

WebSphere®

JOURNAL

The World's Leading Independent WebSphere Resource

www.SYS-CON.com/WebSphere

DECEMBER VOLUME 2 ISSUE 12

INTERNATIONAL CONFERENCE & EXPO

February 24-26, 2004

**Hynes Convention Center
Boston, MA**

see page 40



DEVELOPMENT TECHNOLOGIES EXCHANGE

FROM THE EDITOR

**Offshoring Offers
Opportunities for U.S. IT
Troubleshooters**

BY JACK MARTIN • PAGE 4

PRODUCT REVIEW

**Cyanea/One
from Cyanea**

BY JAY JOHNSON • PAGE 32

FINAL THOUGHTS

Portal Shmortal

BY STEPHEN AREES • PAGE 50

DISPLAY UNTIL FEBRUARY 29, 2004

\$8.99US \$9.99CAN

12>



0 09281 03422 3



INTERVIEW BY JACK MARTIN PAGE 6

Replicating J2EE Success

BY WARREN MACEK

Using server clones to increase application performance and redundancy



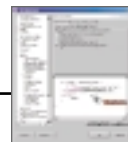
PAGE 12

Legacy Integration in Enterprise Class Applications

It's not just about the technology

BY UDAY KUMAR

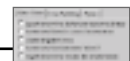
PAGE 16



Viewing Security as a Process

BY CALEB SIMA

Security isn't a one-time event in the application development life cycle



PAGE 22

Formatting Your Code Automatically

The WASD Code Formatter

BY KULVIR SINGH BHOGAL & KWANG SIK KANG

PAGE 24



Are All Systems Go?

How to assess the production readiness of your WebSphere application

BY MAX KING & YATEEN DHARESHWAR

PAGE 26



Building High-Availability Web Applications

BY CHRIS DELGADO

Managing HTTP sessions within clustered WebSphere 5 environments

PAGE 36



Offshoring Offers Opportunities for U.S. IT Troubleshooters

BY JACK MARTIN

Whether you like it or not, the offshoring of IT projects is here to stay, so get over it. If you can't get over it, get out of IT. But before you head for the exit, think about what you'll be missing.

- Over 70% of all IT projects fail in some way. Three-quarters go over budget, are delivered late, fail to meet functional requirements, or become financial black holes with no hope of ever recouping the money invested.
- Enterprise Resource Planning (ERP) products are widely deployed in the U.S., yet roughly half of the companies that have purchased these products view their ERP implementations as unsuccessful.
- Forty percent of IT projects fail to achieve their business case within one year.
- Over 30% of all IT projects will be canceled before completion.
- Only 16% of all software projects are completed on time and on budget.

As offshoring gains momentum, these statistics can only worsen due to a lack of hands-on control. Remember, these statistics are for corporate projects, not rookie developers.

This should tell most people that it is next to impossible to succeed in the field of corporate IT. Maybe the people working in the industry are all idiots. Or perhaps it's just that the people in charge have very different expectations for these projects than the people working on them.

The problem is that the people who make the decisions at major corporations know next to nothing about IT – and have no desire to learn. The guy running a Fortune 500 company – who thinks that learning to send e-mail and use a browser is a major intellectual accomplishment – is the same guy who approves \$100-million infrastructure deals.

Many of them will tell you that they don't want to know the details; they just want to know when the project will be completed and how much it will cost. I have a friend who runs a company. He has often said that "The biggest problem with comput-



er people is that they don't listen." He has no interest in the details, either.

Corporate America has decided that IT costs have gotten out of hand, and the magic formula for solving the cost-containment challenge is to off-shore as much as possible, as fast as possible. For someone who doesn't want to know the details, this idea is

at first glance pure brilliance. The programmers work for 20% of what American programmers make. So what if they make a couple of mistakes; they'll make it up on the volume.

Offshoring is a fad. Like any fad, it seems like a good idea in the beginning, and people ask why no one ever thought of it before. Microsoft thought of it years ago, and indeed has been offshoring a substantial portion of their code writing since Windows 95 was written.

Microsoft's operating system and Office products are riddled with security flaws and coding mistakes. Considering that they currently have about \$48 billion in the bank, you have to wonder what happened. What happened? Profit – not quality – was the Number 1 priority. Microsoft can afford this because they don't absorb the true cost of ownership of their products, their end users do.

The rules for corporate IT are different, and the people in charge all want to be like Bill Gates: super-successful and sitting on a mountain of cash. But these are the same people who don't want to know the details of their IT projects.

A corporation doing a one-shot project will never be able to do as well as Microsoft at writing code. Writing code is Microsoft's core business and the guy running the company *wants* to know all of the details. The second flaw in this line of thinking is that these smaller corporations cannot push off their cost of ownership to third-party end users; they are the end users.

The successful 21st-Century IT professional in the U.S. will be an on-site troubleshooter first, and an IT specialist second. Remember, corporate will not want to know the details; they'll just want to know when you can fix it, and how much it will cost.

Though it may not seem so right now, things are about to get much better for U.S. IT workers.

ABOUT THE AUTHOR... Jack Martin, editor-in-chief of *WebSphere Journal*, is cofounder and CEO of Simplex Knowledge Company, an Internet software boutique specializing in WebSphere development. Simplex developed the first remote video transmission system designed specifically for childcare centers, which received worldwide media attention; and the world's first diagnostic-quality ultrasound broadcast system. Jack is coauthor of the upcoming book, *Understanding WebSphere*, from Prentice Hall. **E-MAIL...** jack@sys-con.com

ADVISORY BOARD

Richard Arone, David Caddis, Mike Curwen, Devi Gupta, Lloyd Hagemo, Barbara Johnson, Shannon Lynd, James Martin, Doug Stephen, Victor Valle

EDITOR-IN-CHIEF JACK MARTIN
EXECUTIVE EDITOR GAIL SCHULTZ
CONTRIBUTING EDITOR PATTI MARTIN
PRODUCT REVIEW EDITOR JAY JOHNSON
SPECIAL PROJECTS EDITOR RICHARD ARONE
MANAGING EDITOR JEAN CASSIDY
EDITOR NANCY VALENTINE
ASSOCIATE EDITORS JAMIE MATUSOW
 JENNIFER VAN WINCKEL

WRITERS IN THIS ISSUE

Stephen Arees, Kulvir Singh Bhogal, Chris Delgado, Yateen Dhareshwar, Jay Johnson, Kwang Sik Kang, Max King, Uday Kumar, Warren Macek, Jack Martin, Caleb Sima

SUBSCRIPTIONS

For subscriptions and requests for bulk orders, please send your letters to Subscription Department. SUSCRIBE@SYS-CON.COM
 Cover Price: \$8.99/Issue Domestic: \$149/YR (12 Issues)
 Canada/Mexico: \$169/YR Overseas: \$179/YR
 (U.S. Banks or Money Orders)
 Back issues: \$15 U.S. \$20 All others

PRESIDENT AND CEO FUAT KIRCAALI
PRESIDENT, SYS-CON EVENTS GRISHA DAVIDA
GROUP PUBLISHER JEREMY GEELAN
TECHNICAL DIRECTOR ALAN WILLIAMSON
SENIOR VP, SALES & MARKETING CARMEN GONZALEZ
VP, INFORMATION SYSTEMS ROBERT DIAMOND
VP, SALES & MARKETING MILES SILVERMAN
PRODUCTION CONSULTANT JIM MORGAN
FINANCIAL ANALYST JOAN LAROSE
ACCOUNTS RECEIVABLE CHARLOTTE LOPEZ
ACCOUNTS PAYABLE BETTY WHITE
ADVERTISING DIRECTOR ROBYN FORMA
DIRECTOR OF SALES AND MARKETING MEGAN MUSSA
ADVERTISING SALES MANAGER ALISA CATALANO
ASSOCIATE SALES MANAGERS CARRIE GEBERT
 KRISTIN KUHNLE
 BETH JONES
CONFERENCE MANAGER MICHAEL LYNCH
NATIONAL SALES MANAGER SEAN RAMAN
ART DIRECTOR ALEX BOTERO
ASSOCIATE ART DIRECTORS LOUIS F. CUFFARI
 RICHARD SILVERBERG
 TAMI BEATTY
WEB DESIGNERS STEPHEN KILMURRAY
 CHRISTOPHER CROCE
CONTENT EDITOR LIN GOETZ
CIRCULATION SERVICE COORDINATORS NIKI PANAGOPoulos
 SHELIA DICKERSON
 EDNA EARLE RUSSELL
CUSTOMER RELATIONS MANAGER BRUNI STARAPOLI

EDITORIAL OFFICES

SYS-CON Publications, Inc.
 135 Chestnut Ridge Road, Montvale, NJ 07645
 Telephone: 201 802-3000 Fax: 201 782-9637
SUSCRIBE@SYS-CON.COM
 WebSphere® Journal (ISSN# 1535-6914)
 is published monthly (12 times a year).
 Postmaster send address changes to:
 WebSphere Journal, SYS-CON Publications, Inc.
 135 Chestnut Ridge Road, Montvale, NJ 07645

WORLDWIDE NEWSSTAND DISTRIBUTION
 CURTIS CIRCULATION COMPANY, NEW MILFORD, NJ

FOR LIST RENTAL INFORMATION:
 KEVIN COLLOPY: 845 731-2684, KEVIN.COLLOPY@EDITHROMAN.COM
 FRANK CIPOLLA: 845 731-3832, FRANK.CIPOLLA@EPOSTDIRECT.COM

© COPYRIGHT 2003 BY SYS-CON PUBLICATIONS, INC. ALL RIGHTS RESERVED. NO PART OF THIS PUBLICATION MAY BE REPRODUCED OR TRANSMITTED IN ANY FORM OR BY ANY MEANS, ELECTRONIC OR MECHANICAL, INCLUDING PHOTOCOPY OR ANY INFORMATION STORAGE AND RETRIEVAL SYSTEM, WITHOUT WRITTEN PERMISSION. FOR PROMOTIONAL REPRINTS, CONTACT REPRINT COORDINATOR, CARRIE@SYS-CON.COM. SYS-CON PUBLICATIONS, INC. RESERVES THE RIGHT TO REVISE, REPUBLISH AND AUTHORIZE THE READERS TO USE THE ARTICLES SUBMITTED FOR PUBLICATION.

ALL BRAND AND PRODUCT NAMES USED ON THESE PAGES ARE TRADE NAMES, SERVICE MARKS OR TRADEMARKS OF THEIR RESPECTIVE COMPANIES. SYS-CON PUBLICATIONS, INC. IS NOT AFFILIATED WITH THE COMPANIES OR PRODUCTS COVERED IN WEBSPHERE JOURNAL.

WEBSPHERE® IS A REGISTERED TRADEMARK OF IBM CORP. AND IS USED BY SYS-CON MEDIA UNDER LICENSE. WEBSPHERE® JOURNAL IS PUBLISHED INDEPENDENTLY OF IBM CORP., WHICH IS NOT RESPONSIBLE IN ANY WAY FOR ITS CONTENT.

**SYS-CON
MEDIA**

ROB HIGH,

IBM DISTINGUISHED ENGINEER AND CHIEF ARCHITECT FOR
IBM WEBSHERE APPLICATION SERVER

Form & Function

Part 1: Principles are the basis of the WAS architecture – and its success

INTERVIEWED BY JACK MARTIN

Jack Martin, editor-in-chief of *WebSphere Journal*, recently spoke with Rob High, IBM Distinguished Engineer and chief architect for IBM WebSphere Application Server. In this exclusive two-part interview, High discusses principles of architecture, the evolution of customer requests into product features, and IBM's longstanding commitment to being on the front lines with their customers.

WJ: You head up architecture for the WebSphere Application Server, which is the heart and soul of WebSphere, correct?

RH: It's the foundation of the WebSphere-branded solution offerings, and quickly becoming so for most of IBM's software portfolio.

WJ: How do you spend a typical day?

RH: I'm responsible for the architecture of the WebSphere Application Server. A great deal of my time on any given day is spent trying to coordinate the variety of proposals, recommendations, innovations, and emerging technologies and standards that the architecture team brings forward. I try to assimilate what those things are, to provide guidance on following certain principles that we adhere to within the architecture, and to coordinate between the components and make sure there is consistency and not too much overlap. If there is overlap, we want to make sure that it is properly rationalized, that the pieces fit together and contribute to the overall value proposition. That's really the majority of what I do – reviewing those designs, giving the architects guidance, providing input to the planning process in terms of priorities, giving feedback to the other architectural organizations that are part of the WebSphere platform, and making sure that things assemble well within a rationale for what is important.

WJ: How does the WebSphere architecture board work?

RH: I've been responsible for architecture leadership going back probably 15 years now with IBM, starting with some-

thing called Financial Application Architecture. One of the things I learned through the years is that the best kind of architecture is one that is completely assimilated into the products being developed. The closer the architects are to the folks who actually write code – the more often the architects actually are writing their own code – the better the architecture is.

WJ: The more reality based...

RH: It's more reality based, and it has better influence in the product. If architects are separated into their own organization, they tend to lose reality, they lose relevance. So that's never worked. The way we structured this architectural organization is that the architects are responsible for their components, and the architecture board facilitates the enabling, communication, and coordination between all those activities.

WJ: Is the application server still a very organic process that is evolving?

RH: Absolutely. There is no end to the innovation and the evolution that you would want to bring to the table. Of course, part of that is continuing that innovation against the fundamental principles that we base our success on – namely, maintaining a degree of stability, maintaining a high degree of integrity, and delivering on the value proposition.

WJ: Is the board made up of only IBMers?

RH: Yes, from all over the organization. We have nine different development sites, all contributing to the application server. Some of them actually report to other organizations but have obligations to WebSphere Application Server development.

WJ: Being that the entire board is made up of IBMers, how does that board take customer input and then eventually productize it, where it becomes a feature of WebSphere?

RH: We have many channels of input. The architects themselves spend a lot of time in front of customers, helping solve problems, doing briefings, helping with the sales technical support and services teams. That's probably the most immediate influence that we have, because all of us end up developing our own sense of what's needed based on our individual experiences with these customers.

ABOUT ROB HIGH

Rob High is an IBM Distinguished Engineer and the chief architect for the WebSphere Application Server product family. In his 26 years of programming experience he helped to define, and then later refine, the basic concepts of container-managed component technology, which is now intrinsic to the EJB specification and implemented by WebSphere and other J2EE application servers. Rob holds a bachelor's degree in computer and information science from the University of California at Santa Cruz.

Second, there is a channel that delivers input from IBM's marketing and sales organizations. They aggregate their customer experiences and marketing requirements, and we capture those requirements and then meet weekly to screen and prioritize them. Eventually, that ends up driving some of the architectural design work that we have to do for new features of the product.

