SYS-CON

VISIBILITY • PERFORMANCE • MAINTENANCE

WebSphere® JOURNAL

www.SYS-CON.com/WebSphere

The World's Leading Independent WebSphere Resource

SEPTEMBER 2005 VOLUME 4 ISSUE 9

Maintaining Visibility and Control in WebSphere 6.0

Key changes in release 6.0 from a management perspective

BY MARINA **GIL-SANTAMARIA** & RUDVIK **DOSHI** PAGE 6

Presorted
Standard
US Postage
PAID
St. Croix Press

NEW!

CodePro AnalytiX™

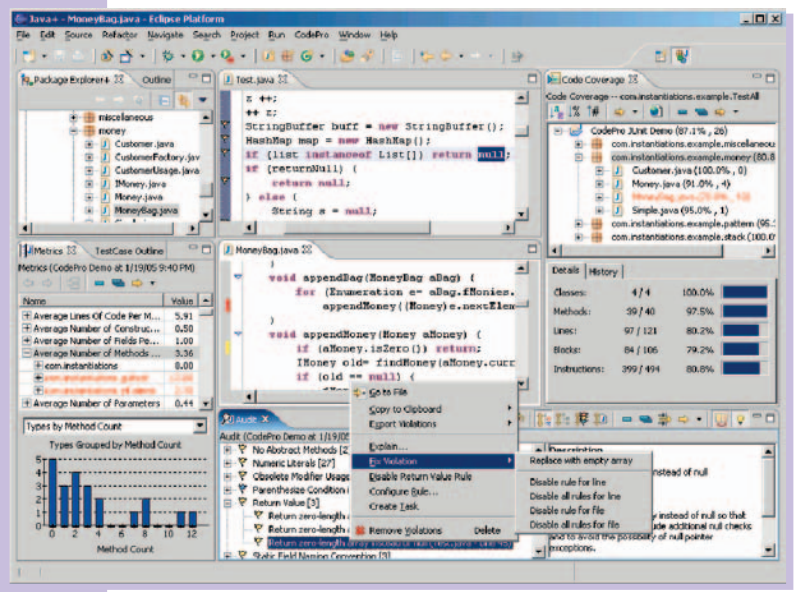
Continuous Collaborative Code Analysis for Rational®, WebSphere® & Eclipse

Imagine your software team creating higher quality, more reliable, Java systems... with less time and money.

Sound like a fantasy?

...*CodePro AnalytiX* lets you do exactly that, or

Your money back.



***AnalytiX* automates the process of building quality into your software as it's developed. Using the latest software development technology *AnalytiX* monitors Java code development in real-time, as its written. It identifies and automatically fixes hundreds of issues that would reduce quality and reliability.**

Powerful wizards generate unit and regression tests and analyze code coverage, helping drive out problems early in the development process.

Less experienced developers code like experts, experts create tight, maintainable systems.

***AnalytiX* also allows you to define, distribute, and enforce corporate coding standards and quality measures across your development teams...no matter where they're located.**

Early identification and correction of code quality issues dramatically improves your application development process and reduces the time and money it takes to develop your high performance Java systems.

From the development tool experts at Instantiations. No one outside of IBM has longer or deeper experience in creating Eclipse-based Java development tools.

Put our experience to work for your team.

www.instantiations.com

1-800-808-3737

Key Product Capabilities

- **Detect and correct code quality issues ...*Automatically***
- **Define, distribute and enforce code quality standards across development teams**
- **700+ rules and metrics/300+ Quick Fixes**
- **Code metrics with drilldown and triggers**
- **Custom rules and rule sets**
- **JUnit test case generation**
- **Code coverage analysis**
- **Dependency analysis and reporting**
- **Integrated team collaboration**
- **Javadoc analysis and repair**
- **Seamless integration with IBM Rational Developer, WebSphere Studio and Eclipse**

Risk Free Trial Copy

www.instantiations.com/analytix

eclipse READY · IBM Business Partner · eclipse FOUNDATION MEMBER

instantiations

Movin' On Up

time to migrate to the next generation of WebSphere

Ask the Experts

Prolifics, a premier-level IBM Business Partner and 1 of 3 WebSphere Service Providers retained by IBM, specializes in WebSphere migrations.

Having conducted numerous migrations, both from earlier versions of WebSphere and from other J2EE application servers, such as BEA WebLogic, Oracle AS or Sun iPlanet, Prolifics has migration tools and a methodology that avoids unnecessary risk, time, effort and cost.

From J2EE App Servers to WebSphere

Prolifics conducts Migrations from other J2EE Application Server environments such as BEA WebLogic, Oracle AS, Sun iPlanet, etc.

From WebSphere v3.5/ v4 to v5

Prolifics migrates from earlier versions of WebSphere to IBM's latest release.

Taking You Step by Step

Prolifics offers MIGRATION ASSESSMENTS, PROOF-OF-CONCEPTS, and WEBPSHERE MENTORING to help you validate, plan, and execute a successful transition.

Prolifics

Winner of 2003 IBM BUSINESS PARTNER LEADERSHIP AWARD

116 John Street
New York, New York 10038
www.prolifics.com
crm@prolifics.com
800.675.5419

Prolifics® IBM Business Partner

Big Blue Rejuvenated

BY ROGER STRUKHOFF

IBM announced an expanded SOA strategy in September building on the recent momentum it has gained over its archrival in this space, BEA, as well as Microsoft's .NET strategy.

IBM may face its most difficult challenge from the collection of Open Source technologies that is now referred to as LAMP (Linux, Apache, MySQL, and Python/Perl/php), but the company seems to be adopting a savvy and aggressive Linux strategy of its own as a way to reach out to the Open Source community and, more importantly, reach those customers who are demanding Linux in their enterprise IT environments.

Steve Mills, senior vice-president and group executive of the IBM Software Group, led the splashy announcement. He said that the new software and services "build on IBM's existing SOA capabilities. Specifically, new and enhanced software from IBM's WebSphere portfolio is designed for business process management."

Mills said the new strategy entails "greater levels of integration" among WebSphere and IBM's Rational and Tivoli product offerings. He also said that IBM will reach customers "of all sizes" with its new strategy and held forth on the overall vision, saying that "SOA implies much more than simple Web Services connections or a yet-to-be delivered set of business components that might work together. It is about realizing the benefits of business process management today with existing application assets. The new software and services we are announcing today build on IBM's leadership – we have the most comprehensive set of capabilities for business process management and integration in the marketplace today."

IBM's Robert LeBlanc, general manager of WebSphere, joined Mills for questions from what sounded like a semi-skeptical press regarding IBM's embrace, or lack thereof, of the Java Enterprise Service Bus (ESB). His answers said, in essence, that IBM will be going its own way down this road. (You can learn more about his remarks in the article about this announcement in this issue of *WebSphere Journal*.)

One would guess that there are technical and marketing reasons for this approach, with IBM seemingly having gained enough market share and clout to bend the rules of SOA development to its will in the future.

Meanwhile, on the Open Source front, IBM maintains that it is the Linux market leader. In an exclusive interview with *WebSphere Journal*, the man who leads IBM's worldwide Linux strategy, Adam Jollans, explains his view of what the company is doing.

Big Blue's *bête noire* Microsoft is reaching its 30th birthday and as OS/2, the technically superior operating system that Microsoft defeated is being put to pasture, IBM is rejuvenating its aggression in the enterprise IT marketplace. Its SOA and Linux strategies are part of a larger picture of a company simultaneously reaching out to keep up with its customers, lead with its software technologies, and leverage its enormous numbers advantage in service to chart its path as the pre-eminent IT provider in the world.

Roger Strukhoff, editor-in-chief of ***WebSphere Journal***, is West Coast Bureau Chief for the SYS-CON News Desk, and President of www.wdva.com. He spent 15 years with Miller Freeman Publications and The International Data Group (IDG), then co-founded CoverOne Media, a custom publishing agency that he sold in 2004. His work has won awards from the American Business Media, Western Press Association, Illinois Press Association, and the Magazine Publishers' Association. Read his blog at http://www.rssblog.linuxworld.com. Contact him at roger@sys-con.com.

WebSphere® JOURNAL

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 ROGER STRUKHOFF
CONTRIBUTING EDITOR PATTI MARTIN
PRODUCT REVIEW EDITOR JAY JOHNSON
EDITOR NANCY VALENTINE
ASSOCIATE EDITOR SETA PAPAZIAN

WRITERS IN THIS ISSUE
Rudvik Doshi, Rainer Dzierzon, Marina Gil-Santamaria, Greg Herringer, Michael Menze, Eric Naiburg, Klaus Nossek, Scott Simmons, Roger Strukhoff, Linfeng Yu

SUBSCRIPTIONS
For subscriptions and requests for bulk orders, please send your letters to Subscription Department.
888-303-5282 201-802-3012
SUBSCRIBE@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
SENIOR VP, SALES & MARKETING CARMEN GONZALEZ
VP, INFORMATION SYSTEMS ROBERT DIAMOND
VP, SALES & MARKETING MILES SILVERMAN
PRODUCTION CONSULTANT JIM MORGAN
FINANCIAL ANALYST JOAN LAROSE
CREDITS & ACCOUNTS RECEIVABLE GAIL NAPLES
ACCOUNTS PAYABLE BETTY WHITE
ADVERTISING DIRECTOR ROBYN FORMA
ADVERTISING MARKETING DIRECTOR DENNIS LEAVEY
ADVERTSING SALES MANAGER MEGAN MUSSA
ADVERTISING SALES MANAGERS KERRY MEALIA
LEAD DESIGNER ANDREA BODEN
ART DIRECTOR ALEX BOTERO
ASSOCIATE ART DIRECTORS ABRAHAM ADDO, TAMI BEATTY, LOUIS F. CUFFARI
VIDEO PRODUCTION FRANK MORICCO
WEB DESIGNERS STEPHEN KILMURRAY, VINCENT SANTAITI, SHAWN SLANEY
ONLINE EDITOR ROGER STRUKHOFF
CIRCULATION SERVICE COORDINATORS EDNA EARLE RUSSELL, LINDA LIPTON
CUSTOMER RELATIONS MANAGER BRUNI STAROPOLI

EDITORIAL OFFICES
SYS-CON Publications, Inc.
135 Chestnut Ridge Road, Montvale, NJ 07645
Telephone: 201 802-3000 Fax: 201 782-9637
SUBSCRIBE@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

NEWSSTAND DISTRIBUTION CONSULTANT:
GREGORY ASSOCIATES, W.R.D.S.
732 607-9941, BJGASSOCIATES@CS.COM

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

Why is it that some Java guys are more relaxed than others?

