectures that characterize all too many JEE applications can make it very difficult to fix even small problems, particularly when the original authors have moved on to other things, and may make those applications prohibitively costly to extend/improve in todays agile business environments. On the other side, consider the cost of "migrating" the application code from the JEE servers to tc Server and the costs of maintaining the migrated application in a tc Server environment.
- Consider the cost of quality issues that can arise due to increased lines of code and the challenges of implementing and testing applications based on a traditional full-stack Java EE, compared with lighter-weight technologies now available, such as the Spring Framework on top of tc Server.

The charts below provides a brief look at the readily measurable outside costs (using average vendor discounting) for a typical JEE server and for tc Server. We will be talking about Total Cost of Ownership (TCO) in more detail in a future whitepaper, including discussion of some of the "hidden costs", such as developer productivity and application maintenance.

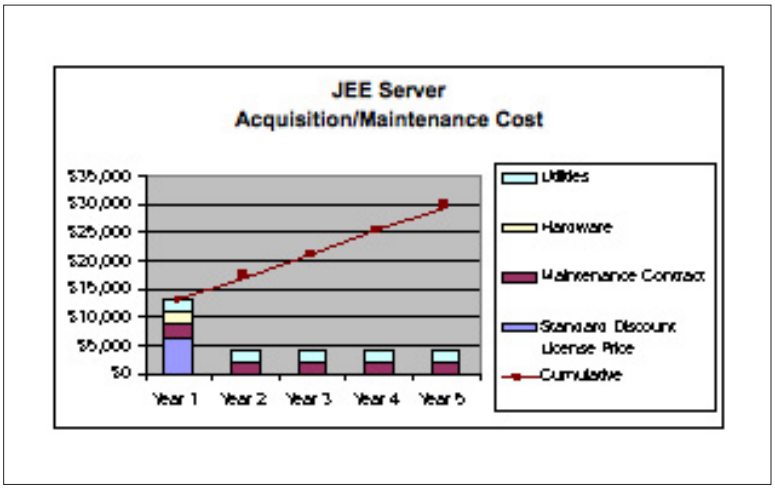

Figure 1: JEE Server Costs

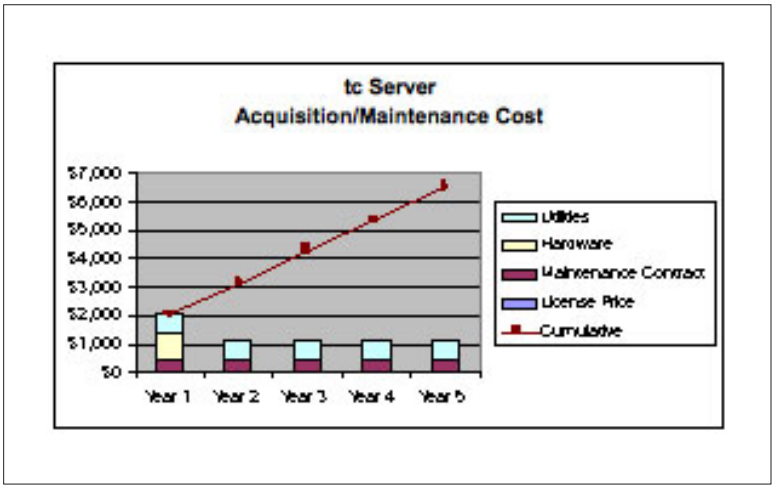

Figure 2: tc Server Costs

As the above clearly shows, the cost savings for acquiring comparable functionality, support, and performance can be very significant, in this case nearly $ 24,000/server. This factor alone creates significant interest in budget conscious IT organizations, although it is only part of the decision process. When we talk about "migration", we are generally not considering prior license acquisition cost, which is a "sunk cost". Looking only at maintenance and operations cost savings, we still see a $ 15,000/server savings over the 5 year period. That said, the ability to recycle the JEE licenses, servers, and maintenance agreements for other uses may offer significant migration savings, because you don't have to acquire new JEE servers.

4.2 Shifts in Application and Deployment Architectures

A second major factor driving IT organizations to consider migrating their applications (or more often selected portions of their applications) from JEE to tc Server is the transition away from monolithic application architectures and toward more modular/layered, horizontally scalable, architectures. While there was nothing in JEE that prevented the development of cleanly layered applications, the developer tendency was to lump everything associated with an application into one place and to take full advantage of the highly integrated services provided by these costly commercial JEE servers.

The unfortunate result of that process was to create huge applications, which have proven to be very hard to maintain and extremely hard to extend to meet today's rapidly evolving business requirements. Because of this situation, many IT organizations have been busy de-composing their monolith applications into more modular layered architectures, enabling portions of the application to be extended without having to tackle the whole thing.