The third way is actually one that we have really been spending a lot of time trying to engage and enable. That is direct contact with our IBM Global Services (IGS) organizations. Since they are on the front lines working with customers in services engagements, they have a very good understanding of what the product can do, as well as what it needs to do to support those engagements. So that is becoming an increasingly important avenue for learning about driving innovation of the product.

WJ: Can a customer actually send in a request for a feature?

RH: Yes, and we get that directly when it is sent to one of the architects, but more often it's the channel that comes into the IBM sales marketing force that will capture those customer-specific requirements. Very often our line items end up being tagged to individual customers.

WJ: So you are doing customization for customers right at the core level of the application server?

RH: Yes.

WJ: If an auto manufacturer decided that a feature is really important, it sounds as if you would

aspect of your Toyota, and the people at Toyota would actually hear you and listen, and potentially incorporate the change.

RH: If you didn't like the way the window switches worked...

WJ: Exactly. "Those window switches should go the other way." All of sudden, the people that make the window switches are hearing it.

RH: Exactly right.

WJ: What is the time lag between a customer saying something and your hearing it? A day, week, a month?

RH: It really varies. We can turn around an interim fix that implements a capability for a customer within a month or two in some situations. For other situations, it goes in the bucket along with all the other requirements and gets rationalized against other initiatives that may color the way that we respond to the requirement.

WJ: But if everyone in the world wanted their own feature incorporated, you would stop shipping products. You would have millions of requests.

RH: And sometimes the requirements come in a somewhat unrefined state that may, for instance, create an integrity problem. Then we step back and look for the right integrity model, and we fit the requirement into a broader approach to address the issue. In other cases, it may turn out to be completely outside of the mission of what we are

"The best kind of architecture is one that is completely assimilated into the products being developed"

look at whether all auto manufacturers agree that it is important, and if so, you would go ahead and include that feature?

RH: Yes. It may be something that's in line with what we've already identified as the strategy, but it's an aspect we haven't thought about. Or maybe we will reprioritize the rate at which we will tackle it. Certain customers have a better perception of how to apply information technology than other customers, so they tend to carry a little bit more weight in their influence.

A lot of it has to do with understanding what the customers need and want, and what their relative perspectives are, and then balancing them. We also take into consideration their general market needs, as well as our own internal IBM strategies.

WJ: That's like being able to buy a car - say a Toyota - and saying you want to change some

currently doing, and it gets prioritized way out, or it gets rejected or transferred to another product group.

It's a fairly intricate process because it does involve an awful lot of review, refinement, and feedback - and that can be cumbersome sometimes, but it is necessary.

WJ: But you are listening and you are responsive. That's really the key.

RH: Yes. This focus on customer input is part of IBM's genetic makeup. I think we have learned how absolutely critical it is to be on the front lines with a customer, to understand their pain points, and be able to drive solutions that meet those needs. So there is a very short distance between the customer and the development organization. Even to the point sometimes where the IBM marketing and business strategy sides of the company have secondary influence.

I try not to differentiate between features and consum-

ability as part of the architectural process. I want the architectural process to not only deliver function, but also to understand exactly how that function is going to be used and how we are going to make it usable. We have to understand exactly where the deployment processing will take place, and how people are going to assemble, administer, extend, and leverage it. The architects understand how it operates. They write code, and if I had my druthers, I would be writing code as well. That's really the way it should be. People ought to understand the bricks and mortar of what they are building.

WJ: The reality of the thing, rather than dealing in abstractions...

RH: Yes. Architecture is about form and function. It's a matter of critical reality; it cannot be an ivory-tower, abstract activity.

WJ: Where do you see J2EE going over the next couple of years in relation to WebSphere? Will it hold onto the dominant position it has now? Do you see it morphing into something else?

RH: To the extent that J2EE has introduced architectures that were intended to be local-remote transparent and

OS to deal with?

RH: One of the principles that WebSphere adheres to is trying to strike a balance between portability and consistency across different platforms in leveraging the individual strengths and the differences of each underlying operating system platform. Each one has its own nuances.

WJ: Are application servers for HP, Unix, Windows, and Linux a little bit different, each one leveraging off each operating system?

RH: I think we are going to see differentiation there, and over the course of time that differentiation may grow. You will certainly see a lot of differences between Unix, Windows, iSeries, and zSeries. Already, in the WebSphere product, we enable for those differences and strengths.

A good example of that is Windows. The plug points for WebSphere allow that on Windows; we use the Windows ActiveDirectory user registry, and thus get tight integration with the underlying Windows operating system and other applications hosted there. Or we can plug in the IBM directory and basically share the same directory that we use for user authentication with iSeries or any other platform. WebSphere plugs right into the operating system and it has been adjusted on that platform to fit well within the

“The closer the architects are to the folks who actually write code – the more often the architects actually are writing their own code – the better the architecture is”

portable in a distributed system, it's been a good start. But it's going to have to take on more of a services-oriented architecture (SOA) flavor. I think SOA is the next major step in the evolution of computing and application design. Obviously, J2EE is going to continue to be a focus in services-oriented architecture. IBM will remain committed to Java, and J2EE will remain a dominant part of what we do.

WJ: Is IBM the biggest Java house in the world? You have more coders than anybody?

RH: That's exactly right. In fact, IBM develops more Java specifications and platforms than any other vendor in the world and makes more use of it than any other vendor in the world. Remember, we're the leader in the application server market.

WJ: Let's talk about platforms. WebSphere runs on every flavor of operating system – Linux, Windows, AIX, Unix, etc. When you look at it as an architect, in your opinion, what is the easiest

management structure of the platform. With the zSeries, it plugs into the zWLM for local classification and resource prioritization and deals with allocation for z/OS. It uses the Sysplex coupling facility for clustering.

WJ: Sysplex?

RH: Sysplex is an architecture within z/OS that enables high availability and reliability. It is fundamentally the core of the architecture on z/OS that enables things like being able to run an application for years without ever being taken out a service.

WJ: How are you leveraging the different operating systems that WebSphere runs on?

RH: The key is to find a balance between enabling completely portable application models and being able to exploit the differentiation that exists on each of the platforms. You enable application developers to create the application one time and deploy and redeploy it on different platforms – without ever having to change your appli-

cation.

We deliver to the customer the individual benefits that each of the platforms may offer, without compromising or degrading the experience for portability. So I talked about the zSeries, iSeries, and the differences we should expect to see over the course of time. We view all those differences as being potential quality-of-service differentiations for each of the applications that are hosted on each of those platforms.

This really goes back again to the core principle for WebSphere – which is to completely and cleanly separate out the concerns of application development and add value to the business by building business applications that really perform the business functions that different enterprises need. The goal is to separate all that from IT

achieve what you are looking for, that it's going to perform the way you want it to perform and produce the results you expect – every time you use it.

Then it has to have a high degree of flexibility and adaptability in different conditions. What we find is that application developers have a tremendous amount of imagination in how they handle middleware to achieve their objectives. Therefore the app server must be adaptable to the different ways people might use it; it can't be rigid – you can't have a finite number of scenarios that you adapt to and then close the door after that.

The other layer of principle that we really try to achieve is a strong separation between the concerns of the application developer and those of building a business application that achieves the domain requirements from the

“I think we have learned how absolutely critical it is to be on the front lines with a customer, to understand their pain points, and be able to drive solutions that meet those needs”

concerns in different IT data center structures or in formal computing scenarios. We can give people the ability to create applications and deploy them wherever they run a WebSphere Application Server, and they will get the same functional end result.

They can choose between running an application on WebSphere on one platform versus running that same application on another platform. The difference is the quality of service they get from the combination of WebSphere with that platform – and you get that transparently without changing the application.


WJ: Can you tell me what your architectural principles are for middleware generically and for WebSphere specifically?

RH: The core principle from a machinery standpoint is to maximize the design around strength, precision, and flexibility. Those are the three basic principles that apply to any kind of architecture, whether systems architecture, building architecture, or machine architecture. Your middleware must be strong enough to handle the kinds of workloads that are coming in, and be resilient and not break under the stress of those workloads. The stronger you are, the more people can depend on you to be the workhorse for their business.

There also has to be a great deal of precision in what they do. You need a pretty good understanding that your application, hosted in WebSphere, is going to actually

underlying IT infrastructure. The better job we do of separating those concerns and maintaining the high degree of isolation of the information system from the application design points, the more portable those applications are going to be, the more reusable the components of the application are going to be for different business scenarios, the less you are going to impact the productivity of that application developer in achieving the primary objectives, and the more WebSphere Application Server will be able to sustain and support the business objectives of the customer.

The more we are able to turn IT into a business asset and less a cost of doing business, the more we are actually helping them to fulfill our customers' business objectives, rather than just simply being a prerequisite to forming a business at all.

Having a clean separation there is absolutely key and something we try to drive into the programming model. We try to drive this principle into all the services and facilities we deliver. We even try to drive this into the administration model and the metadata associated with administration and integration management. This is the principle of encapsulation; isolation from the underlying infrastructure of the information system. We really try to drive the information systems infrastructure down into being a quality of service for the foundation, rather than have it surface as a functional feature of the application programming model. 

Using server clones to increase application performance and redundancy

Replicating J2EE Success

BY WARREN MACEK



ABOUT THE AUTHOR

Warren Macek is a consultant for Candle Corp. Since 1993, he has served as a senior architect and developer of Java and J2EE projects for large corporations in the financial services, manufacturing, and other industries. He has also contributed to an IBM Redbook on WebSphere Application Server J2EE performance tuning and monitoring best practices.

E-MAIL

warren@macek.us

Many developers who have designed, coded, tested, and deployed Java 2 Enterprise Edition (J2EE) applications have learned the hard way that not all J2EE features perform effectively under heavy production loads. One must consider a number of variables during each phase of the IBM WebSphere "build, deploy, manage" life cycle to ensure that enterprise J2EE applications are robust and scalable.

Failure to address key J2EE behaviors in a runtime environment can cause performance slowdowns for specific user requests once Web site traffic exceeds a specific performance threshold.

In order to support today's dynamic e-business infrastructures, developers require a knowledge of J2EE that goes beyond simply understanding how to create syntactically correct Java code. There is no single knob to turn or parameter to tweak that will automatically increase the performance of a Web site or scale a J2EE application. Rather, development teams must use proven best practices to support application functionality and business performance requirements. Failure to adopt common development standards that address the entire WebSphere life cycle may cause developers to spend costly resources recoding or re-architecting J2EE applications to achieve the desired performance.

This article explores best practices for using server cloning techniques to increase application performance

efficiency while providing the redundancy required to support business-critical applications.

Application Server Clone Overview

Optimizing a WebSphere Application Server for a specific application and infrastructure environment can take a significant amount of time and effort. Once a WebSphere Application Server and its J2EE application is tuned, however, organizations can quickly expand their Java environments by creating copies of these applications, called *clones*. The J2EE application clones each run as a separate instance inside their unique Java Virtual Machines (JVMs). Organizations can use clones to increase the performance of a specific server and provide the failover necessary to support business-critical applications. The WebSphere Application Server administration system allows users to create and manage cloning features.

The original tuned application server has a template, called a *group*,

that acts like a stamp to create additional clones. Clones have the same structure as the original server group and may include servlet engines, Enterprise JavaBean (EJB) containers, servlets, etc. Server clones have the same properties as traditional J2EE objects: they receive requests from HTTP Web servers, automatically process requests, and are connected to databases. Additionally, if one clone fails, WebSphere Application Server automatically routes commands to a second clone.

There are two cloning methods: vertical and horizontal. Each method serves a distinct purpose and has a unique set of implementation considerations. Requests are routed to each clone through the HTTP Web server, along with the WebSphere Application Server plug-in. The decision to route requests via round-robin or load balancing can be made at runtime. Also, the WebSphere Application Server plug-in will maintain session affinity between application server instances and will provide failover support if one of the instances fails.

Vertical Clones

Vertical cloning refers to the process of creating multiple copies of an application server on a single physical machine. Through vertical cloning, users can increase the performance of a single machine without increasing the complexity associated with managing multiple JVMs. The inherent concurrency limitations within a single JVM process typically cannot capitalize on the full processing power of application servers, especially when a single JVM is deployed on a large multiprocessor machine. To overcome this challenge, vertical clones are installed to enable organizations to use the full CPU power and memory of the application server hardware (see Figure 1).

Organizations typically deploy between three and five clones on a single machine. Because each clone uses computing resources, adding too

many clones to a single machine will consume too much memory and CPU power, and actually hinder application performance. Each J2EE application requires 1GB of heap space; running three clones on a single machine increases the memory requirements to 3GB. It is important to note that vertical clones create a single point of failure because the clones are located on a single machine. If there is a hardware or database connectivity problem, vertical cloning is not structured to provide failover or redundancy support on other machines.

There are, however, significant advantages to vertical cloning. The ability to centrally manage vertical clones enables organizations to quickly add processing power without additional management requirements. Central management enables administrators to make a single update to the server group, which is applied automatically to each vertical clone. This feature enables administrators to quickly and efficiently control all JVM processes running on a single application server. It is important to note that the server group is a representation of the multiple server clones, but is not a working application server and does not run inside its own JVM process. As such, the server group does not have any application server capabilities.

Horizontal Clones

Horizontal cloning addresses the critical requirement for application failover. Rather than maintaining clones on a single machine, horizontal cloning enables organizations to deploy clones of a tuned application to multiple machines (see Figure 2). If one server shuts down, an application clone on a second server will automatically stand in for the unavailable server. The placement of application clones across the enterprise also enables organizations to distribute performance load across multiple machines. By creating additional J2EE objects, horizontal cloning also provides additional throughput across the enterprise.

Placement of horizontal clones across the enterprise is an important consideration. Organizations that do

not appropriately address the role of EJBs in a J2EE environment may experience performance delays that may be exacerbated, in part, by horizontal cloning. IT managers must, for example, architect and deploy EJBs to ensure that user requests are calling only servers that contain EJBs, or ensure that specific requests are routed to the appropriate EJBs. Similarly, an application server architecture that directs requests to EJBs that can be executed by smaller servlets will cause EJBs to be overtaxed, while other J2EE infrastructure, such as servlets, will be underutilized.

Because each horizontal clone is located on a different physical server, communication among the servers consumes network resources. As such, the number of horizontal clones may negatively affect application performance. The impact of increased network traffic associated with horizontal clones will depend on application and hardware variables for each organization.

Additionally, management of horizontal clones can be complex. Unlike vertical cloning, where administrators make changes to the clones through groups, administrators must make updates and changes to each horizontally cloned machine group that represents the application. IT administrators typically first make changes to one machine in a group. The updated file from the first machine is then transferred to the other machines that are horizontally cloned. In this case, IT managers need to update only unique machine-characteristic information included in the file – such as server names and directory paths.