These days, with everything from customer service to sales and distribution running on Java, keeping your enterprise applications available can be pretty stressful.

Unless of course, you've discovered the power of Wily. No other software offers our level of insight. Which means you'll be able to monitor transactions and collaborate with colleagues in real-time. Even more important, you'll be able to optimize performance across the entire application—end-to-end.

So put Wily to work. And keep performance problems from pushing you over the edge.

Get Wily.™

1 888 GET WILY | wilytech.com

wily technology

Maintaining Visibility and Control in WebSphere 6.0

Key changes in release 6.0 from a management perspective

BY MARINA GIL-SANTAMARIA & RUDVIK DOSHI

Marina Gil-Santamaria is a Product Manager for Unicenter Application Infrastructure Management solutions, including Web Servers, Application Servers and Message Oriented Middleware. She has been with CA since 1997. During the last eight years, Marina has held positions such as Senior Systems Engineer and Specialist across different departments at CA. Marina holds a M.S. in Electrical Engineering from the Universidad Politecnica de Madrid, Spain. marinagilsantamaria@ca.com

The new features and capabilities found in the WebSphere Application Server and WebSphere MQ 6.0 lets organizations combine new and existing heterogeneous IT assets as part of composite applications that reside on an underlying service-oriented architecture (SOA) across distributed and mainframe environments. As you deploy your composite applications, having a complete strategy in place to manage and secure them properly becomes the next goal. This article explains the key changes in version 6.0 from a management perspective and offers pointers on how to meet the management and security challenges specific to the WebSphere 6.0 platform.

What's New in WebSphere 6.0?

The IBM product line known as the WebSphere platform contains more than 350 different products, but in this article we'll focus on the two major elements of the 6.0 release: WebSphere Application Server and WebSphere MQ.

IBM hasn't had a new release of WebSphere MQ for more than two years (although the company has supplied nine support packs during that time), so 6.0 is a major product release, designed to provide a foundation for companies building out composite applications and Services Oriented Architectures. In case you don't know, a composite application consists of functionality drawn from various sources like individual Web Services, selected functions from within other applications, or entire legacy systems whose output has been packaged as Web Services. IBM released WebSphere Application Server (WAS) 6.0 earlier this year, continuing their trend of releasing a new release every year. The WAS 6.0 release isn't as big as the 5.0 release, but it's still a solid release with many new features geared towards improving administration and application deployment, support for new J2EE specifications and APIs, better security, and support for Web Services and SOA.

Let's begin with what's new in WebSphere MQ. The 6.0 release leverages WebSphere MQ's reliable transport layer to send and receive Simple Object Access Protocol (SOAP) messages out-of-the-box that are frequently used in establishing Web Services links between applications. So WebSphere MQ 6.0 can easily provide reliable messaging and connectivity to enterprises looking into converting their existing Enterprise Resource Planning (ERP), Customer Relationship Management (CRM), or legacy applications into Web Services. At the same time, WebSphere MQ 6.0 ships with an interface to the Eclipse Open Source programmer's Workbench, so many Eclipse-based development and testing tools can be used to develop composite apps that make use of WebSphere MQ capabilities to communicate and pass messages across a company's IT infrastructure.

In addition, WebSphere MQ 6.0 includes a large library of connectors to application infrastructure elements – like DB2 or WebSphere or WebLogic application servers – and Oracle, SAP, or Siebel applications and support is provided for both Java and .NET environments. In fact, this release is tightly integrated with IBM's CICS Transaction Server and WebSphere Application Server 6.0 on both distributed and z/OS environments. IBM has also increased the amount and length of the messages that WebSphere MQ 6.0 supports, which again is geared towards giving users more flexibility and scalability, to allow using WebSphere MQ in the enterprise.

Similarly, WAS 6.0 introduces a lot of new standards

and programming models to support Web services and SOA. Every EJB that's deployed on the application server automatically becomes a Web Service, thus integration of business logic with other application components can be done directly without any intermediate steps. Likewise, this release is also leveraging new Java standards, which automatically add a layer of authentication and authorization to secure Web services. As a part of the SOA initiative, WebSphere has included a new Service Data Object (SDO) API. That means that developers can now access data from various sources like databases, custom connectors, and XML, using this single SDO API.

New Management Features in WebSphere 6.0

Now that we've described some of the highlights of the WebSphere MQ and WAS 6.0 releases, let's go over some of the new features that can help you manage more effectively in more detail.

WAS 6.0 introduces fine-grained control over the various application components, which means you can now update only one part of the code without bringing down the entire application. This release also lets you create multiple profiles of the application server in a single installation. One big advantage is that administrators can now install multiple cloned instances of the application server on the same box, and changes to the application or the server configuration can be tested on the duplicate profile first without bringing down the production server. Once changes have been verified, settings can be migrated to the production server with minimal downtime. Likewise, WebSphere MQ 6.0 for z/OS can now be tuned much more dynamically to respond to changes in application workloads, so system resources such as buffers and pagesets can be added and removed without having to restart the queue manager or channel initiator.

Additionally, WebSphere 6.0 also implements all the new administrative APIs and standards from J2EE such as JMX 1.2 – the industry standard used to expose management functionality – Java Management Specification or JSR-77 – the industry standard schema for describing J2EE managed objects – and J2EE Deployment or JSR-88 –an open standard to deploy applications. Therefore, these new standards will make it easy for third-party management tools to interoperate with the application server in a non-vendor-specific manner. IBM's Performance Monitoring Infrastructure (PMI) now uses the standard JMX APIs to expose performance data. PMI has also been enhanced to let administrators enable or disable individual statistics. The old PMI API, though available in this release, is now deprecated and shouldn't be used. In addition, WebSphere 6.0 has been significantly improved by incorporating the new JSR 47 standard for centralized logging to facilitate troubleshooting deployed applications. Because of this, both application developers and troubleshooters

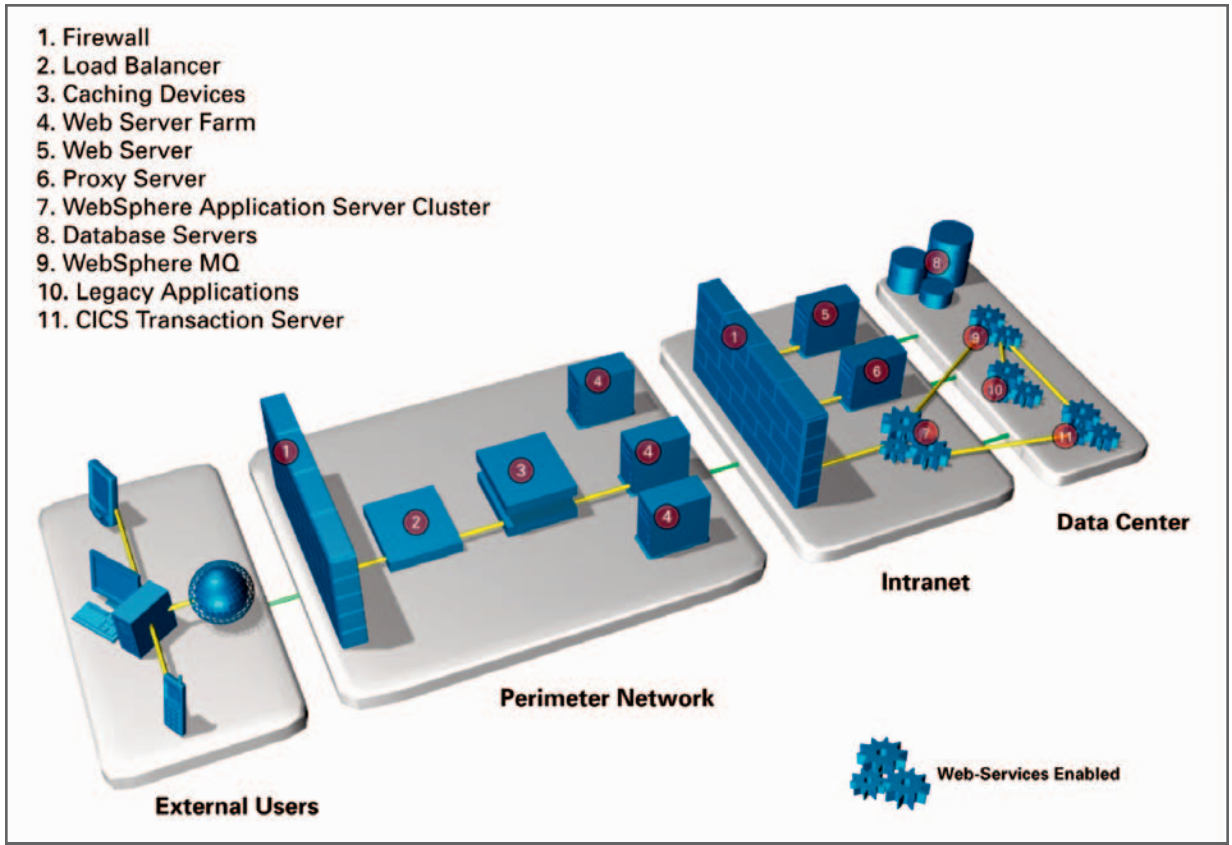


FIG 1: EXAMPLE OF A WEBSPHERE DEPLOYMENT

Rutvik Doshi is a Development Manager for Application Servers and J2EE technology within CA's Unicenter Application Infrastructure Management solution portfolio. He is also involved in application management and Web Services standards groups at DMTF. Rutvik holds a B.S from Indian Institute of Technology.
rutvik.doshi@ca.com

can now hook into the centralized logging framework and look for message patterns. Another nice feature is the log file analyzer that can be used to analyze data from multiple servers. Similarly, there are some helpful new management functions in WebSphere MQ 6.0 to enable actions such as checking which applications are using the queue manager and tracking the objects they've opened, or showing the applications with long-running transactions. These new functions can really facilitate root cause analysis and troubleshooting tasks. Additionally, new attributed have been added to existing MQ commands like DISPLAY QSTATUS and DISPLAY CHSTATUS to provide more visibility into MQ resources. And there's a interesting new report option that can be put in the Message Descriptor so applications that process a message can generate reports on what they have done with that message. Generically, IBM calls them Activity Reports and they include information such as which transmission queues are used, and which channels are transporting the message.