A second approach is to continue to utilize the JEE Application Server for those operations that leverage their rich suite of services (often called "back office" functionality), by encapsulating those functions into services layers that are then utilized by larger numbers of lighter weight application front ends. The resulting application modules rarely require more than a very small portion of the commercial JEE server's capabilities, thus opening the door to improved deployment architectures based on light weight containers such as tc Server with very little additional effort and substantially lower cost because you are not paying for functionality you don't need/want anyway. One financial services firm is creating a dozen or more new business applications per week using the Spring Framework, while allowing them to maintain the "back office" business logic on their JEE Application Servers untouched for months at a time.

Finally, the concept of implementing an application in bite sized, horizontally scalable, application services (what ever we call them, modules/components/etc) offers significant business benefits. Each service is much more readily maintained and new business applications can be created by assembling existing services, plus a few new ones, in a fraction of the time it takes to build a monolithic application from scratch. Development organizations have used this concept to leverage both emerging programming models and cost effective parallel deployment architectures to meet the business demands for more agile and accessible applications. Taking the application module (e.g. "service") to its logical conclusion, we wind up with highly distributed applications that offer truly scalable performance and excellent business flexibility. These deployment architectures are almost made for tc Server, with some business application services now utilizing dozens to hundreds of parallel service instances…the ultimate in redundancy and scalability.

4.3 Harmonize or realign standard development/deployment environments

A third driver for migrating JEE applications to tc Server is to converge the IT organization around a common set of infrastructures and tooling. This is a highly desirable goal and each of the major vendors has their suite of products and insists that they can provide the whole answer, but the reality is that they can not. Even where the single vendor does offer a wide range of capability, the fact remains that some portion of these solutions are "orphan products", merely there to complete the suite, and don't compare well to other alternatives for the same functionality. This can drive individual development teams to select technologies that best suit their particular needs…good for the project, but very costly for the IT organization as a whole.

Another factor that has driven fragmented development/deployment architectures is that with a high degree of business consolidation, it was inevitably the case that individual IT organizations had selected their own favorite technologies and were then forced to "just merge" and behave as one. Almost every JEE Application Server utilization survey shows one very interesting thing, that each organization uses products from multiple JEE server vendors, each of which is complex and no two of which are actually "compatible" from a plug-and-play, suite of infrastructure services, or administrative/management points of view. In many cases, it becomes a major development project to migrate a significant application from one vendor's JEE server to another, for reasons having nothing to do with the JEE standard itself.

Managing all this is a costly and error prone process, leading IT organizations to try to converge on one common, standards based, commonly manageable, deployment architecture. Realistically, converging on a single JEE Application Server is highly costly and fails to accomplish other objectives of migration. This factor, along with those mentioned above, provides significant value to the organization by converging on a more cost effective and agile deployment infrastructure...tc Server.

4.4 Virtual deployment environments:

Increasingly, data centers are turning to virtualization to more effectively utilize their computing resources. Many studies of JEE Application Server behavior show that one of several conditions exist:

- The JEE Application Server is underutilizing the physical resources
- JEE Application Server utilization is frequently dominated by one or two application components, while the rest of the application is basically coasting along.
- Within the data center, there are frequently specific JEE Applications that have very high peak to average utilization ratios, leading to over capacity 90 percent+ of the time, in order to accommodate loads during the peak load 10 percent of the day.

In today's demanding business environments IT organizations simply can not afford to be seeing 15-20 percent loading of their expensive facilities (servers, floor space, utilities, etc), nor 5-10:1 overcapacity to handle peak to average extremes. Focusing on those application components that contribute most of the server load or are responsible for the peak loading under daily variations, we can also decompose those applications into the components of the application that most benefit from highly horizontal dynamic scalability, then virtualize deployment of those components.

By migrating those applications that do not really need the weight and complexity of a JEE Application Server to a lighter weight and more modular infrastructure, such as tc Server, we enable virtualized deployment of applications and application services/components with significantly improved consolidation ratios. Some customers have reported that applications migrated from JEE Application Servers to tc Server have 2-3 times lower resource utilization on the same hardware, thus allowing them to make more effective use of virtualization.

5. Planning the Migration Process

Most IT organizations considering migrating applications from JEE to light weight environments such as tc Server will be focusing on a relatively small subset of their existing applications, at least initially. This is a very good idea because it provides the opportunity to take smaller steps and to assess the results incrementally. Do not underestimate the cost of the learning curve associated with any infrastructure changes.

In many cases, a low impact application, or one with reduced technical risk, is a better choice for early migration than to tackle a high visibility mission-critical application. In some cases the business needs are such that the IT organization does need to migrate a large JEE application. In that case it is critical to very carefully plan the migration, take it step by step, and above all be realistic about the time and effort the migration may take. While it does happen that large scale applications migrate smoothly, they are rarely "plug and play", even between JEE servers from various vendors; thus it is prudent to be cautious when undertaking such a migration.