In real-world applications, organizations typically deploy both vertical and horizontal clones. Organizations should typically install vertical clones first to increase performance. As noted, it is important to monitor the impact that each additional clone has on CPU and memory overhead. After optimizing performance on each server using vertical cloning, organizations should then deploy horizontal clones.

The failover and redundancy that horizontal clones provide is integral to supporting today's business-critical

WebSphere JOURNAL Advertiser Index...

COMPANY	URL	PHONE	PG
CANDLE CORPORATION	WWW.CANDLE.COM/WWW1/PATHWAY	800-488-4246	2,3
CYANEA	WWW.CYANEA.COM/WSOJ/NOMOREMADNESS/HTML877-CYANEA8		15
H&W COMPUTER SYSTEMS	WWW.HWCS.COM/WSO1.ASP	800-338-6692	51
KENETIKS	WWW.KENETIKS.COM	888-KENETIKS	19
MERCURY INTERACTIVE	WWW.MERCURYINTERACTIVE.COM/OPTIMIZEJ2EE800-837-8911		9
MQSOFTWARE	WWW.MQSOFTWARE.COM/QNAMI	800-998-2858	21
NETIQ	WWW.NETIQ.COM/SOLUTIONS/WEB	408-856-3000	11
PROLIFICS	WWW.PROLIFICS.COM/WEBSERVICES	800-675-5419	29
QUEST SOFTWARE	HTTP://JAVA.QUEST.COM/JPROBE.WDJ	800-663-4723	39
SYS-CON MEDIA	WWW.SYS-CON.COM/2001/SUB.CFM	888-303-5282	41
LINUXWORLD CONFERENCE & EXPO	WWW.LINUXWORLDEXPO.COM	800-657-1474	34-35
TRILOG GROUP	WWW.FLOWBUILDER.COM	800-818-2199	52
WEB SERVICES EDGE EAST 2004	WWW.SYS-CON.COM	201-802-3069	42-49
WILY TECHNOLOGY	WWW.WILYTECH.COM	888-GET-WILY	5

WebSphere JOURNAL Coming Next Month...

INTERVIEW

IBM's Rob High, Chief Architect for the WAS Product Family, Discusses the Customer's Influence on Product Architecture, Part 2

INTERVIEWED BY JACK MARTIN

WEB SERVICES

Describing and Discovering Web Services in UDDI

BY YEN LU & ROB BREEDS

INTEGRATION

Community Integration with WebSphere Business Integration Connect

BY SCOTT SIMMONS

ILLUMINATING WEBSHERE

J2EE Pattern Frameworks Provide Template for Flexible and Modular Architecture

BY LLOYD HAGEMO & RAVI KALIDIINDI

Web environments. Also, by vertically and horizontally cloning J2EE applications, organizations will be able to maintain 24x7 uptime when upgrading or redeploying new J2EE code. Horizontal cloning enables organizations to shut down one application server and have all requests automatically routed to a second application server. When the new machine comes back online, the same process is performed on the next horizontally cloned machine.

The Role of the HTTP Plug-in in Server Cloning

The WebSphere Application Server HTTP plug-in plays an integral role in automatically routing requests between servers, maintaining sessions, and ensuring session affinity

when requests are rerouted among machines. The WebSphere Application Server plug-in is generated and deployed to an organization's unique HTTP Web server. The WebSphere Application Server plug-in directs requests from the HTTP Web server to the appropriate application server. The plug-in uses an XML configuration file (plugin-cfg.xml), generated by the administrative server, to determine information concerning the WebSphere domain.

Each application server plug-in is assigned a weighted value, used to determine which application server will receive the next request. When the HTTP Web server is started, the WebSphere plug-in reads information from the plugin-cfg.xml file and stores an assigned server ID, along with a list

of servers in the system. When a request enters the plug-in, it passes through the URL it receives from the client browser (e.g., <http://www.candle.com/application1/servlet1>). It separates this information into two pieces: the virtual host, e.g., www.candle.com:80; and the Uniform Resource Identifier (URI), e.g., [/application1/servlet1](http://www.candle.com:80/application1/servlet1).

The HTTP plug-in then looks in the XML file for a matching URL and URI. If successful, the plug-in then searches the XML file for a route entry containing the `<UriGroup>` and the `<VirtualHostGroup>` just retrieved. This entry is used to identify the cluster that will handle the request (see Listing 1). In this example, the cluster is "CandleCluster1." The cluster contains

LISTING 1

```
<UriGroup Name="Application 1 Name">
  <Uri Name="/application1"/>
</UriGroup>

<VirtualHostGroup
  Name="default_host">
  < VirtualHost Name="*.80"/>
  < VirtualHost Name="*.9080"/>
</VirtualHostGroup>

<RouteVirtualHostGroup="default_host"
  UriGroup="Application 1 Name">
  ServerCluster="CandleCluster1"/>

<ServerClusterName="CandleCluster1">
<Server CloneID="a411s2v1"
  LoadBalanceWeight="3"
  Name="CandleClusterServer1">
<Transport Hostname="128.7.30.21"
  Port="9081" Protocol="http"/>
</Server>
<Server CloneID="b411t2h1"
  LoadBalanceWeight="4"
  Name="CandleClusterServer2">
<Transport Hostname="128.7.30.21"
  Port="9081" Protocol="http"/>
</Server>
<ServerClusterName>
```

either a stand-alone server or WebSphere server clones. Once the cluster is identified, the plug-in must route the request to the appropriate server.

If the client request includes an HTTP session, the CloneID is retrieved from the end of the session ID. The session ID is checked against the CloneIDs in the server cluster until a match is found. The request is then routed to the appropriate server. If the request has no session ID, the plug-in will alternatively route the request to the next server based on the routing algorithm chosen. This process outlines how the plug-in and its configuration file determine server routing requests and maintain session affinity.

Conclusion

The importance of taking the time to tune an application server to support specific business requirements cannot be overstated. Once an application server is tuned, however, organizations will be able to use cloning to quickly scale the application server to support enterprise-wide computing requirements.

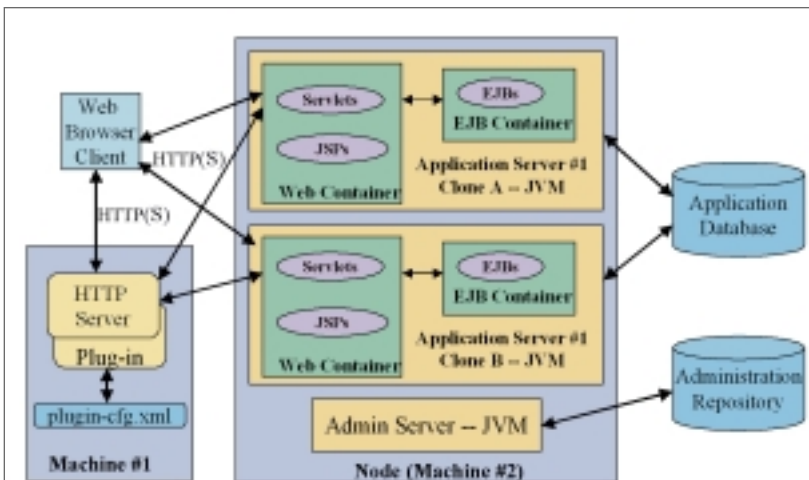


FIG. 1: VERTICAL CLONES ENABLE ADMINISTRATORS TO ADD MULTIPLE J2EE OBJECTS TO A SINGLE APPLICATION SERVER TO INCREASE PERFORMANCE

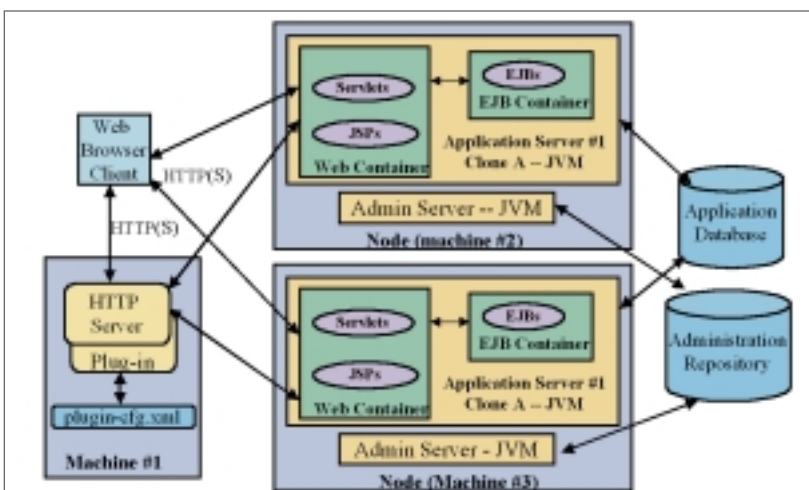


FIG. 2: HORIZONTAL CLONES PROVIDE APPLICATION FAILOVER AND REDUNDANCY



Legacy Integration in Enterprise Class Applications

It's not just about the technology

BY UDAY KUMAR



ABOUT THE AUTHOR

Uday Kumar is the vice president of Technology for Covansys Corporation, a U.S.-based corporation that specializes in providing custom application builds and integration services to its customers. He consults as a chief architect and guides development teams building enterprise architecture solutions using J2EE and .NET technologies and also runs the E-Business practice for Covansys' Western region. His professional interests include operating systems research and virtual machine implementations. Uday is currently researching the comparative merits of virtual machines such as Parrot, Microsoft's CLR, and the JVM. He builds Perl applications as a hobby.

E-MAIL
ukumar
@covansys.com

Large organizations have a considerable investment in their legacy applications by virtue of the fact that they have a sizable IT history. These legacy applications are a smorgasbord of mainframe and pure client/server applications from the '70s and '80s. Today's thrust toward a distributed application development and runtime environment mandates the adoption of enterprise-class component-based architectures and implementations.

The increasing adoption of J2EE-based technology platforms is an indication of the need being felt by enterprises across vertical domains and technology platforms. J2EE enterprise architects are constantly faced with the challenge of making their multitier architectures interact with the rest of the constituents in an organizational ecosystem. This includes not only the above-mentioned legacy applications but also other types of enterprise architectures based on .NET and EAI middleware.

This article takes a look at some of the techniques that my team analyzed for implementing integration with legacy mainframe applications when we built an IBM WebSphere-powered, J2EE-based application for a large health care provider in the Bay area. The article focuses on IBM mainframe-based legacy applications. A future article will deal with issues surrounding the interaction between J2EE and non-J2EE-based enterprise architectures.

The Business Problem

The health care provider has multiple medical facilities scattered over a wide geographical area. Each facility has both primary care physicians and specialists, and each patient would first meet with a primary care physician, who would perform a diagnosis and then, if needed, create a referral to a specialist.

This is an important business process for the health care provider, as the specialist's time is extremely valu-

able and a wrong referral could result in unnecessary overbooking of the specialist's time, while not creating a necessary referral would result in the patient not receiving the proper care. In either case, there would be a deterioration in patient care, which is an unacceptable situation.

The business process of performing a diagnosis and consequently creating a consult that can be referred to a specialist needs to be performed in an efficient and integrated manner, thus leading to improved patient care. In this context, integration involves providing the specialist's staff with access to the referred consults so they can book the specialist's time for an appointment with the patient. In addition, the primary care physician's staff needed the ability to book the specialist's time directly, a process known as direct booking.

There were technological reasons why the above functions couldn't be performed in an integrated and seamless fashion.

Technology Issues

In the past, the business process described above was performed by two different types of applications: the first a mainframe-based application used to create consults and referrals to specialists, and the other a Lotus Domino-powered client/server application that had an intuitive GUI. The applications had similar functionalities but operated in their own hardware and software environments without any intercommunication.

Some medical facilities had adopted the Lotus Domino-based application, while the rest were still powered by the mainframe application. The specialists' time-scheduling application was on an IBM mainframe, and while an electronic communication was possible between the mainframe-based physician referral application and the scheduling application mentioned earlier, no such integration was in place between the Lotus Domino-based physician referral application and the specialist scheduler.

Any exchange of information between these applications was possible only through manual reentry of data. In addition, the Lotus Domino platform did not scale well when used in an enterprise-wide environment.

Architecture

These issues related to the technology and the enterprise nature of the business problem led us to a J2EE-based architecture for the application. Our objective was to replace the Lotus Domino-based application and the mainframe application with an *n*-tier architecture powered by IBM WebSphere Application Server.

The architecture would provide certain basic services split across multiple layers:

- **Presentation tier:** User-interface generation, screen-flow navigation, session management, etc.
- **Application tier:** Business logic hosting, authentication services, workflow processing, etc.
- **Infrastructure tier:** Printing services, e-mail services, integration services, object persistence services, etc.,
- **Data tier:** Enterprise data source connectivity services

The data tier was on the mainframe, as it was an enterprise standard and needed to serve as a repository for many other applications that reside within the health care organization's IT portfolio.

The focus of this article will be on the integration services provided as part of the infrastructure tier. There were many different integration points that were required for making such an application work successfully, and the architecture team faced the challenge of designing an integration architecture that would scale and perform well in the face of increasing load.

Integration Tier

The term *integration*, as used in this article, describes a link to legacy mainframe applications that reside on an IBM mainframe and are powered by CICS (Customer Information Control System) – which is the OLTP (Online Transaction Processing) monitor from IBM. The link described is a two-way connectivity that allows information to flow from the Web portion of the application to the mainframe and vice versa.

The integration is at a transaction level as well as a data-tier level; this means that data-level integration – direct to the mainframe database – as well as CICS transaction invocation from the Web tier is involved.

The integration services provided as part of the infrastructure tier are complex and varied enough to mandate a separate tier of their own in the multitier architecture. Current literature from vendors, as well as technical literature, normally discusses adapters as the one solution for all legacy integration needs.

In the J2EE world this has been incorporated as the JCA standard, which looks at integration problems from a connector perspective. Connectors and adapters are related, with adapters considered to be a mechanism to do some type of conversion or adaptation of protocols, for example, when connecting an ERP implementation to a legacy application in the enterprise. Connectors, on the other hand, provide a connection between applications and do not necessarily have to perform any other protocol conversion.

However, in the arcane world of mainframe integration a simplistic approach centered around adapters does not work, because in a majority of cases the integration occurs between two custom-built applications. Adapters and connectors are normally built for connecting a COTS (com-

mercial off-the-shelf) product to a custom application. To integrate two custom-built application components, a well-engineered integration tier is required.