How To Manage WebSphere 6.0 Environments

When thinking about your strategy to manage WebSphere MQ and WAS infrastructures there are three different approaches: product consoles, scripts, and management vendors.

- ***Product Consoles:*** WebSphere MQ v6.0 ships with a WebSphere MQ Explorer, which enables users to administer and configure WebSphere MQ resources and configure a WebSphere MQ network remotely from Intel-based Windows and Linux systems. Similarly, WAS 6.0 includes a Web-based console that can be used to manage all the WAS instances in a network deployment. What are the pros of this approach? Well, they have a visual interface so they're more user-friendly than scripts, and once you pay for the application server or WebSphere MQ software they're free, which is a plus nowadays. And what about the cons? Well, these administrative consoles are built to support a "one at a time" manual process, not a "one to many" automated one, so they don't scale up too well, and they differ across product versions. Besides, these embedded consoles are more geared towards administration and application deployment than day-to-day monitoring of servers. Another factor to consider is their lack of integration. So what if you're on the application support team? How will you know if the application problem or performance degradation is related to a WebSphere MQ link on the application server side or an MQ channel that has stopped? Lack of integration can really mean additional time spent on troubleshooting the root cause of an application problem and finger pointing across various departments.
- ***Scripts:*** WebSphere MQ MQSC commands and the WAS wsadmin scripting client provide powerful functionality to control WebSphere MQ and WAS in an environment from configuration to performance management to administration. The pros? Well, they already come with IBM's WebSphere and WebSphere MQ software, and if you're a mainframe or a Unix guru you probably already feel comfortable with this approach. However, this process is manual, tedious, and time consuming, and certainly prone to errors. You also need physical access across all systems and platforms, and a lot of in-depth knowledge to use these tools effectively.
- ***Management vendors:*** Management vendors, either point solutions players or enterprise management vendors, offer solutions geared toward monitoring and maintaining J2EE environments such as WebSphere. What are the cons? Well, you have to pay extra, and depending on the management vendor you select, this amount could be relatively high. What are the benefits? Their solutions are usually architected based on different agents or observers overseeing few or all elements of the WebSphere infrastructure – Web Servers, applications servers, WebSphere MQ, databases, CICS systems, messaging servers, network devices, etc., and report back to a consolidated manager. If your vendor of choice provides full visibility across all elements of the WebSphere stack down to the network layer, this approach is powerful, because it provides end-to-end application-centric discovery and visualization, so you can manage all the different resources that are contributing to the overall application performance as a unit. This is Computer Associates' philosophy where it maps different solutions to each element of the WebSphere platform. For example, CA offers Unicenter solutions for Web servers, WebSphere, WebSphere MQ, CICS, Web Services, databases, or networks and systems to monitor the underlying WebSphere infrastructure from end-to-end and automatically identify measure, track, and report on application service levels across the WebSphere stack.

As you can see in Figure 1, WebSphere MQ 6.0 and WAS 6.0 are part of a broad SOA that has to be understood in the context of a larger implementation. What does this means from a management perspective? Well, there are multiple points of failure since application problems can happen in your application's source code, in a WebSphere MQ system, in a method's call to a back-end database, or in the network itself. Because of this, using scripts or independent product consoles may not allow adequate understanding as to how the entire infrastructure supports your applications or business processes at all levels. Instead, using a management vendor that's going to give you a lot of depth into each infrastructure silo – down to the network layer – combined with visibility across the end-to-end WebSphere infrastructure is the right way to go. And there are other benefits when selecting this option. Usually management tools are designed to be used across different roles and departments to facilitate

cross-departmental communication – from development, to administrators, to operations, to application support teams – so you get powerful functionality without having to have the expertise or administrate access required to build scripts or the knowledge to understand topological dependencies and how each application has been deployed in your environment. And this is critical when considering that business processes in the WebSphere context are particularly dynamic in nature, as application cycles have shortened considerably, and applications are moved into production faster than ever before.

Management vendors usually spend time researching best practices in terms of what resources you need to manage and provide out-of-the-box recommendations on what should be acceptable performance metrics based on custom base-lining analysis. This can really help in terms of filtering out what parameters and metrics you really need to oversee from a wealth of information. These solutions usually provide corrective actions that are executables, scripts, or programs that can be automatically executed to fix problems without IT intervention. These tools usually have intuitive UIs so they're user-friendly and look-and-feel the same across WebSphere MQ and WAS versions and platforms so you don't have to spend a lot of time with manuals or learning the management tool itself.

Similarly, management vendors usually provide both passive and active management and this provides a lot of value. Passive or reactive management is what a product console offers in the sense that you oversee performance metrics to detect application problems or threshold breaches in real-time. Active or preventive management lets you periodically exercise and verify the behavior of your applications via virtual user interactions. These synthetic monitoring capacities provide a complete picture of the health and behavior of your applications, detecting and correcting application problems before real end users run into exception errors or invalid return data. This preventive management approach is only available from management vendors.

Summary

Managing WebSphere 6.0 applications and underlying infrastructures is a complex, difficult, time-consuming job. We've talked about what's new in WebSphere 6.0, focusing more on those features that have the most impact from a management perspective and described how WebSphere 6.0 supports Web Services and Service Oriented Architectures better. In this context, we've commented on some of the pros and cons of the three main management approaches: product consoles, scripts, and management vendors, highlighting the criticality of managing WebSphere 6.0 environments in the end-to-end enterprise.

Take a look at a WAS for z/OS messaging approach

WAS for z/OS and MQ

BY: LINFENG YU

As a J2EE application server, WAS for z/OS supports the Java Messaging Service (JMS). There are three choices for JMS providers in WAS for z/OS V5.x: WebSphere MQ (WMQ), Embedded Messaging (EM) JMS provider, or a third-party product.

Linfeng Yu is a software architect with ISO, Inc. He has extensive experiences in developing large-scale, complex enterprise-wide architectures and cross platform software development. He has been working with WebSphere for both distributed platform and z/OS since version 3. He can be reached at lyu@iso.com.

The EM JMS provider is a simplified WMQ. It provides an isolated messaging environment in WAS for z/OS that can't be used to interoperate with any other messaging systems. Obviously the JMS support in WAS for z/OS was a temporary solution. Now it's been deprecated in WAS for z/OS V6. Instead a new JMS provider named Service Integration Bus (SIB) has been introduced. SIB is part of IBM's Enterprise Services Bus (ESB) infrastructure solution, which isn't just a simple JMS provider. I'll discuss WAS for z/OS and SIB in a separate article.

There are few third-party messaging products that run on z/OS. So this article will focus on discussing WMQ JMS provider connectivity architecture and Java messaging development. If you'd like use a third-party product as the JMS provider, all the messaging resource definitions in WAS for z/OS are indirections to the provider's resource. You should refer to the third-party product's document for details.

If not mentioned specifically, this article refers to WAS for z/OS V5 (or later) and WMQ for z/OS V5.3 (or later).

To make the article self-contained, I'll start with some zSeries terminology even though most appeared in my previous articles. If you're familiar with them, you can directly jump to the WAS for z/OS WMQ Connectivity Architectures section.

z/OS Concepts and Terminology

zSeries hardware can allocate resources to multiple logical partitions (*LPARs*) in a single zSeries machine, with each partition supporting a single z/OS image.

A *SysPlex* is a collection of LPARs joined together to form a single logical entity or view to an external observer.

A Coupling Facility (*CF*) is a zSeries machine with microcode that permits high-speed communication between LPARs in a SysPlex as well as a common repository for sharing data by subsystems like DB2 and WMQ that are in different LPARs in the SysPlex.

Resource Recovery Service (*RRS*) is a z/OS component that can do transaction management for multiple subsystems such as CICS, DB2, WebSphere, and WMQ on the same LPAR.

Workload Manager (*zWLM*) uses installation-defined policies and service-level commitments to govern workload performance in the system.

Dynamic Virtual IP Address (*DVIPA*) is a common external IP address for an application residing or executing on different LPARs in the SysPlex.

Sysplex Distributor (*SD*) is a z/OS component (part of the TCP/IP stack) that consults zWLM to distribute inbound DVIPA requests to the best suitable LPAR in the SysPlex.

Automatic Restart Management (*ARM*) is a z/OS component that will try to restart a job or task after a failure.

The *Batch Adapter* is the interface between WAS for z/OS and WMQ on the same z/OS image. It lets applications in WAS for z/OS use the MQI (the de facto WMQ programming interface). A connection thread is provided from an application task control block (*TCB*) to WMQ.

Queue-sharing group (*QSG*) lets a group of queue managers access the same shared queues. Only WebSphere MQ for z/OS has the queue-sharing group feature.

WAS for z/OS WMQ Connectivity Architectures

There are two basic connection modes from WAS for z/OS to WMQ: binding mode and client mode. To exploit the scalability and availability features provided by SysPlex on the zSeries platform, the connectivity architecture with WMQ queue-sharing group has to be considered. Different architecture options are discussed as follows.

Non-QSG Local Connectivity Solution

In Figure 1, WAS for z/OS and WMQ coexist on the same z/OS image. Applications in WAS for z/OS communicate with WMQ through a cross-memory mechanism provided by the Batch Adpater. This trans-

port type for JMS is called a binding mode. The architecture discussed is called Basic Non-QSG Local Connectivity Solution in this article.