Focus on the long term as well as the short term. Understand the scope, complexity, and technical challenge. Migrating a codebase involves more than simple direct effort cost. Consider both legacy issues and expected new, "greenfield" development as part of the analysis.

Form a cohesive technology and architecture vision for the organization that addresses these factors:

- Migration criteria and process for legacy applications
- Best practice technology, architecture, and process for new development
- Application server (or in the case of tc Server, additional services) licensing costs
- Cost of managing the new tc Server environment, compared with the current JEE application servers
- Resource utilization, hardware, and environment (space, power, HVAC, etc) costs.
- Complexity and direct cost of migration effort, including, where code refactoring is needed, a thoughtful balance between "minimal effort" and "best practice" conversion. This is an area where migration costs can skyrocket, without corresponding benefit.

Fortunately, vendor and customer studies have shown that almost all applications only utilize a small portion of the suite of services provided by JEE Application Servers. This means that the JEE application server infrastructure has all the capability to do practically everything for anybody (market forces drive this, with a very few extremely demanding customers driving the feature sets), while becoming so huge and complicated that the products may not be a particularly good choice for any one application.

With tc Server, you have to opportunity to select the add-on services you need and leave out all those you don't need, significantly reducing complexity, cost, and computing overheads. tc Server provides an excellent environment for today's improved frameworks such as Spring, and with tc Server we add enterprise scale administration/management to both.

With tc Server, you have to opportunity to select the add-on services you need and leave out all those you don't need, significantly reducing complexity, cost, and computing overheads. tc Server provides an excellent environment for today's improved frameworks such as Spring, and with tc Server we add enterprise scale administration/management to both.

5.1 Assessing Applications for Migration

One approach is to group candidate applications by the amount of refactoring effort. Assuming no specific external requirements or pressures, it is best to begin migration with those applications that require the least effort. Another approach is to focus on one business application, typically one which already exhibits one or more of the characteristic drivers mentioned above. In either case, this is an area requiring development involvement at a detail level, because the cost of migration is highly influenced by both the development time expended and the lost opportunity cost while those developers are not producing new business applications for the business.

Successful migration projects require close coordination between IT and development, both during the cost assessment and during the subsequent project in order to assure that the cost benefit balance gets achieved as planned. Every company or individual will have their own criteria, but as an example, consider this grouping:

Minimal effort

- One day or less of work
- Runs-as is (packaged as WAR, no use of unsupported APIs).
- Testing effort only (but DO NOT underestimate this!)

Low effort

- Several days to two weeks of work
- Few changes, plus testing effort

High effort

- Many weeks or months of work
- Higher risk
- Very large testing effort

Minimal effort migration projects are relatively unusual for monolith applications hosted on JEE servers, but it is fairly common to discover a module/component that can be abstracted relatively easily and provide disproportionate benefit. See the whitepaper "Migrating JEE Applications to SpringSource tc Server™: A Developers Guide for more detail on the development cost/risk analysis.

6. Implementing the Migration

One the assessment is complete and the decision made, the migration process can begin. Typically, some things will be discovered during the project that were not obvious during the assessment, but if the planning process was sufficiently through, these can generally be handled without major disruption to schedule or resources.

We have seen projects go better than planned, and projects that hit significant speed bumps. The former are typically a result of very conservative planning, the latter the result of some significant dependency on one or more of the JEE Application Server services that wasn't surfaced during the planning stage.

The range of time/effort for enterprise scale application migration has run from as short as a couple of person months (Spring based application running on WLS) to multiple person years (significant use of JEE running on jBoss JEE Application Server). In both cases, the cost estimates were fairly accurate and the benefits met or exceeded expectations.

7. Conclusion

While not all applications benefit equally from migrating from JEE Application Servers to tc Server, it is clear that IT organizations can dramatically reduce their total operating costs and improve their resource utilization significantly. The large portion of today's JAVA applications that are underutilizing commercial JEE Application Servers and the number of applications and application components/services that benefit from lightweight, modular, horizontally scalable deployment architectures both provide significant opportunities.

Perhaps the most important step when considering migration of applications from one environment to another is planning. While the temptation is to jump in and start coding, it is critically important to understand the objectives and benefits, assess the risks and costs, and then make the business decision. There are many successful migration projects and one common characteristic is that the up front planning process was completed before launching the project.

In the whitepaper "Migrating JEE Applications to SpringSource tc Server™: A Developers Guide", we will dig deeper into the details of the technologies involved in the JEE Application Server to tc Server migration process.

vmware

VMware, Inc. 3401 Hillview Avenue Palo Alto CA 94304 USA Tel 877-486-9273 Fax 650-427-5001 www.vmware.com