The technologies that go into this integration tier are fortunately standardized, especially when integrating with an IBM mainframe environment and a CICS subsystem. IBM provides several standard technologies to integrate Web-based applications into its mainframe environment; these are referred to as “standard” in this article – not from an industry standards perspective, but more from the accepted or best practices way of doing things in an IBM mainframe environment.

Integration Scenarios

Given the above business and technology factors, our architecture team spent a few weeks analyzing the different options and came up with a few scenarios for performing the integration. The scenarios are not distinguished by the technology options but more by the specifics of the target platform.

SCENARIO 1

Figure 1 depicts the first integration scenario we analyzed. It involved the usage of CICS Transaction Gateway (CTG) APIs from Java classes to integrate with the CICS subsystem. While this is considered to be a standard way to integrate with CICS, it requires engineering on the mainframe side. As depicted in Figure 1, CICS systems normally define maps, which are a declarative way of specifying the presentation details of the user interface. The user interface is usually a 3270 terminal or a terminal emulator running on a modern-day PC.

The engineering on the mainframe would involve creating mapless versions of the mainframe programs, which can then serve as APIs for the Java classes to invoke via the CTG interface. Thus in Figure 1 you will see the mapless versions of both the APEX and Booker transactions to which there is a CTG interface. The data tier on the mainframe is completely insulated from the Java classes themselves, thus allowing the coexistence of a variety of technologies such as VSAM and RDBMSs.

The Booker application also exposes an API via a middle tier hosted on an IBM AIX platform. This is another example of complexity in action: this API predated the Java API and was in existence when we did the analysis. This involved running a version of CICS on the AIX box, with CICS Distributed Program Links (DPLs) between the mainframe CICS programs and the AIX middle-tier COBOL programs. The external world would then use the APIs exposed by the AIX CICS system to connect to the mainframe Booker application logic.

SCENARIO 2

Figure 2 details the second integration scenario. In this case we were exploring the possibility of using a DB2 instance hosted on the existing mainframe environment as the data tier for our Web-based application. While most Web development projects look on the mainframe environment as a monolith that needs to be covered and confined behind the safe walls of some integration API, the reality is that the mainframe environment does provide one of the most robust ways of storing corporate data. The level of operational support that is available for a data

administrator in the mainframe environment is unparalleled in most modern server farms.

This scenario raises some interesting questions that are very relevant to any enterprise architect: Where does the business logic reside? In Scenario 1, the business logic resided partly on the mainframe (on the APEX application) and partly in the Java classes (either EJBs or normal JavaBeans) deployed on the WebSphere Application Server. In the second scenario, the APEX mainframe logic has been completely reengineered and is now resident on the application server tier.

The APEX application continues to coexist with the Java application. The other interesting question that arises is how to keep the business logic in these two locations in sync. There is no easy answer to this question, and only a

This would give the Web tier application development team impressive advantages in productivity because changes to the data tier are now made on a development server right on the premises of the development team, as opposed to a mainframe served by its own acolytes in distant data centers. Once again, in accordance with the theme of this article, the issue at hand has less to do with technology and more to do with process-related aspects of enterprise architecture.

In addition to the problem of synchronizing business logic, this scenario also throws up the problem of synchronizing data across the two different databases. Thus, the architecture had to evolve to allow for a synchronizing bridge. This is a great example of how architectural tiers evolve in practice – a whole topic in itself!

SCENARIO 4

In Scenario 3 we came up with a variation of Scenario 2, in which we had created a separate database for the Web application. In Scenario 4, the data tier for the Web application shifts to the mainframe environment again, but it is a separate instance from the APEX data tier. This difference between Scenario 4 and Scenario 2 is significant. Data synchronization issues still need to be resolved. We developed Scenario 4 because of the requirement at this client location to preserve the investment in the mainframe data tier – not just the investment in hardware and software, but also the investment in people.

The client had a very skilled database design and administration group that was expert in designing and maintaining large

mainframe databases. So the rationale here was that this group would maintain the data tier of any new application that was to be built, whether Web-based or not, and hence it made sense to have the database instance on the mainframe. At the same time, creating an instance that was separate from the APEX instance eliminated some of the delays in ringing in changes to the database schemas, for example. This is another example of how the unfolding of scenarios depends on the process- and people-related needs of the enterprise.

SCENARIO 5

However, we couldn't entirely dispense with the idea of Web services, and that scenario is explored in the last integration option, Scenario 5. The idea here is that the integration mechanism we built for this application could potentially be used by other applications that need access

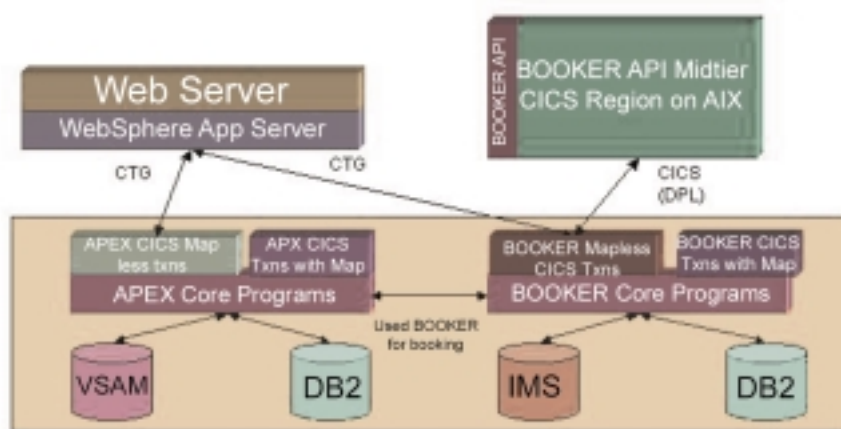


FIG. 1: THE FIRST INTEGRATION SCENARIO

close correlation between the different maintenance team activities, coupled with effective development processes, can solve the problem. This is another instance where technology alone doesn't solve enterprise architecture issues.

SCENARIO 3

The catch in the previous scenario was that while the mainframe does indeed offer a highly available and redundant data tier, it is also a cumbersome beast to deal with for development teams used to the free world of Unix- or Windows-based relational databases. There is a technical and administrative overhead involved in dealing with mainframe-based applications, including the base operating system and database, that causes modern Web development teams to cringe. Hence Scenario 3 looks at placing the Web application's data tier on a relational database instance that is outside the mainframe environment.

“Current literature from vendors, as well as technical literature, normally discusses adapters as the one solution for all legacy integration needs”

to the Booker application's services. This leads us naturally to the idea of a service-based architecture wherein the functionality of the Booker application is wrapped into some form of a service and exposed to the outside world.

This is the most hyped solution in the technology landscape today – and unfortunately, also the one that has the least chance of success. This is because creating a Web service wrapper for a mainframe application involves some significant engineering on the mainframe side, which results in process- and technology-related delays that are not always amenable to project deadlines. And

tion solution for this customer. The variation introduced was to engineer the application in such a way as to pave the way for retiring the APEX application over a period of time. The customer has multiple medical facilities, all of which use APEX and Booker on a daily basis to service the patients.

The deployment plan was to roll out the new Web application gradually, starting with a select set of medical facilities that already had experience in using the Lotus Domino-based application. There would be a period of time when there would be two sets of medical facilities:

“In the arcane world of mainframe integration a simplistic approach centered around adapters does not work”

when there are alternative ways (some of which are explored in the earlier scenarios) to expose mainframe application services, the need to provide a Web service interface is not very compelling.

There is also the very important issue of performance, which is always a consideration for an enterprise architect. Providing a Web services wrapper implies multiple conversions on the transaction path between the Web tier and the data tier. Since the specific enterprise integration problem addressed here involves mainframe-based data tiers, this would lead to unacceptable performance meas-

one that had switched over to the Web-based application and the other still relying on the APEX application.

Deploying the new Web application in a medical facility implies acquiring PCs and installing them in the offices where primary care physicians would examine their patients. This was one of the main reasons why a staggered deployment plan was formulated.


Conclusion

There is no such thing as a perfect integration solution; there are only optimal solutions that are tailored to fit a particular situation. That is the reason why the prevailing belief that adapters are the universal solution to all integration problems leads to a serious underestimation of the effort involved in connecting to custom legacy applications.

Custom legacy applications need custom integration solutions as well; sometimes instead of a neatly packaged adapter, the solution is a collection of technologies that must be cobbled together to efficiently access the legacy application.

Thus development teams chartered with developing integration solutions need to pay considerable attention to fitting together these disparate technologies that make up the overall integration solutions, so that the overall performance is acceptable.

This article described the strategies that our team formulated to solve the integration needs of our customer, which was the first step in the creation of the application and integration architectures.

The key theme of this article illustrated that our integration scenarios had to take into account issues such as the customer's deployment needs, the mandate to preserve investment in existing mainframe applications and leverage the prevailing database expertise in the customer's IT division, and the reduction of the development cycle. The point being that it is often not just technology issues that influence the evolution of enterprise architectures in the field. 

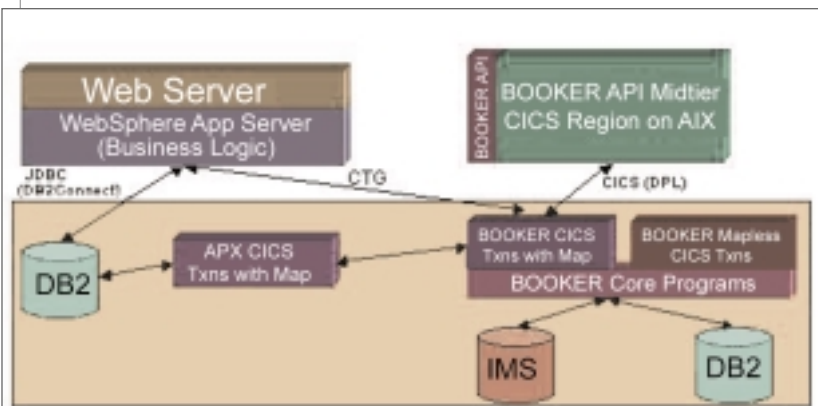


FIG. 2: THE SECOND INTEGRATION SCENARIO

ures for the application. Nonetheless, this could become a viable solution at a later time when the organization decides to move away from the mainframe environment gradually; such is not currently the case with our customer.

SELECTED OPTION

After creating and presenting the different scenarios to the client organization, our team of architects selected a variation of Scenario 2 as the most appropriate integra-

Security isn't a one-time event in the application development life cycle

Viewing Security as a Process

BY CALEB SIMA

According to a 2002 study by the National Institute of Standards and Technology (NIST), buggy software costs the national economy \$60 billion, more than a third of which could be saved through improved software testing alone.



ABOUT THE AUTHOR

Caleb Sima, founder and CTO of SPI Dynamics (www.spidynamics.com), is widely known in the Internet security community for his expertise in penetration testing and his ability to identify emerging security threats. He began his security career at the S1 Corporation, before joining Internet Security Systems as a member of the X-Force.

E-MAIL

csima@spidynamics.com

Delivering quality applications to the market has become a mandatory requirement as enterprises learn that the cost of fixing security defects after deployment is almost 15 times greater than detecting and eliminating them during development.

Developers, QA professionals, and corporate security experts worldwide are beginning to realize the urgency of developing and delivering secure Web applications at all phases of the life cycle. According to the NIST study, the most cost-effective way companies can increase IT security is to buy and build secure applications. Many large software companies have already begun adding numerous security improvements to their development environments to assist developers in their endeavor to develop attack-proof, secure code.

70% of Hacker Attacks Target the Web Application Level

Industry analysts estimate that over 70% of recent security breaches occurred at the Web application level. Many are due to the exploitation of security defects created with-

in the code during the development phase. Historically, the network layer of corporate systems has had the greatest vulnerability to security breaches, but this is no longer the case. Despite corporate firewalls, attackers today can easily gain unauthorized access to unprotected Web applications, and to confidential information. Now hackers can penetrate Web applications given the smallest opening in the code of a corporate Web site.

And reports of Web hacking incidents continue to grow. For example, as recently reported by Kevin Poulsen in *SecurityFocus*, "At least 1,000 automobile shoppers who submitted online credit applications to any of 150 different automotive dealerships around the U.S. had their personal and financial details exposed on a publicly-accessible Web site. The exposed site was an administrative page at the Tennessee-based hosting company Dealerskins, a firm that provides turnkey Web solutions to automotive dealerships. The page – which was not password protected and included no warnings that it was private – allowed visitors to view, in reverse-chronological order, all of the infor-

mation that had been typed into Web forms on Dealerskins-hosted sites, like autocentersdirect.com and courtesyflm.com."

Additionally, on October 21, 2003, New York State Attorney General Eliot Spitzer announced an agreement specifying that Victoria's Secret Direct, LLC, must compensate New Yorkers whose personal information was unintentionally left accessible on the Internet. Under the terms, the company was ordered to implement and maintain a series of programs to improve Web site security. Victoria's Secret Direct must also pay costs and penalties of \$50,000 to the state of New York. In a statement about the incident, Attorney General Eliot Spitzer said, "A business that obtains consumers' personal information has a legal duty to ensure that the use and handling of that data complies in all respects with representations made about the company's information security and privacy practices."

Removing Application Vulnerabilities During Development

Even in mature development organizations, one of the most commonly overlooked areas of the application development process is Web application security. This oversight occurs primarily for two reasons. First, the methodology of Web application security is relatively new and unknown in the market. Security has traditionally been focused on the network and server layers of an application's architecture. However, even architectures that have secure network and server layers are exposed to attacks if the application layer is insecure.

Second, developing secure Web applications is usually an afterthought for most development organizations. Since security is not directly related to functional requirements, users do not focus on it and developers generally fail to put in the neces-

sary time to ensure that applications are secure. In addition, some developers do not feel that application security is necessary. They think that if they use IIS or implement their applications behind a firewall, they will be secure from malicious attacks. Even developers who do see the importance of Web application security usually view it as task that is performed as part of the QA process. As a result, many Web applications may be functionally rich but vulnerable to unwanted intrusions and attacks at the application layer.


Furthermore, many development organizations view security as an event to be completed once during the development process. In these cases, security becomes the responsibility of a single organization like the quality assurance or internal audit departments. Once these departments sign off on an application, it is deemed secure by the organization. However, Web applications are not static systems. Changes to Web applications create risks, so what was once secure is now vulnerable. If security is viewed as a single event, a vulnerability that enters the system after the audit is performed will go undetected.

The process of uncovering coding vulnerabilities manually is the Achilles' heel for application developers. Yet, in an ever-changing, dynamic environment like the Web, it is essential to maintain a security mindset. After all, attackers are constantly updating their approaches and techniques – and developers must follow suit.

Web application security does not need to be an oversight, an afterthought, or viewed as a one-time event. Rather, it should be viewed as a process and incorporated throughout the development life cycle to ensure that Web applications are built securely. It should be incorporated into the practices of every team member associated with the development and operations of the Web application.