Binding mode is the best performance option because of its memory-to-memory communication mechanism. The basic binding mode depicted in Figure 1 doesn't provide failover solutions for the WMQ queue manager. (You wouldn't add more WMQ queue managers because that would cause application code changes.) ARM can be used to decrease the WMQ queue manager's outage time.

System automation tools like Netview can be used to stop the application server whenever the WMQ queue manager fails. So the storm drain risk could be reduced.

Let's take a look at the WAS for z/OS. As long as the control region is up, zWLM will always make sure that there are enough servant regions handling workloads. Should a servant region fail, zWLM will start a new servant region. From this perspective, a single WAS for z/OS server has better availability than distributed platforms.

To scale WMQ, more zSeries resources should be allocated or added.

There are two ways to scale the application server: adding more servant regions to the single server or creating a vertical application server cluster by adding new servers. Creating a vertical server cluster doesn't offer better performance. It consumes even more resources than adding more servant regions to the single server. However, it does provide better availability because the vertical cluster removes the single point of failure (SPOF) – the control region.

The basic binding mode's level of security is good. When a user attempts to access a WMQ resource, SAF security checks the relevant user ID or IDs to see if the access to the resource is allowed. You may still need the authorization control for WMQ resources.

The WAS for z/OS' thread identity function is available in this solution. The user identity can be passed to WMQ and the RMF audit trail records are cut. This security option is available in WAS for z/OS exclusively. Most z/OS customers would like to use this feature to make their applications more accountable.

The Batch Adapter supports single-phase commit. To support two-phase commit protocols, a WAS for z/OS stub, CSQBWSTB, which leverages the RRS facility, should be used.

The architecture can be expanded to achieve better availability and scale out. As shown in Figure 2, the revised architecture is called Advanced Non-QSG Local Connectivity Solution in this article. In this solution architecture, a cross-LPAR horizontal server cluster is configured. Each server in the cluster connects to the corresponding WMQ queue manager in binding mode.

The user requests are distributed to the most suitable application server in the cluster by SD (consulting with zWLM) via DVIPA**.** If one of the application servers fails or one of the LPARs goes down, the requests will be routed to the server running on different LPARs for processing. The best application server availability can be achieved when the LPARs reside on separate physical machines.

If the WMQ queue manager fails, a storm drain is possible (for WMQ outbound workloads). Similar to the basic binding mode solution, Netview can be considered as avoiding the storm drain. For fast recovery, a z/OS facility like ARM should be used.

To failover and balance WMQ inbound workloads, it's recommended that a WMQ queue manager cluster be set up as part of the solution. Should any one of the WMQ queue managers fail in the cluster, the new inbound messages would be routed to other available WMQ

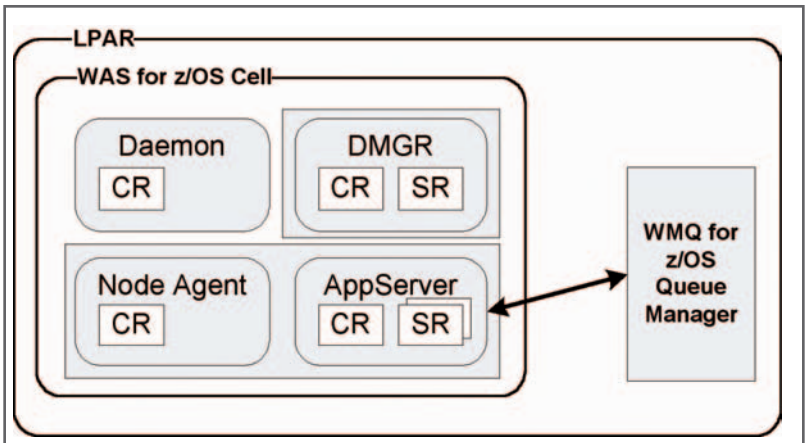


FIG 1: BASIC NON-QSG LOCAL CONNECTIVITY SOLUTION

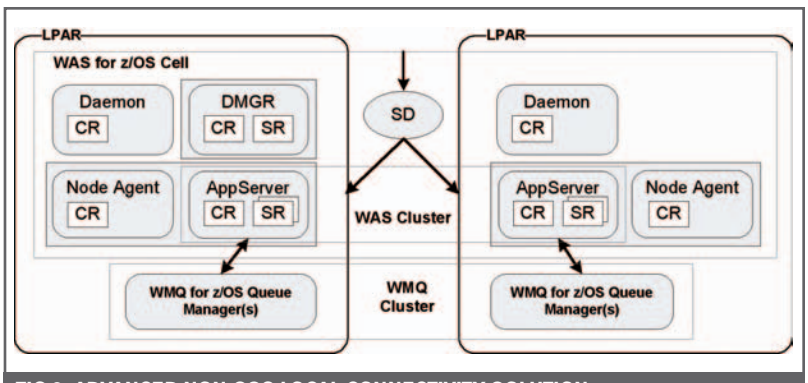


FIG 2: ADVANCED NON-QSG LOCAL CONNECTIVITY SOLUTION

queue managers in the cluster using certain load-balancing algorithms. In the meantime, there's no need for a message producer to know which queue manager a message is sent to. However, the WMQ cluster doesn't solve the message availability problem. The messages sitting in the queue that belongs to the failed WMQ queue manager can't be processed until the failed WMQ queue manager comes back up. This might be a big problem especially for time-critical transactions.

The advanced binding mode has one more scalability choice than the basic binding mode. The system can be scaled out by adding more servers on different LPARs to the horizontal server cluster.

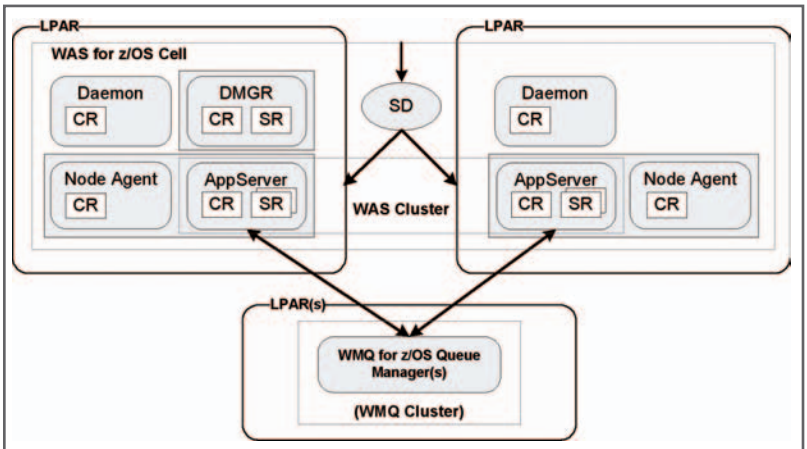


FIG 3: NON-QSG REMOTE CONNECTIVITY SOLUTION

Non-QSG Remote Connectivity Solution

If you prefer to separate WAS for z/OS and WMQ, as in Figure 3, and have them run on separate LPARs for software cost or isolation reasons, use WMQ's remote connection through TCP/IP. This transport type for JMS is called client mode. The architecture here is called Non-QSG Remote Connectivity Solution.

The client mode's performance is worse than that of binding mode because the application server and WMQ queue manager communicate through TCP/IP instead of cross-memory. Even the remote connection can be improved by using zSeries facilities to reduce TCP/IP delays; the local connection will always perform better.

The architecture's overall availability is worse than that of the Non-QSG Local Connectivity Solution because there are more potential failure points. Since the application servers and WMQ queue managers are on different LPARs, they can scale independently. The architecture does offer better scalability.

The application severs' availability can be improved by creating a horizontal cluster. User requests are distributed to the most suitable application server in the cluster by SD (consulting with zWLM) via DVIPA. The total number of the servant regions limitds the scalability of the application servers.

Running more WMQ queue managers can improve WMQ's availability and scalability. But a message consumer must connect to a specific WMQ queue manager. Otherwise you have to change the application code for handling multiple WMQ queue manager connections. Nevertheless, a WMQ queue manager cluster should be created to balance the inbound message workloads and avoid message producer's code change when you scale WMQ. However, the WMQ queue manager cluster doesn't solve the message availability problem.

ARM can be used for fast recovery. And you should use system automation tools like Netview to avoid the storm drain (for WMQ outbound workloads).

Furthermore, you can have several WQM queue managers running on separate LPARs. But you should to be aware that more zSeries resources will be recruited. At this time, the architecture might not be a software cost-saver any more.

Since the client-mode connection is established through the WMQ Channel Initiator address space using a special channel, SERVCON, WAS for z/OS' thread identity security is no longer available.

WMQ resource protection is addressed by WMQ itself through the SAF interface. The transport layer security has to be set up in the transport mechanism itself using secure protocols such as SSL or IPSec to protect data flow. WAS for z/OS provides the component or container-managed authentication alias to pass authentication data to WMQ for creating WMQ connections.

The architecture realizes two-phase commit processing using XA protocols with the RRS facility. The semantics of the JMS session object determine if two-phase commit coordination is use. WMQ can either be the participant or the coordinator in the transaction.

One of the architecture variations is shown in Figure 4. It just indicates that the client mode can be used to connect to a local WMQ queue manager too. However, that doesn't mean it's recommended.

QSG Solution

The WMQ message availability problem mentioned in the advance binding mode section can be addressed by using shared queues as shown in Figure 5. Let's call it QSG solution.

The QSG solution can deliver the highest availability (especially message availability). The application servers connect to the WMQ QSG

Motive brings an enlightened approach to application management, building management intelligence into the application itself. Only Motive transcends the fragmented, disjointed reality of traditional application management, to deliver the visibility, insight and automation that support and development teams need to keep applications running smoothly.