The intricacies involved in Web application development have driven the recent introduction of new application-level automated security tools,

providing built-in application testing, remediation, and secure development practices. Easily integrated into software development environments, these tools allow developers to view the application through the eyes of a security expert.

In short, these tools can analyze any Web application, identify potential security flaws, and supply the developer with the latest information necessary to resolve security issues before attackers are able to penetrate the system. By implementing Web application assessment practices at the initial steps of the development life cycle, enterprises can remedy vulnerabilities before Web applications are placed into production, saving time and potentially billions of dollars. 

MANUAL VS AUTOMATED AUDITS

Moody's KMV, a provider of quantitative credit analysis tools, developed their own return-on-security-investment model by comparing the time required to assess Web applications manually versus an automated solution, according to Jim Reavis in *CSO Informer*.

Moody's KMV has over 10 significant Web applications, each having 3–4 significant updates annually, according to Mario Duarte, MKMV's security manager. "Web-based product audits must be conducted. The question is how?" said Duarte. In order to answer this question, Duarte selected an actual MKMV Web application and performed both a manual and an automated audit. The findings were compelling: a manual audit took 52 hours, while the automated audit took 10 minutes, plus an hour for documentation. The findings from the two methods of assessment were of comparable quality.

Manually auditing all Web applications would require two additional full-time employees, according to Duarte. Without increasing staff or implementing automated assessment software, MKMV would be limited to auditing a maximum of 25% of its applications per year, slowing business productivity and increasing risk for the organization.

In the final analysis, automating Web application assessment provides MKMV a net savings of nearly \$200,000 per year, while offering complete coverage of MKMV's Web assessment needs, the report concludes.

For additional information on the risks developers face when developing Web applications, download a free white paper, "Building Web Application Security into Your Development Process," located at www.spidynamics.com/whitepapers/Webapp_Dev_Process.pdf.

Using WebSphere Studio Application Developer's Code Formatter



ABOUT THE AUTHOR

Kulvir Singh Bhogal works as an IBM Software Services for WebSphere consultant, devising and implementing WebSphere-centric solutions at customer sites across the nation. He has over 50 patents pending in a myriad of technology areas.

E-MAIL

kbhogal@us.ibm.com

Formatting Your Code Automatically

BY KULVIR SINGH BHOGAL & KWANG SIK KANG

We all have our preferences for how we like to have code formatted. Even when working in a team, you commonly see a myriad of coding styles. Manually formatting someone else's code to meet your coding style preferences can be a tedious process that can leave you cursing at inanimate objects. A little-known but highly useful feature of IBM WebSphere Studio Application Developer version 5 is the ability to use the Code Formatter to format Java code.

To specify the preferences for code formatting, go to Window>Preferences to open the Preferences window. From there, choose the Java>Code Formatter, as shown in Figure 1.

The options for the Code Formatter include options for formatting new lines, line splitting, and specifying coding style options.

New Lines

The New Lines tab includes selections that allow you to format code using the following options:

- Insert a new line before an opening brace
- Insert new lines in control statements
- Clear all blank lines
- Insert a new line between "else if"

- Insert a new line inside an empty block

You can preview the formatted sample code inside the Preferences window formatted according to the option selected. For example, Figures 2 and 3 show before and after screen shots of choosing the "Insert a new line before an opening brace" option.

Line Splitting

The Line Splitting tab contains an option that allows you

to specify the maximum line length in characters. Based on the specified line length, the Code Formatter will automatically split your code accordingly. By default, the maximum line length is set to 80 characters.

Style

The Style tab allows the programmer to set preferences for:

- Compact assignment
- Indentation represented by a tab

Compact assignment simply means that assignment code such as "a = b" will appear as "a=b" (i.e., no space preceding the assignment operator) when the option is turned on.

Using the Style tab, you can also specify the number of spaces that represent a tab. By default, the number of spaces representing a tab is set to 4.

Invoking the Code Formatter

You can format code in your Java editor in a number of ways:

- Selecting Format from the editor's context menu
- Using the shortcut of Ctrl+Shift+F
- Selecting Source>Format from the menu bar



ABOUT THE AUTHOR

Kwang Sik Kang, consultant for IBM's Software Services for WebSphere, has extensive experience in providing J2EE and messaging-based enterprise solutions.

E-MAIL

kkwang@us.ibm.com



FIG. 1: SETTING THE CODE FORMATTER'S PREFERENCES

Specifying What You Want Formatted

By default, if you do not have anything selected, then all of the content in the current Java editor will be formatted when you invoke the Code Formatter. If you would like to format just a portion of your code, select only that portion of your code.

Conclusion

WebSphere Studio Application Developer version 5's built-in Code Formatter allows you to format Java code according to your personal preferences. In this article, we introduced you to the Code Formatter and the various preferences you can set for it. As the exercise of writing code within a team commonly involves combining code from developers with different style prefer-

ences, the Code Formatter is a handy tool. 



FIG. 2: THE PREVIEW PANE BEFORE THE "INSERT A NEW LINE BEFORE AN OPENING BRACE" OPTION IS APPLIED



FIG. 3: THE PREVIEW PANE AFTER THE "INSERT A NEW LINE BEFORE AN OPENING BRACE" OPTION IS APPLIED

Are All Systems Go?

How to assess the production readiness of your WebSphere application

BY MAX KING & YATEEN DHARESHWAR

Your team has just spent several months hammering out an enterprise-critical application and it feels as if you've been on the hot seat forever. The once vibrant and enthusiastic development team now resembles the cast of "Thriller" as they burn the midnight oil night after night.

Finally, the finish line is in front of you – you just need to cross it and relax into maintenance mode. The client and management wait in anticipation. You flip the go-live switch, logs start to fill up, users start to log on, Operations pats you on the back, and all systems are go – or so you thought. How long will it be before you get that all-too-familiar call from the customer? What exactly went wrong? Three weeks of root-cause analysis and four specialist consultants later, the executive summary is: "D'oh!"

Do these sound familiar?

- "The server license expired."
- "The LDAP server has been taken down for maintenance."
- "The network is not configured to support that load."
- "The messaging system was never designed for that format."
- "We deleted the running log files to create space."
- "Oh, that module is supposed to support more than one user? I didn't test that eventuality."
- "Our whole application revolves around this singleton object that never got instantiated!"

These are just a few of the many possibilities.

Production Readiness

Plan to avoid these surprises – along with the corporate politics and the rounds of firefighting that follow. By preparing for production readiness, you can get a handle on the many issues involved in going live with an enterprise-level J2EE application. Remember that production readiness is an all-encompassing state that means different things to different teams, from going live to releasing code to ensuring for redundant backup systems.

This article provides guidance on how to assess your production readiness and avoid the scenario outlined ear-

lier. Having helped several organizations certify the production readiness of their sizable applications, we'll share our recommendations for how to initiate a project; set proper roles, processes, and expectations; assess risk; and roll out a successful application.

We'll begin with some essential words of advice:

- Start preparing for production readiness early on – during the planning and design stages. It is never too early to lay the foundation for successful application deployment.
- Conduct a certification process that includes multiple group-level checklists and a review of the application architecture and representative code sections by someone other than the original programmer.
- Consider setting up a cross-disciplinary task force and/or utilizing an external services team to conduct the certification, lending an unbiased outsider view. In our experience, it is during this step that we generally uncover potential problems that can create bottlenecks or that are show-stoppers. At this point there are still opportunities to implement recommended preemptive actions.

Planning for Success

An enterprise-wide project that intersects over multiple departments, divisions, geographies, and/or business units is vastly more complex than a more traditional project. In fact the critical two stages are at the beginning and the end – planning the project out, and that last stretch toward completion. Dedicate a great deal of attention to properly assessing project plans and time lines, allocating resources, setting skill set requirements, and defining how to collaborate with other teams or partners. Here are some considerations.

ROLES

First identify the essential roles for your project and make everyone aware of their roles. Some roles to consider are shown in Figure 1. There does not need to be a 1:1 correlation between people and roles, nor does every role exist in every stage of the project. Consider mapping out the roles to project stages.

OWNERSHIP

Contrary to traditional divisions of labor and clear-cut accountability among individuals, we strongly recommend joint ownership across individuals and teams. Project components may work well in isolation, but in a middleware-



ABOUT THE AUTHOR

Max King, based in Prolifics' London office, has extensive experience delivering WebSphere solutions for clients worldwide, including JP Morgan Chase, NYSE, Wal-Mart, Lufthansa, BNP Paribas, MetLife, Honda, and Hertz. As a senior member of the Prolifics WebSphere Consulting Division, he specializes in J2EE architecture and best practices, application development and deployment, production "crit-sit" support, production readiness assessments, application migration, and messaging and integration.

E-MAIL

mking
@prolifics.com

intensive environment often it is the component interaction that is the true source of problems. Resolve ownership boundaries early on; consider identifying a liaison, such as the middleware expert or J2EE architect, to make sure the groups work together to jointly own a problem.

Also remember that enterprise-wide solutions require cross-departmental enterprise-level communications. You will need experts to understand experts, recognizing each other's value as well as the value of consensus, teamwork, and common purpose.

SKILLS AND STAFF

You will have to come up with the right composition for your team, considering any predetermined staffing guidelines that are handed to you.

- Make sure to review all of your options against budget and other constraints.
- Ask yourself if you can rev up your existing staff and get them up to speed in time.
- Determine if hiring new talent will be required and beneficial in the long run.
- Consider whether using external professional services in an advisory level or implementation level can enable you to draw on previous expertise and gain a portfolio of lessons learned.

Once you have the right team in place, ensure that everyone is at the right skill level. For instance, you cannot assume someone with traditional database or Unix administration skills can adapt to J2EE or WebSphere administration without some level of training or private mentoring.

PROJECT UNDERSTANDING

Develop an understanding of all the stakeholder needs, both at a high level (such as service-level agreements and user requirements), and also at a detailed technical level (such as ports, corporate guidelines, and compliance with industry standards).

COLLABORATION

Understand existing boundaries and the need to introduce new processes to support a different type of collaboration between the teams. Having remote sites for development and operations provides better infrastructure support but also introduces additional complexity.

Make sure you know how and when you can get hold of people in your team. Time zone differences have an interesting effect on workflow for projects with an international team. There are smaller windows of opportunity for meetings, or alternatively smaller windows of opportunity to work normal hours. It has become a common phenomenon, so it is worth mapping out who is where when (see Figure 2).

Plan not only for when but how you can establish meetings. Determine whether having a physical presence has any extra benefit, such as providing closer communications, to meeting your requirements. Consider whether the use of teamroom collaboration software and Web conferencing technology can satisfy your needs.

ENVIRONMENT

You will need to set up different environments for different functional purposes.

- Your developer workstations will need to allow for com-

ponent unit testing as well as development.

- It is strongly recommended to create a sandbox environment – a protected and limited area where developers can “play” without causing damage. Development teams can test deployment to the server environment, validate build and deployment scripts, and test integration of different components within the solution as well as with external systems.
- Your QA department will require a testing environment that allows for system testing, regression testing, acceptance testing, and integration testing.
- We highly recommend a staging area for load testing, tuning, and preproduction validation. This is a scaled down replica of a production environment. It is the exact configuration of a production environment, but with reduced capacity (e.g., two application servers in staging versus four in production).
- Finally, there is the actual production environment.

Project Design

The J2EE standard encompasses a wide range of components for proven enterprise Web architectures. WebSphere supports J2EE and has the ability to extend applications into different directions with legacy connectors and other enterprise services. You're on the right track using J2EE and WebSphere.

DESIGN FOR SUCCESS CRITERIA

Every project has criteria for success, sometimes agreed upon in service-level agreements (SLAs). Know your transaction, load, scalability, availability, security, and maintainability requirements. You will need to design to meet these criteria and test that the criteria are met.

ADOPT BEST PRACTICES

Best practices exist for Java coding, J2EE component development, and WebSphere architecture and development. To ensure that your application has growth potential and is designed for an optimal deployment, you should make sure that your team refers to these best practices and design patterns and adopts them where relevant. IBM, for example, has published these guidelines in their Developer Domain Web site and their IBM Redbooks. Other recommended best practice books are *Bitter EJB*, by Bruce Tate; *Core J2EE Patterns: Best Practices and Design Strategies, Second Edition*, by Deepak Alur; and *Design Patterns: Elements of Reusable Object-Oriented Software*, by Erich Gamma, Richard Helm, Ralph Johnson, and John Vlissides.

Additionally make sure that your development team is aware of deprecated API functions and API cross-compatibility issues. Refer them to the product documentation and release notes.

BE REALISTIC ABOUT SCOPE

Each company has its own internal standards – its enterprise architecture – that has history and complexity behind it, and that in some cases is even a market differentiator. Now that you are deploying on WebSphere, it is necessary to adapt the company's existing framework to WebSphere. This is a nontrivial, often very custom effort, and should be scoped and designed very carefully.

Other areas to scope carefully are integration aspects,



ABOUT THE AUTHOR

Yateen Dhareshwar is a senior member of Prolifics' specialized WebSphere Consulting Division – a team that IBM calls on to service IBM customers, including MetLife, Morgan Stanley, and Dupont. Yateen's core expertise spans J2EE development and WebSphere administration, and he is highly specialized in deployment, configuration, maintenance and high-availability strategies.

E-MAIL

ychar
@prolifics.com

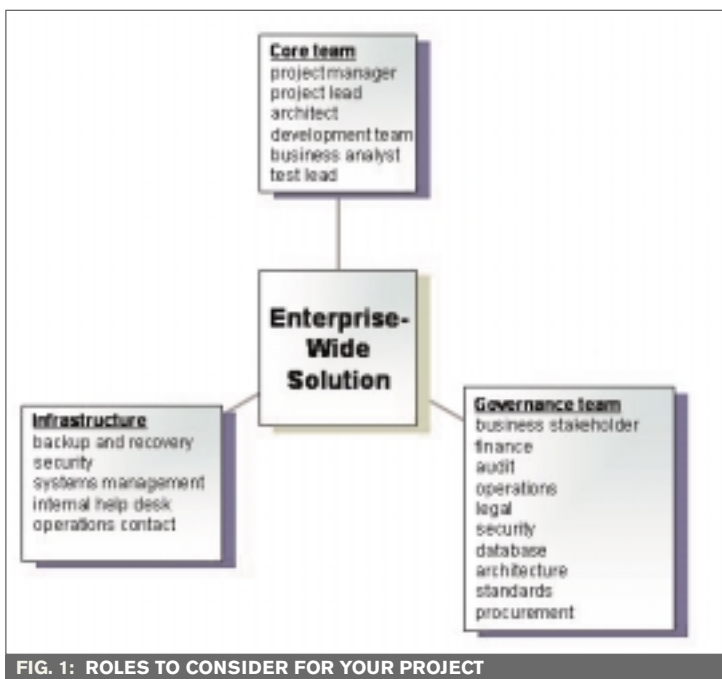


FIG. 1: ROLES TO CONSIDER FOR YOUR PROJECT

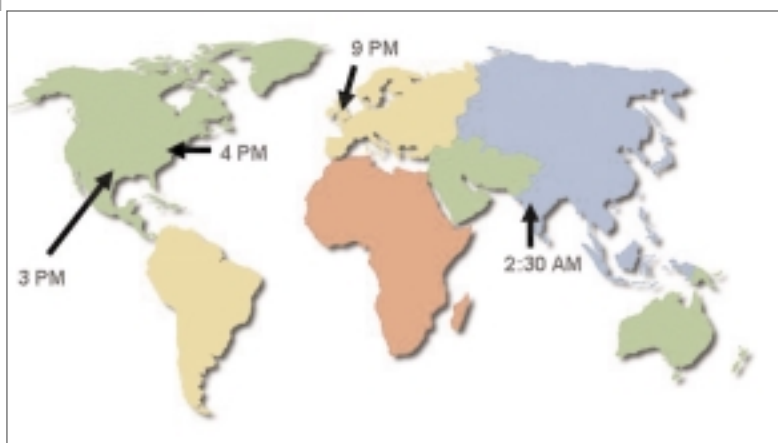


FIG. 2: MAPPING OUT TIME ZONES FOR GEOGRAPHICALLY DISPERSED PERSONNEL CAN BE HELPFUL

which are always underestimated. In fact they are usually worthy of a separate project themselves. In general, any factors external to the immediate team are vulnerable to all sorts of changes.

Project Methodology

Most projects follow a sort of time line. All of them have to go through the cycle of stages shown in Figure 3.

The time line is affected by two major dates, the initial project deadline and the actual release date (i.e., drop-dead date). These are significant dates because they are visible to people outside the core team and become constraints under which you need to bring together multiple factors at the same time. Production readiness is based on projecting where you are today against those dates and forming a plan to meet them.

Traditional project methodology sometimes implies a big bang at the end when you hit production. We suggest that it does not need to be this dramatic. A few test runs to production will alleviate the pain when the time comes. Conducting a few mini-projects within the primary proj-

ect, where all results end up in production, will give you that effect (see Figure 4).

The Rational Unified Process (www-3.ibm.com/software/awdtools/rup) supports this flow naturally, defining an iterative cycle of activities categorized into themes: inception, elaboration, construction, and transition. These repeat every few weeks, and it is implied that they make their way to production within each cycle.

Testing

There once was a trader who went to a fish market for lunch in downtown New York. Bringing back a fresh fish to the trading floor, he decided to tease his buddy with a little “fish-in-the-face” routine. The second trader, caught by surprise, hit the fish, sending it flying across the trading floor and onto a keyboard. The sequence of keys pressed brought down the exchange. Perhaps better “use case” scenarios (see Figure 5) in the test plans could have prevented this?

All kidding aside, testing is essential for assessing and preparing for production readiness. Be careful not to view testing as a buffer between code drops and going live that can be trimmed down on both sides by slips in the release date and aggressive rollout plans. The more you test, the better off you will be in the long run.

Do not neglect any of these crucial testing cycles:

- Load testing
- Integration testing with external applications or systems (e.g., security, messaging hub, packaged applications)
- Validation of build and deployment scripts
- System testing
- Integration testing (to outside systems)
- Functional testing
- Regression testing
- Acceptance testing (functional)

We will drill down on two of these key areas.

LOAD TESTING

Observing the behavior of an enterprise application under realistic load is particularly revealing when trying to estimate the readiness for a production environment. This assumes that the test sequences used by the load generation utility are representative of real end-user activity. There are various suites of tools available to generate load, and they vary widely in functionality, flexibility, administrative skill set, and price.

Observing the application under load is made easier with tools that look into the interplay of the J2EE components, and how they interact with back-end components, the network, the file system, etc. Some third-party tools let you instrument the application by adding agents into the code that act like sensors. Running load with instrumented code allows you to observe the application in a top-down approach, drilling down to different parts as it becomes apparent where bottlenecks are coming from.

Another key to load testing is to use a near-production replica. This type of testing is often ignored because it is extremely difficult to obtain buy-in from infrastructure support or to get external partners to link up their systems in a non-production environment. This may involve legacy access, messaging systems, LDAP servers, or external com-

ponents. However, we have found that the alternative – simulating access to these systems – fails 90% of the time.

USER ACCEPTANCE TESTING

The importance of user acceptance testing should not be underestimated. This is also a good place to flush out quirks in the infrastructure prior to production. Again, it is essential to use a near-production replica. It is also a good way to build up the development team's problem-isolation and troubleshooting skills.

Production Factors

When moving into production it is important to establish proper post-production roles, to set up processes for troubleshooting and deployment, and to have recovery strategies. Systems management is not an afterthought. You need to define or adhere to operational standards – which are often not easy to change. Things to consider include:

- Disaster recovery/business continuity
- Backup and restore
- Hardware upgrades
- System software upgrades
- Application software product upgrades
- Business applications upgrades
- Performance monitoring
- Application configuration and tuning
- Logging
- Auditing

We will highlight a couple of these key areas.



FIG. 3: TRADITIONAL APPLICATION DEVELOPMENT LIFE CYCLE

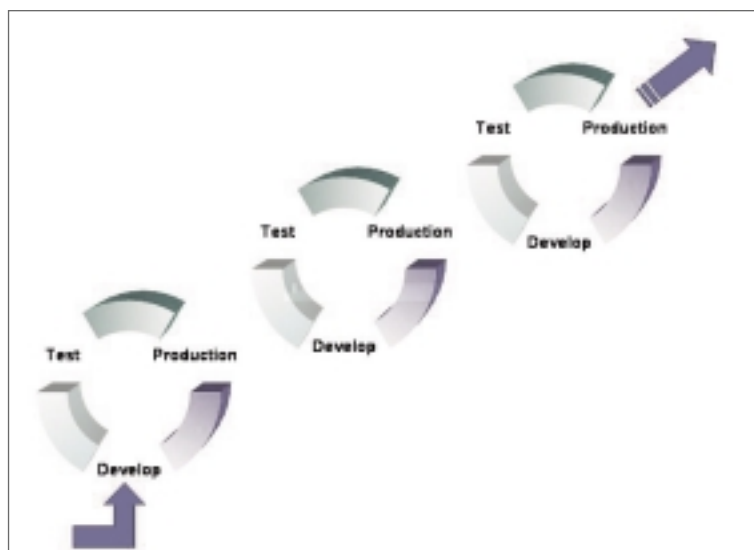


FIG. 4: ITERATIVE DEVELOPMENT LIFE CYCLE

PROBLEM DETERMINATION AND TROUBLESHOOTING

The application logs are usually of little value since many applications have a generic catch-all exception that

is not helpful for the support staff. Therefore troubleshooting a problem often consists of problem reproduction, isolation, some thorough analysis, and resolution. It is often impossible to use a live production site to reproduce problems, as it will affect the user community, so this is another argument for a near-replica release environment. Isolating problems requires a feel for the architecture, and analysis of these problems is most productive if performed by middleware experts. If you cannot reproduce the problem, it will be difficult to validate potential fixes.

MONITORING

There is great value in detecting problems and outages before your user community does, and it may mean you can be proactive in taking corrective measures. A sound monitoring strategy can improve your team's ability to identify and resolve problems.

A monitoring tool should provide the ability to detect a full outage – and also a partial outage involving one critical component. The tool might also be able to report more granular information and measure performance against thresholds (e.g., to determine if the application is violating an SLA requirement such as “no response should take more than two minutes”). There is a trade-off between the level of detail and the overhead required to collect, generate, and store the details.

THROUGHPUT

Throughput spikes can be the result of activity spikes in the daily usage, holiday spending, end-of-month processing, marketing campaigns, etc. – but also can be the result of unforeseen activity. Monitoring can help identify trends and alert personnel to unpredicted spikes.

DEPLOYMENT

How fast do you need to deploy? Generally, under normal circumstances the time to release is not the cause of worry; the main concern is the time it takes to get everything up to a clean state after an outage. Your business sponsors will define the parameters for how an outage will affect the cost of doing business. You need to ensure that you work within these bounds.

DISASTER RECOVERY/BUSINESS CONTINUITY

How will the business operate if the site hosting the solution is no longer functional? A proper disaster recovery plan based upon business impact and a strategy for business continuity, such as reverting to a legacy system until the problems are fixed, will greatly reduce impact to the business in case of a disaster.

AVAILABILITY

As Table 1 shows, an application's requirements for availability impact the design, processes, and cost of ownership.

Redundancy is a good, but expensive, way to guarantee availability. If one server goes down, there will be a running backup in place to capture its requests. If one database server goes down, a backup server will be there to take over. This can be a smooth or a delayed switch, depending on the availability mechanism used.

MAINTENANCE

Defining and knowing in advance when the produc-

tion system can be changed greatly increases your ability to plan. If it is once a night, consider performing troubleshooting at night. If it is only two hours a week, this is harder to do. Not having a maintenance window over an extended period requires a highly customized solution.

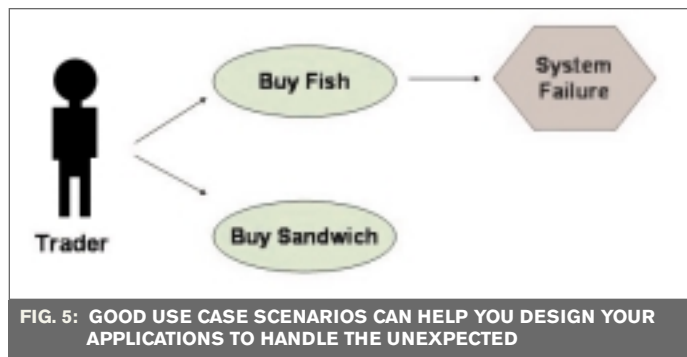
ROLLBACK STRATEGY

Due to the complexity of a new release, it is inherently vulnerable to problems. If the release process fails or exceeds the maintenance window, and the old system is no longer available, you are now left without a production system. Business may demand a rollback strategy, but realize in advance that it requires complex planning and implementation. You will need to assess whether you can revert to older versions without downtime. It is also important to determine how to manage multiple concurrent, distributed systems and make sure they are all independent of one another. Finally, although it may be obvious, be sure to test your rollback facility!

HARDWARE

While assessing the production readiness of a project, one of the basic questions to answer is whether you have enough hardware. You should take the following factors into consideration:

- **How much daily volume to expect per server?** This value is not the same for all servers. For instance, for database servers the volume is measured in transactions, for Web servers in terms of the number of users, and for J2EE application servers in terms of the number and sizes of JVMs.
- **What is the footprint of your application, in terms of memory and disk space?** Just because you set the maximum heap size of the JVMs to 512MB doesn't mean it will stay there. We have seen JVMs grow to 2GB in size. This is true especially if you don't restart them periodically and the JVM uses a Type 2 database



AVAILABILITY	IMPACT ANALYSIS
60%	This is commonly achieved by using a reliable platform.
80%	To gain this level, follow best practices.
90%+	This will require a custom solution. Invest 50% or more effort into gaining the additional availability.
97%+	The focus of the entire application is to stay up 24x7; all else is secondary.

TABLE 1: BALL-PARK FIGURES REPRESENTATIVE OF THE FOCUS AND EFFORT NEEDED FOR HIGH AVAILABILITY SITES FOR A MID-SIZE USER COMMUNITY

- **Do you need additional “invisible” components, such as firewalls, routers, and LDAP servers?** Some of these could be shared resources, so you need to understand what kind of load is already on these devices.

From a hardware standpoint, everything revolves around the three main components: CPU, disk, and memory. Make sure that you are aware of the expansion options of the hardware you order.

Remember that hardware consumption will be affected by the amount of application traffic, so identify and plan for these requirements. From what we have seen in the industry, if what you have is a truly enterprise-size application, there is no such thing as too much hardware; your

“It is never too early to lay the foundation for successful application deployment”


driver with native libraries. When you size memory it would be a good idea if at least 25% of the total system memory is free to accommodate spikes and avoid swapping.

- **Is this a 24x7 application?** Do you need to plan for high availability and failover? This means different things at various levels of the system. In disk terms it means RAID, in terms of the operating system it would mean a redundant system, in terms of the network it would be multiple NIC cards, in terms of application servers it would mean workload management.
- **Do you need a disaster recovery site?** This could double your hardware cost, but it is important to plan for.

users and applications will grow to consume the hardware. That does not necessarily mean that you have to budget for mainframe-size systems.

Be sure the infrastructure group is aware of your deadlines so they schedule the ordering, installation, and configuration of hardware. Communication is key and timing is everything.

Conclusion

In conclusion, consider this: your team has just spent several months hammering out an enterprise-critical application, and it feels as if you have been on the hot seat forever. You planned for production readiness. You’ve certified for production readiness. You flip the go-live switch, logs start to fill up, users start to log on, Operations pats you on the back – and all systems are go! 

*Manage and tune server clusters
– plus debug your apps*

Cyanea/One

from Cyanea

BY JAY JOHNSON

Sure, you can use WebSphere Studio Application Developer to debug a single WebSphere application, but how do you debug applications running together on a cluster of WebSphere servers? WebSphere Application Server has two low-level APIs that can be used to find out and adjust what goes on in a server: PMI and JMX. In some cases, these can be very useful, but to debug a cluster, you're better off using a tool such as Cyanea/One also known as WebSphere Studio Application Monitor (WSAM) version 2.1.1.

Some monitors show only general performance characteristics for each server in a cluster. That is enough to tell you that stack or heap memory is running low on a server instance, but the real trouble is often in a transaction executing across multiple applications, possibly on multiple server instances.

Cyanea/One displays statistics for each Web server or application server in a cluster, provides administration capabilities to manage the servers, and provides facilities to tune system performance. From a WebSphere developer's point of view, however, Cyanea/One's most important features come under the heading of "problem determination," or debugging, as it is commonly called.

Debugging a Server Cluster

Okay, maybe you've been, as I have, in the midst of system load testing, only to have performance slow

down mysteriously as requests come in – until it falls below the service-level agreement for performance. There are a number of debugging strategies you could use in this situation, but the most direct approach would be to monitor requests and raise a threshold alert. A monitor could then provide in-depth information when the request, or set of requests, causes a dramatic drop in performance. In Cyanea/One you can set different kinds of thresholds to trigger a list of actions.