This is no mere vision for the future. It's here now in application management products from Motive. Learn more about Motive's breakthrough approach in a new white paper about built-in management at www.motive.com/within1.

motive

www.motive.com

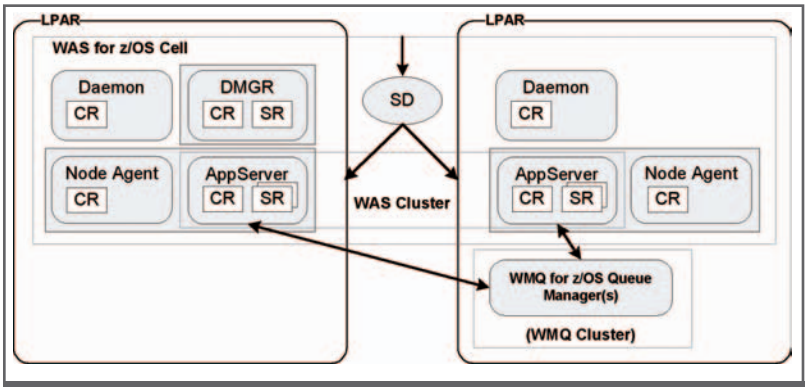


FIG 4: VARIANT NON-QSG REMOTE CONNECTIVITY SOLUTION

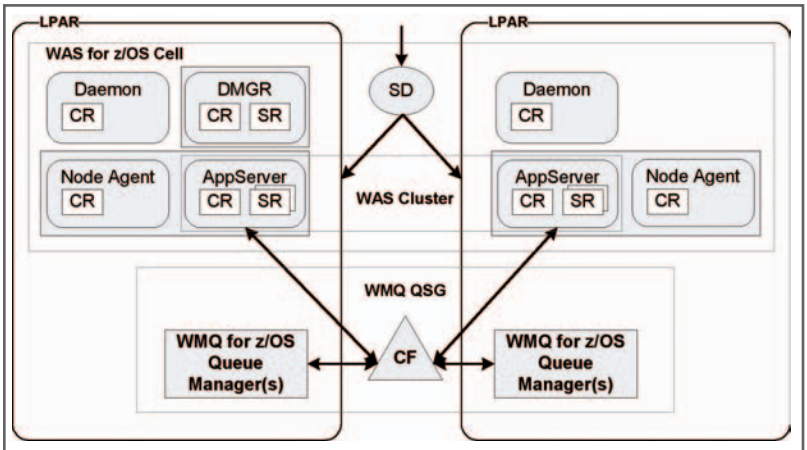


FIG 5: QSG SOLUTION

instead of a specific queue manager. This is known as group attach. Because all the queue managers in the QSG can access the same set of shared queues, the application doesn't depend on the availability of a particular queue manager; any queue manager in the queue-sharing group can service any shared queue.

Each application server can connect to the QSG in either binding mode (memory-to-memory) or client mode (TCP/IP) in the QSG solution. In the binding mode, by defining more than one WMQ queue managers on the same z/OS image to the QSG, you can avoid the storm drain (for outbound WMQ workloads). In the client mode, the storm drain won't exist if the server connection channel is defined with a group disposition. Using binding mode connections is recommended for better performance.

QSG peer recovery further enhances the availability of messages in a queue-sharing group. Should you need more information on queue-sharing, shared inbound/outbound channels, etc., please refer to resource section at the end of this article.

Because messages can survive WMQ queue manager failure in QSG, there's no need to use persistent messages with the shared queues. As a result messaging system performance will be boosted. However, there might be a little extra network traffic because of the additional products involved. Using non-persistence messages might compromise the performance gain. It's worth noting that you have to make sure of the CF's availability when you make the architectural decision to use non-persistent messages.

WAS for z/OS and WMQ can scale both up and out by exploiting the SysPlex technologies. The SD leverages DVIPA and zWLM, routing the client requests to the appropriate server in the WAS for z/OS cluster for load balancing and checking availability.

If application servers connect to the WMQ queue managers in binding mode in this architecture, the security and transaction models are exactly the same as the Non-QSG Local Connectivity Solution. Otherwise they're the same as the Non-QSG Remote Connectivity Solution.

To summarize, the QSG solution is the most sophisticated architecture and offers the highest message availability, better scalability, and good performance. If message availability is critical to your application, you should choose the QSG Solution. The system will be more complicate and, of course, more expensive than those without QSG. If message availability isn't critical to your application, you can choose the Non-QGS Local Connectivity Solution for better performance or the Non-QSG Remote Connectivity Solution for software cost saving and isolation. Note that setting up a QSG environment means every application must use the shared queues. You can still use private queues as though the QSG doesn't exist. The architecture decisions are dictated by your business applications and infrastructure constraints (if any).

WMQ for z/OS Java Application Development Tips

In this section the focus will switch from the connectivity architectures to WMQ Java application development and related configurations. Ten tips will be given along with the discussion.

There are two Java APIs that can be used against WMQ for z/OS: WMQ Java interface and JMS. The WMQ Java interface is implemented as a Java wrapper of MQI. JMS is the Java messaging standard. It defines a generic view of a message-passing service. The generic JMS model is based on the interfaces defined in

Sun's javax.jms package. The WMQ JMS implementation maps the generic JMS model to the MQI programming model. It sits on top of the WMQ Java Interfaces.

Tip 1: Apply the Messaging Patterns to design your WMQ messaging application solution. <<Enterprise Integration Pattern>> is a very good resource for the Messaging Patterns.

Tip 2: Use the JMS model if possible, even if the programs using JMS are slightly slower than the programs using WMQ Java Interface directly. Since JMS is a product-independent standard, your program won't be tied to WMQ.

There are a few things that can only be done in the WMQ Java interface, for example, setting your own message id. In those situations, the WMQ-specific code should be encapsulated in a separate class. If you want to use different messaging products, the code change only needs to be done in one place.

Tip 3: Don't forget to remove the JMS header from the message if the message is going to a non-JMS consumer. You can't use any user-defined JMS properties at this time. In WAS for z/OS, the JMS message header can be removed by setting *JMS Client* to *MQ* in the queue/topic destination configuration in the WAS for z/OS administration console.

Tip 4: Use MQFMT_STRING as the message format for inbound/outbound messages from/to non-JMS message producers/consumers. In JMS term, the message is a text message. Any other message format might cause an interoperability problem. Some Java programmers like using JMS object messages because they're convenient for Java message consumers. I wouldn't recommend it. Besides the interoperability problem with non-JMS message producers/consumers, using the JMS object message tightly couples the message producer and consumer. WMQ is supposed to be used to create a loosely coupled messaging solution without any implementation knowledge between the message producer and consumer.

Tip 5: Use the reply-to-queue and reply-to-queue-manger correctly to achieve the highest message routing flexibility in your application. When the destination queue is created, you can pass both the reply-to-queue-manager and reply-to-queue to the *createQueue* method as uniform resource identifiers (URIs), as shown in Listing 1. This form lets you specify the remote queues (queues on a queue manager other than the one you're connected to). It also lets you set the other properties contained in a com.ibm.mq.jms.MQQueue object. When a client application sends in a message for processing in WAS for z/OS it can control where the response message should be sent by setting the reply-to-queue and reply-to-queue-manager in the request message

Tip 6: The generic JMS header fields/properties that can be set by the Message Object are JMSCorrelationID, JMSReplyTo, JMSType, JMSXGroupID, and JMSXGroupSeq. Other generic JMS header fields/properties are actually set by the Send Method. For example, if you want to send a message with a certain priority, the code in Listing 2 won't work because the priority set by the Message Object is ignored when the message is sent. Instead, the code in Listing 3 should be used. This type of JMS implementation is quite strange. Should you need more information, please refer to the resource section.

Tip 7: Use a persistent message only if it's absolutely necessary. WMQ v6 has introduced a new WMQ queue attribute called *NonPersistentMessageClass*. The value of the attribute determines whether non-persistent messages on the queues are discarded when the queue manager started. This attribute can be exploited to improve JMS persistent message performance. Detailed information can be found in the resources.

Tip 8: JMS connections and sessions are pooled at the Java layer in WAS for z/OS. At a lower level the MQ JMS classes also pool the physical connections to the WMQ queue manager. The JMS-related connection pooling architecture in WAS for z/OS is completely different from that of the JDBC connection factory. The WMQ JMS connection pool doesn't have any association to the connection factory connection or session pool. The connection in the connection factory connection pool is a different entity from the physical connection to WMQ queue manager. In fact, a JMS session corresponds to a physical WMQ queue manager connection. The word "connection" is confusing here. Figure 6 shows the relationship between the JMS connection entities in WAS for z/OS

Many times I've seen developers assume that the JMS connection factory works like the JDBC connection factory. The direct consequence of making that assumption is that the

"WAS for z/OS and WMQ for z/OS are key building blocks in IBM's SOA solution"

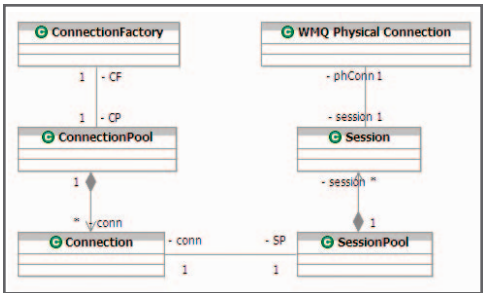


FIG 6: JMS CONNECTION ENTITY RELATIONSHIP

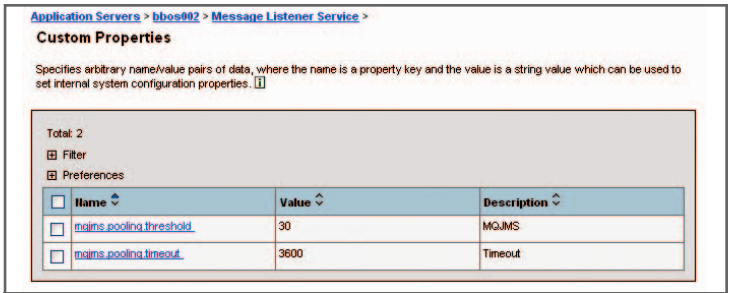


FIG 7: ENABLING WMQ JMS CONNECTION POOLING

connection factory, connection and session pool, and MDB listener ports aren't configured correctly so the JMS applications (including MDB) don't behave correctly. Furthermore, the assumption causes the WMQ JMS connection pooling not to be enabled, which hurts the JMS application performance in WAS for z/OS.

The WMQ JMS connection pooling can be enabled by adding *mqjms.pooling.threshold* and *mqjms.pooling.timeout* as custom properties to the *Message Listener Services* of the application server as shown in Figure 7.

Tip 9: Use the WMQ message Backout Count and maximum retries (an MDB listener port property) to prevent the MDB listener port in WAS for z/OS from shutting down. Should the Backout Count reach its threshold, you can either discard the message or forward it to the dead-letter queue. How to deal with poison messages can be specified through the connection factory configuration in the WAS for z/OS administration console.

Tip 10: Deal with the MDB processing throttle in WAS for z/OS wisely. MDB throttle support is exclusively supported in WAS for z/OS. The reasoning behind the MDB throttle is to control the flow of the resource at hand from one stage to another. Satisfactory message retrieval should be obtained by adjusting the rate at which message references (MRs) are handled and work requests (WRs) are created. Without any kind of control over the number of WRs that are queued in the zWLM queue, an excessive number of WRs accumulate and are less likely to get processed by a minimal number of resources in the Servant Region. Control is based on available resources. Configuring MDB throttle support involves listener port settings, connection factory settings, and MQ property settings. Please refer to the resource section for the configuration details.

I have merely scratched the surface of MDB in WAS for z/OS here. I've frequently been asked how MDB support was implemented in WAS for z/OS. Since the MDB support is getting more complicate in WAS for z/OS v6, I'm planning to discuss it as a separate topic in the future.

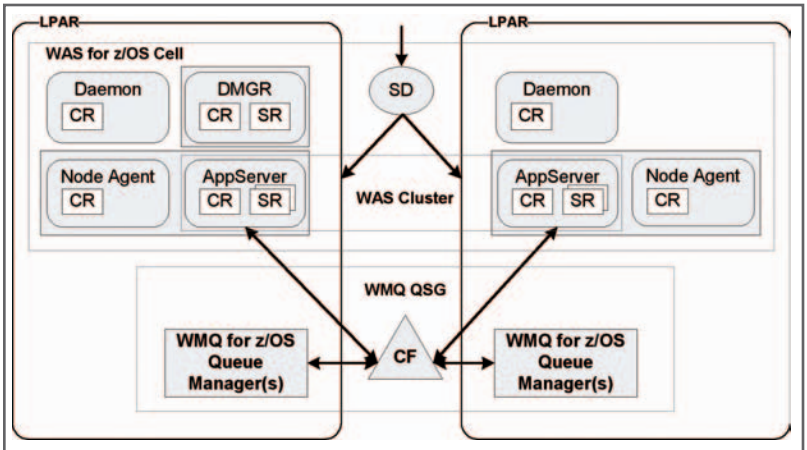


FIG 8: ENABLING WMQ JMS CONNECTION POOLING

Conclusion

This article has explored the WAS for z/OS and WMQ for z/OS connectivity architectures and given 10 tips for JMS application development and related configuration. The WAS for z/OS and WMQ for z/OS is an attractive enterprise messaging solution platform that is a key building block in IBM's SOA solution.

Resources

- WebSphere for z/OS information center http://publib.boulder.ibm.com/infocenter/wasinfo/v6r0/index.jsp?topic=/com.ibm.websphere.zseries.doc/info/welcome_nd.html
  http://publib.boulder.ibm.com/infocenter/wasinfo/v5r1//index.jsp?topic=/com.ibm.websphere.zseries.doc/info/welcome_zos.html
  http://publib.boulder.ibm.com/infocenter/wasinfo/v5r0/index.jsp?topic=/com.ibm.websphere.zseries.doc/info/welcome_zos.html
- WebSphere MQ using Java, IBM Manual http://publibfp.boulder.ibm.com/epubs/pdf/csqzaw13.pdf
- WebSphere MQ for z/OS, z/OS Concepts and Planning Guide http://publibfp.boulder.ibm.com/epubs/pdf/csqsat03.pdf
- Enterprise Integration Pattern http://www.eaipatterns.com

LISTING 1: USING URI TO CREATE A QUEUE

```
Session session;
Queue ioQueue;
boolean transacted = false;
String qMgrName = "DestQMgrName";
String qName = "DestQName";
String qUri = "queue://" + qMgrName + "/" + qName;

try {
   …
   session = connection.createSession(transacted,

      Session.AUTO_ACKNOWLEDGE);
   Queue ioQueue = session.createQueue(qUri);
   ...
} catch (JMSException e) {
   // TODO Auto-generated catch block
   e.printStackTrace();
}
```

LISTING 2: SETTING MESSAGE PRIORITY WITH MESSAGE OBJECT

```
TextMessage outMessage;
int msgPriority = 6;

try {
   …
   outMessage = session.createTextMessage();
   outMessage.setText(outString);
   outMessage.setJMSPriority(msgPriority);
   MessageProducer messageProducer = session.createProducer
(ioDestination);
   messageProducer.send(outMessage);
   …
} catch (JMSException e) {
   // TODO Auto-generated catch block
   e.printStackTrace();
}
```

LISTING 3: SETTING MESSAGE PRIORITY WITH SEND METHOD

```
TextMessage outMessage;
int msgPriority = 6;

try {
   …
   outMessage = session.createTextMessage();
   outMessage.setText(outString);
   //outMessage.setJMSPriority(msgPriority);
   MessageProducer messageProducer = session.createProducer
(ioDestination);
            messageProducer.send(outMessage, DeliveryMode.
NON_PERSISTENT,

                                 msgPriority, 2000);
   …
} catch (JMSException e) {
   // TODO Auto-generated catch block
   e.printStackTrace();
}
```

Part two of a two-part article

OpenSTA

BY: GREG HERRINGER

Greg Herringer is an IT Architect with 15 years experience in customer relationship management and contact center technologies, with a focus on the financial services and public sector industries. His background cuts across the entire application development lifecycle.
gherring@ca.ibm.com

Testing distributed transactional systems involves wrangling a lot of test data and test scripts. IBM's Greg Herringer describes how OpenSTA helped to automate his project's system integration testing activity without breaking the testing budget.

This article describes how the open source tool OpenSTA facilitated system integration testing of a middleware solution based on IBM WebSphere Application Server and the WebSphere Branch Transformation Toolkit (BTT). OpenSTA automation improved test execution accuracy and reproducibility while preserving the project's tight test budget. With the benefit of the lessons learned and key OpenSTA workarounds documented in this article, you may want to consider OpenSTA for your own distributed application testing strategy.

Customizing the Test Script Template

The generic design of the test script template results in only four lines that have to be modified for each specific test script:

- Line 4 contains the test script name per the project's naming convention. This name appears in the test results.
- Lines 15 and 17 point to the IFX-encoded request and model response specific to the test script.
- Line 46 calls the subroutine designed to make specific comparisons of the actual response to the model response based on the type of test being run.

Test Script Subroutines

The OpenSTA Script Control Language supports several structured programming techniques. The test script template exploits OpenSTA's support for user-defined subroutines. By defining a generic test script template that was used for all tests, any logic changes or enhancements made to the test script capabilities could be contained in two files, IFX_SUBROUTINES.INC and IFX_VARIABLES.INC. This eliminates the need to refresh the entire set of test scripts when a scripting enhancement is identified. Instead, these two common files could be deployed to the test clients.

Of the 24 custom subroutines supporting these test scripts, the following three subroutines from the IFX_SUBROUTINES.INC file are representative of OpenSTA's various capabilities:

- **Read_Rq** – this subroutine makes use of the character string variable REQUEST_LINE shown in Listing 1 to read in the IFX-encoded request message at script execution time. The OpenSTA Script Control Language reference suggests using the built-in command OPEN for file handling; however, there is an outstanding bug in OpenSTA, so only the first line of a file can be read. The Read_Rq subroutine provides a custom alternative to using the built-in command.
- **Remove_NewUpDt** – this subroutine makes extensive use of the string manipulation commands available in OpenSTA. Its purpose is to remove those IFX elements from both the model response and actual response that contain timestamps. Since the timestamp elements of the actual response is different for every execution of the test script, they are removed from the comparison of the actual response with the model response. For the initial system integration test cycles, the test team visually inspects the timestamp elements of the actual response to ensure that the value is in the correct format and time range.
- **Generate_RqUID** – this subroutine uses OpenSTA's current date, current time, and random number generation features to generate a unique identifier for each IFX-encoded request sent to the middleware. This unique identifier is subsequently associated with the actual IFX-encoded response per the project's IFX specification.

The IFX_SUBROUTINES.INC and IFX_VARIABLES.INC files are included in the OpenSTA script package available with this article online at http://websphere.sys-con.com/ (*WebSphere Journal* archives Vol: 4, Iss: 8).

There are some limitations imposed by OpenSTA on the design of the subroutines and the use of variables in those subroutines. For

instance, the variables defined are all global. This means that local variables can't be used in the subroutines. Local variable techniques could have reduced the amount of duplicated logic in the subroutines.

Another OpenSTA limitation is that there can be at most 128K of character string data defined by any one script. This proved to be a challenge since some of the IFX-encoded messages can be up to 15K in length. In combination with the variables used to store message fragments for various parsing routines, the scripts come close to breaking this character string data maximum.

Loop control also proved to be a challenge. Specifically, OpenSTA can't specify a "do while" or "do forever" control for loops in which the number of iterations isn't known in advance. The solution is illustrated in Listing 1.

Listing 1 shows that a loop count maximum has been specified (MAXFILESIZE). If the last line of the IFX-encoded message is encountered before the loop count maximum, additional subroutines are called and the loop is ended. If the loop count is exceeded, a message is recorded in the test script log and is considered by the tester to be a failure in the test script. Fortunately, this message wasn't encountered during the formal system integration test cycles.