Cyanea/One supports an OLAP-like approach, allowing you to start with the big picture then search for the request that causes the problem. Cyanea/One's top-down approach manages total availability from the application perspective. The J2EE architecture, including the JVM, makes it possible to capture data from inside the application server. This enables performance manage-

ment, exception handling, and capacity planning.

Starting from the widest view and drilling down to specific running applications, Cyanea/One begins by displaying an enterprise-wide view of J2EE application and server availability. The Application Overview page is a data center dashboard that shows all applications running in assigned server groups, making it possible for production staff to spot problem conditions before production users are impacted.

For example, Figure 1 displays a group of servers running a Quotes application. The readout shows the overall availability, performance, and throughput of the application in five-minute increments over the past hour. In this example the red bars indicate a potential problem condition, as response times have exceeded a specified threshold while throughput levels are steadily decreasing. As such, further investigation into this condition is warranted, which can be accomplished by leveraging Cyanea/One's tool set to peer more deeply into the J2EE environment.

Many of the problems in applications concern memory management. Cyanea/One's Memory Diagnosis capability – composed of Memory Analysis, Heap Analysis, and Leak Detection – makes it possible to locate memory issues without resorting to trial-and-error methods (such as removing and replacing portions of code to localize a problem).

The Memory Analysis function enables developers or operators to generate real-time reports about system memory indicators over time, for example, by plotting heap utilization over hourly intervals. If, for instance, the report shows that heap size is growing, it is possible to correlate this trend relative to other critical metrics, such as garbage collection frequency. As shown in Figure 2, although GC frequency is consistent, heap size is steadily increasing, suggesting that a memory leak condition is active.



ABOUT THE AUTHOR

Jay Johnson is an independent J2EE architect and the product review editor for WebSphere Journal.

E-MAIL

aguy4ejbs@yahoo.com

If increased heap size is determined to be a legitimate factor, the next logical step is to use the Heap Analysis function to view a breakdown of all classes consuming heap space. If specific classes appear to be contributing to the problem, the Memory Leak component shows a side-by-side comparison of Java class memory footprints captured at different times. If the size of a class has grown considerably during an interval, the source of the leak has likely been identified, and should be further investigated.

Expand Information at Trouble Spots

Locating a malfunctioning application is difficult in a server farm. Cyanea/One provides the In-Flight Request Search to achieve this. When a client makes a request for a particular server resource, the resource is often a Web page or a Java application. To search for an in-flight request, enter in the search argument using alphanumeric characters and select a group of servers. All requests that contain the search string and are currently active on any server specified in the group will be displayed. You can also restrict a search to a specific server, or simply list all in-flight transactions in the entire server farm.

Under normal conditions, applications hum along nicely when there are no service requests from users. Failures happen as a result of servicing a specific request, or a series of requests. To track down problems starting with a request, Cyanea/One allows traps to be set on requests. These traps act as breakpoints, similar to those in IDE debuggers. Once a trap is hit, the user will be notified immediately when the system meets the conditions of the trap and the alert that was set. The trap could result in sending an e-mail or an SNMP message, collecting Stack Trace, Component Trace, Method Trace, or Thread Dump information. View the Trap History on the Trap Action History page.

An Application Trap detects metrics in a request, method, or SQL call. The system triggers the trap after exceeding the threshold you set for

the metric. When the system meets the definition of the trap, an alert occurs. For example, you can set a trap to alert you after more than 10 requests that include the words "John Smith". The alert may include sending an e-mail or an SNMP message to track the occurrences.

To get detailed information, you can collect a method Trace, Stack Trace, or Thread Dump. The Trap Action History page provides a record of traps that met the conditions set. You may view the trap history, such as the date and time that the action was taken, trap properties, server name, severity, and the type of action that was taken.

A trap can also be set on application performance, to verify whether or not a specific SLA requirement is being met, for example. A Server Resource Trap measures a variety of Target Types. The system will trigger a trap after exceeding the threshold you set for a particular metric. When the system meets the definition of the trap, an alert occurs. For example, set a trap to send an alert when a server is unavailable twice, and you can choose to receive an e-mail after a server becomes unavailable.

Method-Level Debugging

Cyanea/One features a Problem Determination function that provides the ability to trace method execution in order to find bugs in applications. As part of this function, Cyanea/One displays a nesting summary for events and method invocation. From the summary page you can drill down to see the execution details of any method listed.

The Method Trace shows the path of execution for a request. The data displays how the request arrived at the current point in its execution. Additionally, the trace shows looping behavior. When you see excessive, repeated entry and exit records for the same method, it could indicate that the method is caught in a loop. In addition, Cyanea/One shows the cumulative CPU time and the cumulative elapsed (wall clock) time for each trace record. In other words, Cyanea/One calculates the resources consumed by the CPU and the

elapsed time while going from entry to exit, entry to entry, exit to exit, or exit to entry by subtracting the CPU time of the previous trace record from that of the current one. Using this information, you can ascertain which methods are taking the longest to execute and/or are using large amounts of CPU.

You can view the data for a particular trace by setting a search criteria and search value. There are several search criteria for you to choose from, such as Delta Elapsed Time, Delta CPU Time, Elapsed Time, CPU Time, Event Type, and Event Data. Except for the Event Type and Event Data, which are by default sorted in ascending alphabetical order, the other criteria are sorted in descending numerical order.

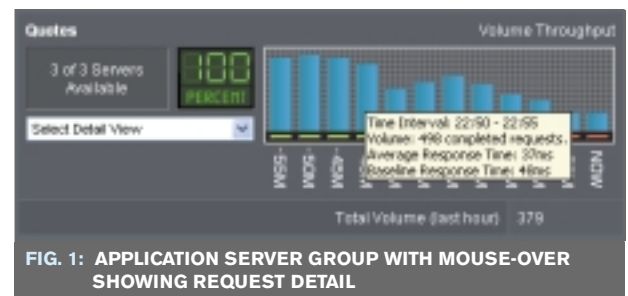


FIG. 1: APPLICATION SERVER GROUP WITH MOUSE-OVER SHOWING REQUEST DETAIL

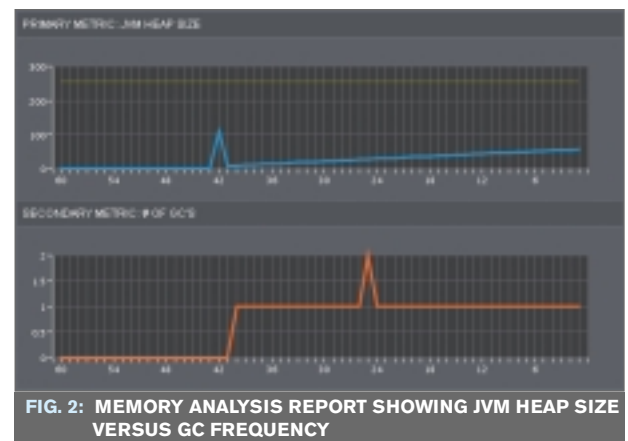


FIG. 2: MEMORY ANALYSIS REPORT SHOWING JVM HEAP SIZE VERSUS GC FREQUENCY

Conclusion

In summary, Cyanea/One provides some much-needed capabilities for managing and tuning a cluster of servers, monitoring server availability, and tuning performance.

Even more needed is the ability to debug applications as they execute on servers – and Cyanea/One provides this.

COMPANY INFO

Cyanea Systems Corp.
2001 Broadway
Third Floor
Oakland, CA 94612
Main (510) 587-7000
Phone: 877.CYANEA8
Fax: 510.587.7002
Web: www.cyanea.com
E-mail: sales@cyanea.com

PRICING

For pricing information, please contact sales@cyanea.com or call 510.587.7007.

Managing HTTP sessions within clustered WebSphere 5 environments

Building High-Availability Web Applications

BY CHRIS DELGADO

To provide the best performance and availability for WebSphere applications, administrators and developers count on scalability features found in the software, hardware, and networking components that host their WebSphere domain. More than ever, the availability of our Web applications can impact critical business processes.



ABOUT THE AUTHOR

Chris Delgado is a consultant based in Atlanta, Georgia, working with clients in the areas of WebSphere, Java, and relational databases. Chris is an IBM-certified specialist in WebSphere and DB2 and has worked as a developer and architect for nearly 10 years.

E-MAIL

christopherdelgado@yahoo.com

Recognizing this, WebSphere technologists are implementing high-availability (HA) clusters using WebSphere Application Server Network Deployment Edition v5, hereafter referred to as ND. Along with HA functionality found in databases, Web servers, and operating systems, ND can be implemented to deliver the benefits of HA architecture: increased performance, failover, recovery, and workload management.

Let's now consider the Java applications hosted within WebSphere. Are they affected when the application servers that contain them are cloned? In the development life cycle, did you consider that your code might be deployed into a clustered WebSphere topology? For the servlet and JSP developer, this means that requests to servlets or JSPs might not be serviced by the same application server, even though the requests might all be for the same session.

In this article, I will examine HTTP session management strategies in clustered WebSphere environments from the perspectives of the administrator and the developer. I will make

the case for implementing ND's framework for session management as an alternative to client-tier sessions, describing the tracking and persistence functionality that ND delivers.

Client-Tier Session Management

The HTTP session is implicitly stateless; this limitation must be considered when designing an HA solution. One way developers can overcome this limit is by storing the session at the client (browser) and delivering it as part of each server request.

There are several ways developers can store the session data at the client, including URL rewriting, cookies, and hidden HTML form elements. These are not to be confused with ND's session tracking mechanisms. The difference is that ND uses URL rewrites or cookies only to track the session ID. Storing the session in the client means sending all or most of the session data back and forth between the client and the server clone. Here are the major problems found with client-stored session data:

- **Network traffic:** In each request to the server, a potentially large amount of session data will be along for the ride. Depending on the implementation, the server might also respond with the entire session. Another drawback is that usually most of the data is unchanged and is unnecessarily exchanged.
- **Varying browser support:** When using URL rewriting or cookies to transport an entire session, you might run into size limitations at the client. There aren't standards that specify how long a URL can be or how much data can be stored in a cookie. Each browser enforces different rules.
- **Security:** HTML forms can be saved anywhere the browser is rendered. Also, cookie files could be edited and URLs could be altered and pasted into another browser. While it's possible to add encryption, this will mean added complexity and performance overhead.

Since the entire session is part of each HTTP conversation, there is no need for stateful interaction with server clones. In this scenario, it doesn't matter which server clone services the request because the session data is contained in the request.

ND offers the alternative of server-managed sessions through affinity, tracking, and persistence mechanisms. This lets the developer take advantage of server-tier session management, with the added benefits of the HA infrastructure built into ND clusters. The rest of this article describes ND's support and functionality for server-tier session management.

ND Session Affinity

The conversation between a browser and a server clone can contain many exchanges. Since the browser initiates the conversation, the server clone that first services the

request must be linked in some way to this new session so that subsequent requests can also be serviced by the same clone. Take, for example, the online shopping cart. As the user shops, items are added to the virtual shopping cart before checkout. This interaction could involve multiple HTTP exchanges. If items in the shopping cart are kept at the server clone, every one of those HTTP exchanges must be serviced by the same clone.

Uniquely identifying this browser-to-server relationship is what's known as session affinity. ND supports session affinity through the plug-in architecture, which defines the transport parameters between the Web server and the server clones. The plug-in parameters are built by the ND administrative server and stored in the plugin-cfg.xml file. When a server cluster is created, ND assigns each clone a unique identifier. Listing 1 shows what it looks like in the plugin-cfg.xml.

Since clone IDs are automatically assigned when the cluster is created, session affinity is enabled by default. The process of matching a session to a clone can impact performance, so if you don't need session affinity, disable it by removing the clone IDs from the plugin-cfg.xml. Also note that the plug-in will bypass session affinity if it doesn't detect session ID as part of the request. I'll discuss later how ND identifies the session and the options we have to track it.

The mechanics of session affinity are pretty straightforward, but it's worth noting that affinity guarantees only that requests for a single session are routed to the same clone. If the clone fails or is shut down, the plug-in will route the request to a different clone. If the alternate clone can't recover the session, then it is lost.

Using our shopping cart example, such a failure will result in an empty cart. To avoid this, ND has session persistence options that make the session recoverable in failover scenarios. But first let's cover session tracking functionality in ND to complete the affinity picture.

ND Session Tracking Mechanisms

In addition to unique clone IDs, establishing affinity requires unique session IDs and an ability to track them in a multipart, browser-to-clone conversation. ND offers three options for tracking the session ID: SSL ID tracking, cookies, and URL rewriting. You can enable more than one tracking mechanism, and ND will resolve them in the following order: (1) SSL session IDs, (2) cookies, and (3) URL rewriting. A good reason to enable more than one mechanism is that each has deficiencies. If you decide to track sessions by SSL ID and the Web server fails, ND can fall back on an alternative tracking mechanism. The great part is that this functionality is built into ND, ready to go. Figure 1 shows ND's session tracking mechanisms.

An advantage of SSL tracking is that it doesn't require any code changes. The SSL ID is negotiated between the browser and the Web server, so even an application server failure won't change the session ID. Keep in mind that SSL IDs are affected by the Web server configuration (httpd.conf for Apache-style Web servers) and this may overlap with ND's session management controls. For example, session timeout is set by the SSLV3TIMEOUT parameter in the Web server, and is also set in ND's session management settings. Another possible conflict might occur if your code contains custom session invalidation routines. Be careful to verify each layer (Web server, app server, Web apps) to ensure one doesn't step on another.