Executing the Tests

Once the test scripts were prepared, the formal system integration testing could begin. OpenSTA provides a test execution framework that lets testers execute one or more scripts in various combinations and a configurable number of iterations. Figure 1 shows a typical OpenSTA Test designed to run all the tests in project test category 00_4A one at a time with a delay of five seconds between each test execution as indicated in the Start column. Note that since there's only one iteration of each test script, the delay options shown at the bottom of Figure 1 don't apply.

As the tests were executed, OpenSTA tracked the test results in two files: TestRep.txt and TestLog.txt. OpenSTA provided additional test result files, but these were the two used by the test team.

The TestRep.txt file recorded all the messages issued by the REPORT command in the test script, including the REPORT commands that wrote the actual IFX-encoded response. It also recorded the pass/fail flag for the test case. The testers used the TestRep.txt file to verify the

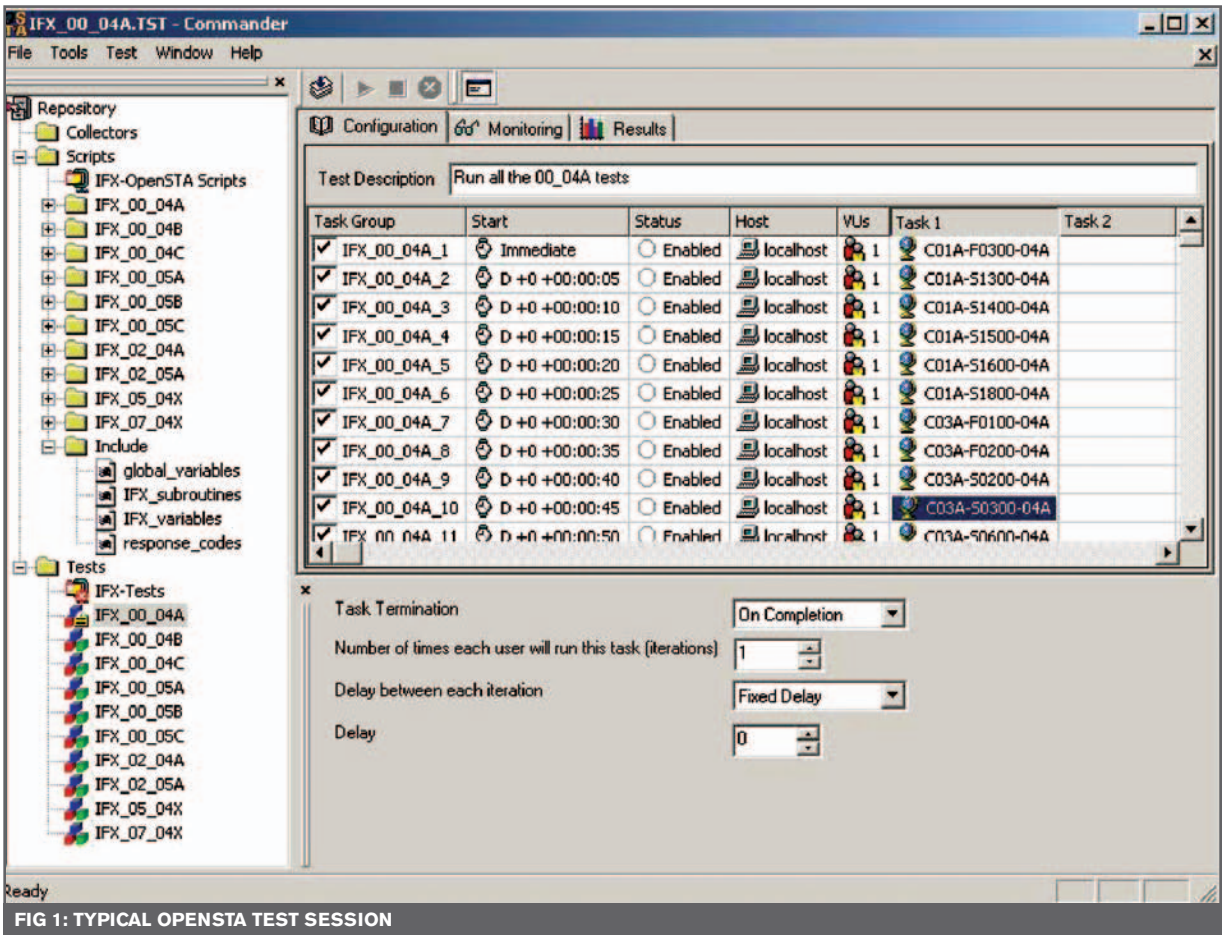


FIG 1: TYPICAL OPENSTA TEST SESSION

"The test scripts were designed to exploit OpenSTA's modular architecture"

test case results, investigating the test cases flagged as failed.

The TestLog.txt file tracked the execution of the test, recording each subroutine used by the test and any script execution errors. This file was useful while testing the logic of the test scripts and as a double check during formal execution of the test scripts.

Lessons Learned

In summary, OpenSTA satisfied our requirement for a test automation tool. The following lessons learned provide additional detail.

OPENSTA MET OUR NEEDS WITH SOME WORKAROUNDS

The test team was able to stay within the project test budget and schedule by creatively using a combination of OpenSTA features. As noted throughout this article, several workarounds were devised when OpenSTA didn't perform as expected or documented; however, the time spent finding and circumventing these issues was minimal. It should be noted that though OpenSTA can only be executed on a Windows platform, it can send and receive HTTP requests to servers deployed on any technology.

OPENSTA USERS SHOULD BE COMFORTABLE WITH SCRIPTING LANGUAGES

The OpenSTA Script Control Language lets testers devise complex test scenarios complete with test result validation. To do this, the test script authors must have programming experience in languages similar to the Script Control Language such as Unix shell scripting or IBM REXX.

MODULAR DESIGN GREATLY REDUCES THE EFFORT TO DEPLOY TEST SCRIPT ENHANCEMENTS

The test scripts were designed to exploit OpenSTA's modular architecture. By designing a generic test case template with calls to common subroutines, script enhancements were easily deployed to the test environment by simply updating two files: IFX_SUBROUTINES.INC and IFX_VARIABLES.INC. The hundreds of test case scripts developed for the project would automatically reference the script enhancements in these files.

TEST SCRIPT NAMING CONVENTION IS ESSENTIAL

The test team found it beneficial to adopt a rigorous naming standard for the test cases and the test scripts even though this wasn't a mandatory OpenSTA requirement. The naming convention helped the team organize all the test script components and facilitated incident reporting.

Part 1 of this article appeared in the August 2005 issue of WebSphere Journal *Vol: 4, Iss: 8.*

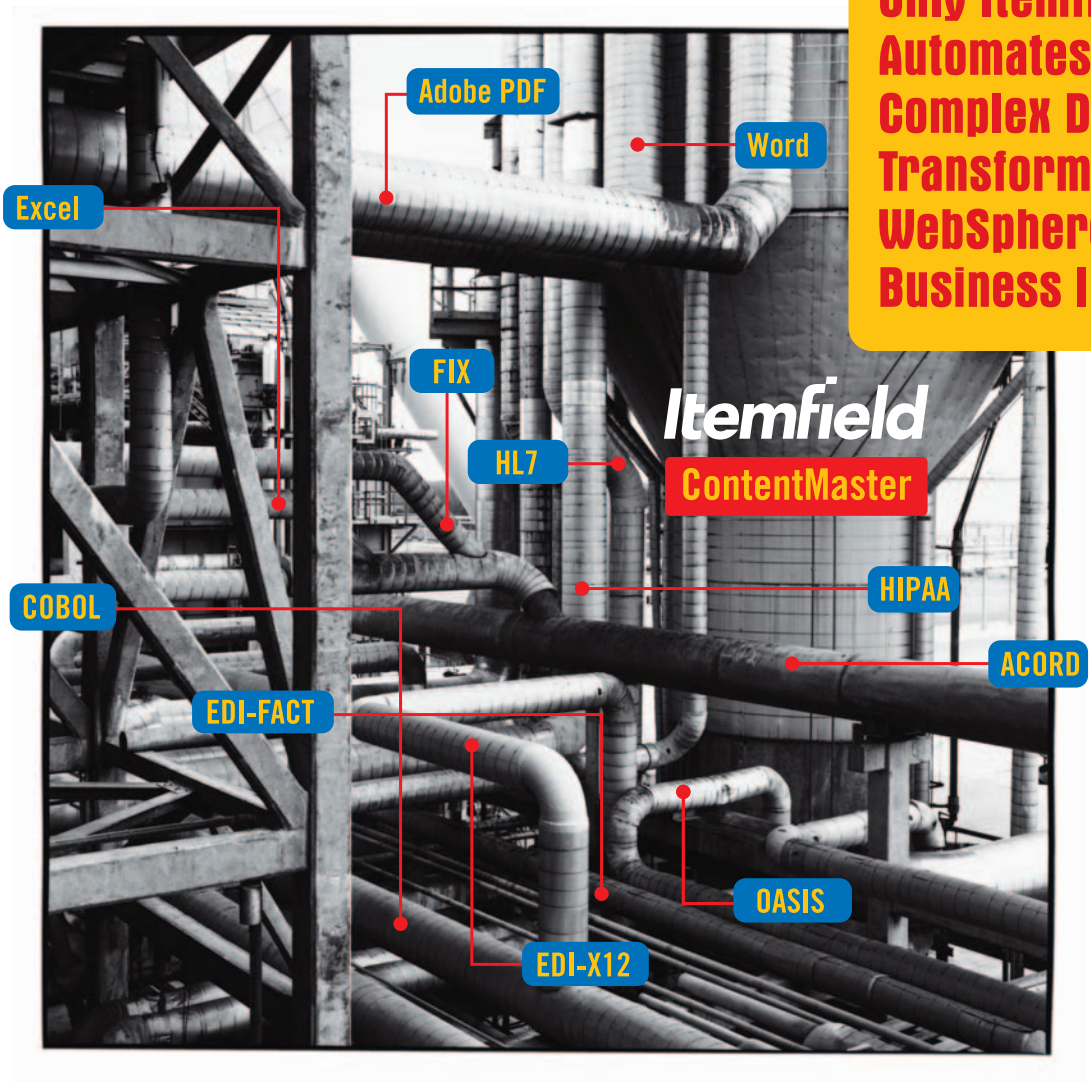
Resources

- http://opensta.org – the OpenSTA project's home page
- http://www.ifxforum.org – the IFX forum and source of the latest IFX specification
- http://www-306.ibm.com/software/awdtools/studiobranchtransformation/ - the home page of the IBM WebSphere Branch Transformation toolkit