As a tracking mechanism, cookies can be problematic. Many users block cookies out of concern for privacy. Some network administrators filter out cookies at the firewall. Still, beyond any privacy concerns, there are valid technical reasons for avoiding cookies. If cookies are permitted they can easily be manipulated at the client through browser controls or at the file system. Cookie files can be purged, expired, and changed manually. Any of these actions will cause loss of the session ID. If we don't trust cookies to hold session data, then we

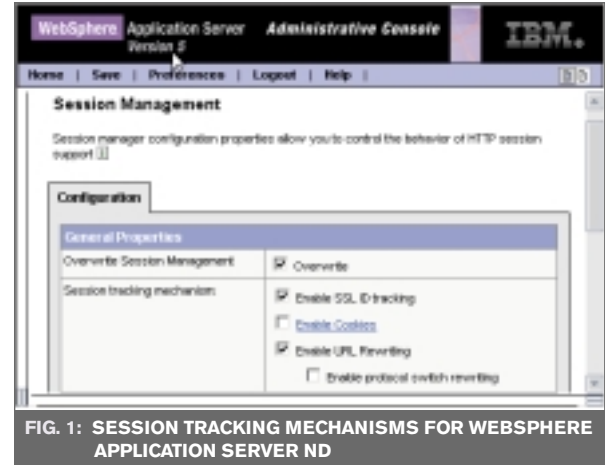


FIG. 1: SESSION TRACKING MECHANISMS FOR WEBSHERE APPLICATION SERVER ND

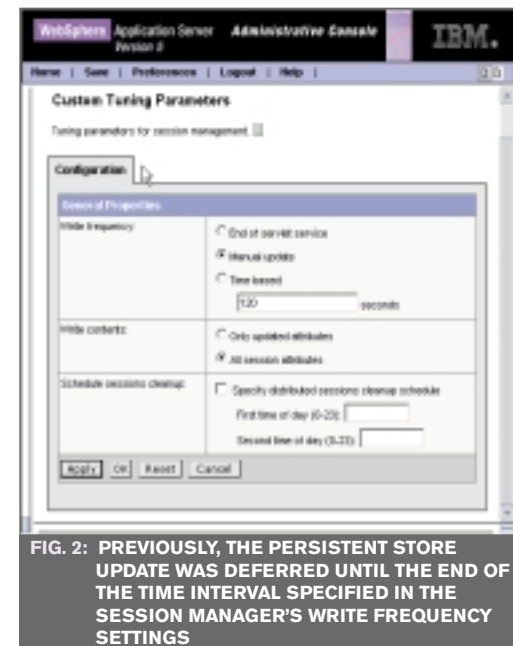


FIG. 2: PREVIOUSLY, THE PERSISTENT STORE UPDATE WAS DEFERRED UNTIL THE END OF THE TIME INTERVAL SPECIFIED IN THE SESSION MANAGER'S WRITE FREQUENCY SETTINGS

shouldn't trust cookies to hold the reference (session ID) to the session data either.

URL rewriting is fairly straightforward and involves appending the session ID to the end of a URL. There is a simple code requirement that the servlet or JSP must encode the URL for ND to properly recognize that a URL has a session attached to it.

The following code snippet shows the URL encoding in the servlet:

```
out.print(response.encodeURL("/  
storefront/chiisai"));
```

The following code snippet shows the URL encoding in the JSP:

```
<%= response.encodeURL(request.get
RequestURI())%>
```

A big advantage for URL rewriting is that it works everywhere. It also supports concurrent sessions for single users, but this could be troublesome if the user pastes the URL of an active session into a new browser instance. The server wouldn't know which browser instance was active. From a code perspective, especially



FIG. 3: THE DISTRIBUTED ENVIRONMENT SETTINGS IN ND'S ADMINISTRATIVE CONSOLE

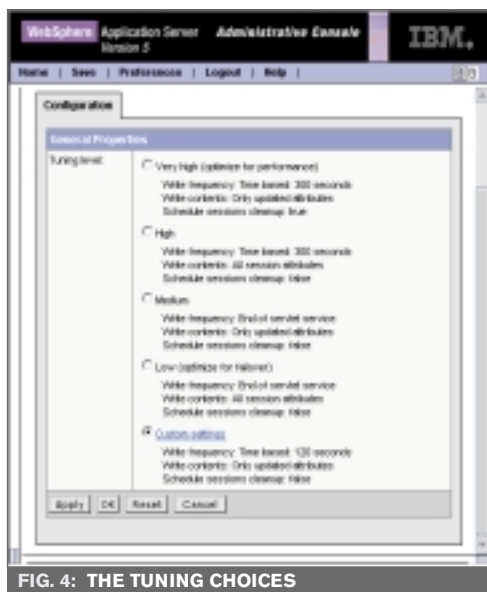


FIG. 4: THE TUNING CHOICES

for JSPs, URL rewriting can be awkward. Any page links in the JSP that need stateful session information would need their URLs rewritten. Also off-limits are any static HTML pages, because there's no way to insert the session ID.

The IBMSession Interface

It is quite common for Web applications to store session data at the server in a session object without persistence. This nonpersisted, local session exists only in JVM memory; if it is not persisted through ND, it is not made available outside this JVM and doesn't play a role in the HA infrastructure. Be careful not to confuse in-memory local sessions with ND's memory-to-memory replication. In-memory local sessions are limited to a single server (per the Servlet 2.3 specification), and don't offer recovery options after failover.

Servlet developers using local sessions store information through the HttpSession interface. The HttpSession is obtained through the HttpServletRequest, and is handled in the servlet's doGet() and doPost() methods. The HttpSession methods to access and update the session include:

```
HttpSession hsession =
req.getSession();
Object o =
hsession.getAttribute("sessAttri
buteName");
hsession.setAttribute("sessAttri
buteName", sessObject);
```

The IBMSession interface extends the HttpSession to add the following functionality: session overflow control, user identity association, and the ability for the developer to force a session to be persisted externally. While the first two fall under the umbrella of ND's session management offerings, it is the session persistence method that is important in a clustered topology. The developer has complete control of when the session gets persisted by calling the sync() method. This level of control is new in ND; previously the persistent store update was deferred until the end of the time interval specified in the session manager's write frequency settings, as shown in Figure 2.

Developers who store session data at the server can easily implement the IBMSession interface, and won't have much work left to achieve a server-tier session management system that supports failover. An added bonus is

that very little, if any, code needs to change to implement persistent sessions with ND.

The WebSphere implementation, through the IBMSession interface, includes methods to access the extended functionality:

```
IBMSession isess =
(IBMSession)req.getSession();
if (isess.isOverflow()) throw new
ServletException ...
String userName =
isess.getUserName();
isess.setAttribute("sessAttri
buteName", sessObject);
isess.sync(); // persist external-
ly
```

Now we can turn our attention toward what ND offers us for session persistence. We've already seen how the IBMSession interface allows the developer to choose exactly when the persistent store is updated, so let's look at the other way that persistent stores get updated, along with the details about ND's persistence mechanisms.

ND Session Persistence Mechanisms

To briefly review, we've identified the session and the server (IDs), associated the two (affinity), and routed the request (plug-in). As we move on and discuss ND's session persistence functionality, keep in mind that ultimately it is the two persistence options that make failover recovery possible for the session.

Figure 3 shows the distributed environment settings in ND's administrative console. Here you can specify either database or memory-to-memory replication. Memory-to-memory replication is new in ND and uses WebSphere internal messaging.

Before the release of ND, database persistence was the only option available. The concept of persisting session objects to a database is fairly straightforward. Usually, when discussing this option, the first question asked is about performance. The general concern is that writing the objects out to a database is slow to begin with, and will collapse under any volume. While it's possible to

PERFORMANCE TIPS

- By default, JSP pages create a session for every page. To avoid creating unnecessary sessions on JSP pages that don't need a session, set the scope to "false" with: `<%@page session="false" %>`.
- If session affinity is not required, disable it by removing the clone IDs from the plugin-cfg.xml file to improve performance.
- If using database persistence, use a dedicated session database. Storing session data in a shared database can adversely affect performance.
- Build a small-scale replica of the HA topology and conduct functionality and performance validation testing.
- Keep session objects small. Large session objects can limit options for session persistence topologies and are resource intensive. Limit data stored in the session to a minimum.

overload even a very highly tuned system with enough volume, the real problem is the size of the session. When selecting database persistence, you must carefully consider page sizes (if using DB2) and whether to use a multirow schema. A single-row schema means one session object per row. A multirow schema means one session attribute per row. The trade-off is between performance and space in the database. The consensus best practice is to keep the session objects as small as possible and carefully analyze the average session size to determine database structure.

Memory-to-memory replication is a new alternative for persistence released in ND. One of the best reasons to consider this option is that it doesn't require a database. At first look, it might not be a big deal to create and manage a simple database for session persistence; however, the database needs to be part of a clustered solution to avoid a single point of failure.

There are three possible topologies: single replica, dedicated replication server, and peer-to-peer (default). The basic idea is that the session objects are "replicated" in memory between server clones, and the different topologies offer varying degrees of recovery. If a clone fails, the next request is routed through the plug-in, which then, depending on

topology, can access a replicated copy of the session. The session size is even more important in memory-to-memory replication than in database persistence. A large session object combined with a high number of users can exhaust JVM heap memory quickly. The performance trade-off comes in the number of servers that participate in replication (based on the three available topologies).

Immediately after a session is persisted, it starts growing stale. Any session updates that happen after the last write won't be recovered in a failover scenario. We must decide between the performance trade-off of low update intervals and an acceptable level of session "freshness." Frequent updates (session synchronization) can hurt server performance. The best strategy is to find the right balance that still keeps your data fresh. In addition to the manual control through the `IBMSession` interface, you can choose a prepackaged tuning level that doesn't require any code change. The tuning choices are shown in Figure 4.

Since performance is an overriding factor in deciding on a persistence method, load testing is important and can help identify any weakness in an HA design before it is implemented. For many developers and administrators it is difficult to stage the entire HA implementation for testing and proof-of-concept purposes. There are some relatively inexpensive options, however, that can help you build a replica of your HA design. The ND environment described in this article was built using VMware's virtual machine software (www.vmware.com), allowing a single workstation to host software components (Web server, database, WebSphere) in separate virtual machines. To test performance and functionality, Apache's JMeter (<http://jakarta.apache.org/jmeter/index.html>) was used to simulate HTTP requests.


Developers can build use cases and test them against a planned HA configuration. For example, if you plan on implementing memory-to-memory replication using the default peer-to-peer topology, you might be concerned about the replicated sessions exhausting JVM memory. Such

virtualized testing can help validate your choice of persistence methods. Developers can also simulate failover by shutting down server clones or even an entire virtual machine to observe recovery behavior.

Conclusion

WebSphere Application Server ND supplies rich session management support in the areas of session tracking and persistence. Since the hardware, network, and software components on the server side of the topology are usually very reliable, it makes good sense to trust application-critical data, like session data, to the server tier. While this article focuses on the HA aspects of ND session management, it should be noted that there are other session-related controls in WebSphere 5 that we did not cover (check the referenced documents for more information). Developers and administrators are encouraged to take advantage of this functionality to build Web applications that meet the demand for application availability.

References

- Wang, H. and Bransford, M. *Server Clusters for High Availability in WebSphere Application Server Network Deployment Edition 5.0*: <http://ibm.com/support/docview.wss?uid=swg27002473>
- *IBM WebSphere Application Server v5.0 System Management and Configuration (IBM Redbook)*: <http://publib-b.boulder.ibm.com/Redbooks.nsf/RedbookAbstracts/sG246195.html?Open> 

LISTING 1

```
<ServerClusterName
Name="t40p_HAclstr">
  <Server CloneID="uvcggu3u"
    LoadBalanceWeight="2"
    Name="eStoresClstr1">
    <Transport Hostname="cdelgado-
t40p" Port="9080"
    Protocol="http"/>
  </Server>
  <Server CloneID="uvcg7kfr"
    LoadBalanceWeight="3"
    Name="eStoresClstr2">
    <Transport Hostname="cdelgado-
t40p" Port="9081"
    Protocol="http"/>
  </Server>
.....
```


Portal Shmortal

2004: It's time to get serious

BY STEPHEN AREES

The “P” word surely gets thrown around a lot these days. It's time to determine what's real and what's not – and for the sake of this article, let's keep our focus on enterprise, or corporate, portals.

Make no mistake – portals are changing the way people work. Beyond mere buzzwords, portal software technology has advanced to the point where companies in all industries are benefiting from the deployment of enterprise portals (by definition, enterprise portals present a single gateway to all corporate information and services so that employees may do their jobs better).

Unfortunately, many of the recent adopters of state-of-the-art portal software (such as open standards-based WebSphere), have something equivalent to “shelfware” right now. Whether it was the smiling salespeople, the fancy Powerpoint presentations, or the slick advertising, 2003 was the year that many companies spent lots of money on portal software. Yet many customers made these purchases with dreams of out-of-the-box or “quick” deployments dancing in their heads, only to find that a successful enterprise portal deployment is, as Donald Rumsfeld would say, “a long, hard slog.”

A successfully deployed enterprise portal will, in fact, truly pay for itself many times over. Thinking back to what the first version of corporate intranets did for employee productivity, when just getting all that data and those hard-to-find documents onto the Web was a victory, it is easy to see what getting truly disparate systems, some new, some ancient, into one nice, slick screen will do for the productivity and satisfaction of your employees, partners, and customers.

If your company hasn't rolled out an enterprise portal, and you are trying to imagine what it would be like, think about this real-world example: www.sprintpcs.com – an external, B2C portal. For years I found Sprint to have the worst customer service ever (of course I was locked into my cell number). And at first, this site was nothing more than an informational portal, offering stale information at that. Now I can tap into my bill, pay my invoice, change my address, check on how many minutes I have gone over on my monthly plan, etc. – all by using my own voice mail pin number for sign-on! So Sprint has a much happier customer, and they cut down on their call center costs.




A successful approach to building an enterprise portal requires understanding the following factors that determine whether portals either succeed or fail:

- **Political:** Ensure top management backing (establish an advisory group) and make it easy for departments to join the portal early on.
- **Organizational:** Get contributors to comply with the portal rules, enter decent metadata, and refrain from fielding “maverick” intranet servers beyond the intended scope.
- **Technical:** Use a true open-standards, comprehensive portal application that will allow for growth and scalability; avoid specific “portal” interfaces (i.e., ERP portals) and proprietary systems.

My own company, Rave Software Solutions, has been somewhat guilty this year in failing to help our customers successfully move their portal deployments into the business units. Of course, we have been very busy helping their IT departments architect, configure, install, and troubleshoot just their portal infrastructures.

Now we are focusing on delivering value to the business units. After much market research we have selected the sales force as one group of employees that could truly benefit from an enterprise portal. Our Sales Portal solution now helps sales reps access their disparate order entry, marketing, CRM, collaboration, training, commission and expense tracking, travel, and other applications in one screen. Look for these targeted industry or function-specific portals to be popping up from the business partner community this year.

Remember, portals are smart. Corporate portals deliver the right information and services to the right individual users, thereby increasing employee productivity, which, after all, is what drives a strong portal ROI. And if we are to keep employee productivity on the rise as in the third quarter of this year, we have our work cut out for us. My suggestion: start building a portal now.

When you really get down to it, as one IBMer told me, WebSphere Portal is really just fancy plumbing, and it is up to us to decide which faucets to turn on. So, here's to 2004 being the year we all go out and build that fancy information “fixture.” 

ABOUT THE AUTHOR... Stephen Arees is director of Business Development and Marketing for Rave Software Solutions, a division of Computer Generated Solutions (CGS), which provides a range of offerings in the e-business, application development, e-learning, and vendor management spaces. Rave hosts the IBM WebSphere Innovation Center in New York City.

E-MAIL
sarees@thinkrave.com