LISTING 1: READ_RQ SUBROUTINE WITH LOOP CONTROL

```
1  !////////////////////////////////////////
2  ! This routine is called to read in the actual
     request message
3  !////////////////////////////////////////
4  !Read in the Request IFX Message
5  SUBROUTINE Read_Rq
6
7      DO looper = 1, MAXFILESIZE
8        NEXT REQUEST_LINE
9        SET THEXML = THEXML + REQUEST_LINE
10
11       !We are done if we have encountered the </IFX>
           string
12       Set Offset = ~Locate("</IFX>", REQUEST_LINE)
13       if (Offset >= 0) then
14               CALL GENERATE_RQUID
15               CALL REPLACE_RQUID_RQ
16               RETURN
17       endif
18     ENDDO
19
20  LOG "The Request loop count was exceeded"
21  END SUBROUTINE
```

Only Itemfield Automates Complex Data Transformation For WebSphere® Business Integration



Extend The Power And Reach of WebSphere Business Integration

Transform Any Data Format to XML and Back Again—Without Writing Code

Together, IBM and Itemfield solve the costly problem of data transformation to reduce the cost of business integration with a combination of products:

From Itemfield

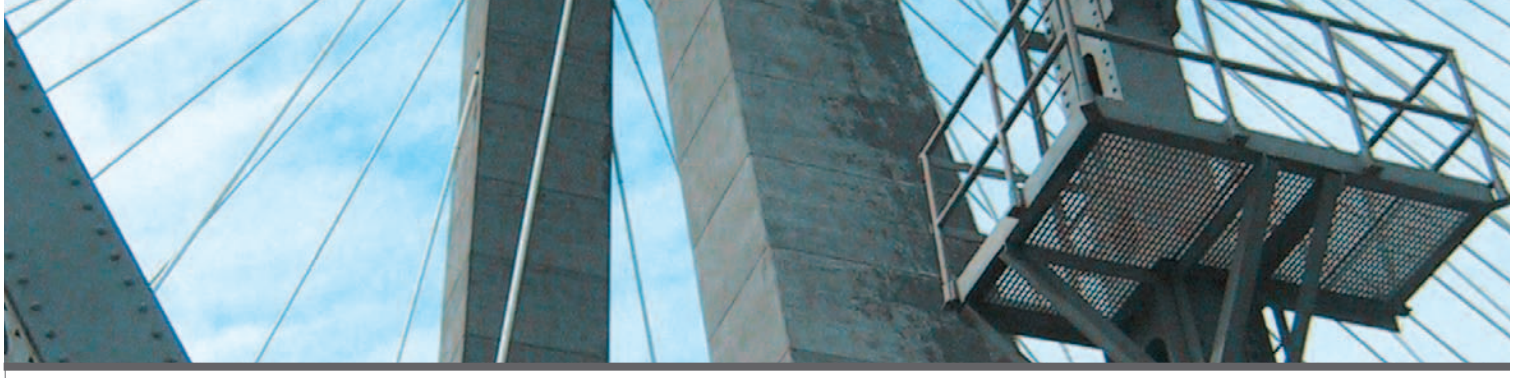
Itemfield ContentMaster™ Agent for WebSphere Business Integration Message Broker simplifies solutions to a broad range of integration needs and data formats with WebSphere Business Integration Message Broker, providing a high throughput interface between any data formats.

From IBM

IBM WebSphere® Business Integration Data Handler for Complex Data leverages Itemfield ContentMaster to enable the bidirectional conversion of a vast number of text and binary formats to and from WebSphere Business Integration business objects and can be used with any server in the WebSphere Business Integration product family.

Visit our website **www.itemfield.com/wbi** to obtain a copy of our latest white paper *"The Complexity Advantage: Solving the Requirements of Complex Data Transformation in Business Integration"* and to view the ContentMaster Solution for WebSphere Business Integration demo.

Itemfield | 1 800 838 1055 | www.itemfield.com/wbi



Information Reference Architecture

Part two of a two-part article

BY SCOTT SIMMONS

Scott Simmons is the worldwide technical lead architect for B2B integration for Worldwide WebSphere Business Integration and is an IBM Certified Senior IT Architect. Scott joined IBM in March 2002 from Peregrine/Extricity, where he was the director of Solution Technologies for Peregrine's Office of the CTO. In this role, Scott specialized in B2B solutions for the high technical manufacturing sector. Scott has over 20 years of experience in the IT industry. scottsim@us.ibm.com

Last month's article introduced the WebSphere Integration Reference Architecture, as well as an SOA framework and some services. This month's concludes by looking at more services and showing the architecture in action.

Development Services

The WebSphere Integration Reference Architecture rigorously recognizes the need for a comprehensive software development platform as core. It's important that the development platform encompasses the entire lifecycle of software development, including requirements analysis, modeling and design, component development, testing, and code maintenance. The tooling must be compatible with the concepts of Model Driven Architecture (MDA) and support the use of the best practices emerging via Service-Oriented Modeling and Architecture (SOMA) and Service Oriented Development of Applications (SODA) methodologies. These characteristics are incumbent on the IBM software development platform.

At a high level, development services in the WebSphere Integration Reference Architecture enable people to complete specific tasks and create specific output based on their skills, their expertise, and their role in the enterprise:

Business analysts, who analyze business process requirements, need modeling tools to design and simulate business processes.

Software architects need tools so they can model data, functional flows, and system interactions, and develop system topologies.

Integration specialists require the capabilities to configure and orchestrate components in developing integration solutions.

Programmers need tools to develop new business logic, such as J2EE components, portlets, and other custom service components.

Most importantly, the integration tooling environment promotes joint development, asset management, and deep collaboration between development roles through asset access and asset sharing. It's important to note that an organization's tool technologies and competencies will come from multiple vendors and, as a result, the presence of a multi-vendor framework like Eclipse is imperative in reducing the learning curve for the disparate roles in the development process. A standard tool framework like Eclipse also provides a common repository and base functions common across all the developer perspectives (for example, version control functions such as CVS and ClearCase, and utility functions such as edit, file, and print services). The development services provided through the WebSphere Integration Reference Architecture leverage the Eclipse base for their implementation.

Regardless of specific development roles, a high degree of collaboration is required in software development so each role is productive and efficient. The development tool platform provides an integrated set of tools that addresses the scope of integration development through role-based activities and across a multi-vendor tool environment. By separating concerns in the development process, each role can design, develop, and deploy artifacts specific to an individual's skills and responsibilities. A number of service functions are exposed as part of the development framework:

- **Model services** let analysts build visual models that are representative of business processes.
- **Design services** let models be staged into design, and include the ability to attribute the process with service components. These functions also enable the design and development of new integration components.
- **Implementation services** provide a way to move

developed artifacts into production as part of an organization's configuration management standards.
- **Test services** support unit test as well as integrated test capabilities as part of the overall development lifecycle.

Business Innovation and Optimization Services

In combination with many of the services mentioned above, business innovation and optimization services provide an infrastructure for continuous improvement and innovation, enabling businesses to adapt to changing market dynamics and everyday operational disruptions. Well-engineered BPM solutions support a holistic approach to business management, enabling aligned objectives, role-based visibility, contextual insight, and in-time actions. It's critical to note that BPM requires a set of differentiated capabilities that supports and incorporates the needs of both business and IT professionals.

The Common Base Event (CBE) specification provides the common base event model for the WebSphere Integration Reference Architecture. Originally developed by the IBM Autonomic team in collaboration with partners, CBE is an emerging OASIS standard that defines a common XML schema-based representation of events, supporting encoding of logging, tracing, management, and business events.

Within the WebSphere Integration Reference Architecture, BPM services consist of three primary groups, each of which supports both IT and business events:
- **Common event infrastructure (CEI) services** provide emission services for filtering and converting native events into CBE form, event store services for storing, querying, and managing the events, and event catalog services for storing, querying, and managing the event schemes.
- **Correlation services** provide policy-driven filtering and correlation of events to detect situations of interest.
- **Monitoring services** use an observation model to define the appropriate monitoring context, map events to the context, and compute and manage the associated performance measures and key performance indicators (KPIs).

Presenting metrics and KPI are done using interaction services. Likewise, any analytic and data integration needed to support a BPM solution is done using information services.

The link between the development platform and business innovation and optimization services is a key aspect of the WebSphere Integration Reference Architecture. The ability to characterize key performance indicators as part of the modeling environment and generate specific event flags as part of the process model enable analysts to build management functionality into their business processes. Following the implementation of the integration components, the BPM layer captures and delivers event data and statistics that can be input back into the modeling environment. This approach lets organizations support iterative process re-engineering through a continuous business process improvement cycle.

IT Services Management

Beneath all the capabilities of the WebSphere Integration Reference Architecture are management services for security, directory, system management, and resource virtualization. The security and directory services include the authentication and authorization functions required for implementation, for example, single sign-on capabilities across a distributed and heterogeneous system. System management and virtualization services include functions across the operating environment to manage server, storage, network, and other resources; for example, clustering and virtualization services enable the efficient use of computing resources based on load patterns and other factors. The ability to leverage grids as part of a grid computing platform is an integral part of IT services management. While many management services perform functions tied directly to hardware or system implementations, others provide functions that interact directly with the integration services provided in other elements of the architecture through the ESB. These interactions typically involve services related to security, directory, and IT operational systems management as part of the support operating environment.

Hardware and software management services provide the capabilities needed to run and operate enterprise systems effectively. Many of these services are independent of the other integrated services; others provide capabilities and data to other integrated services so they can effect business performance management and system operation.

"Using the concepts of 'separation of concerns' provides a clear alternative to traditional integration approaches"

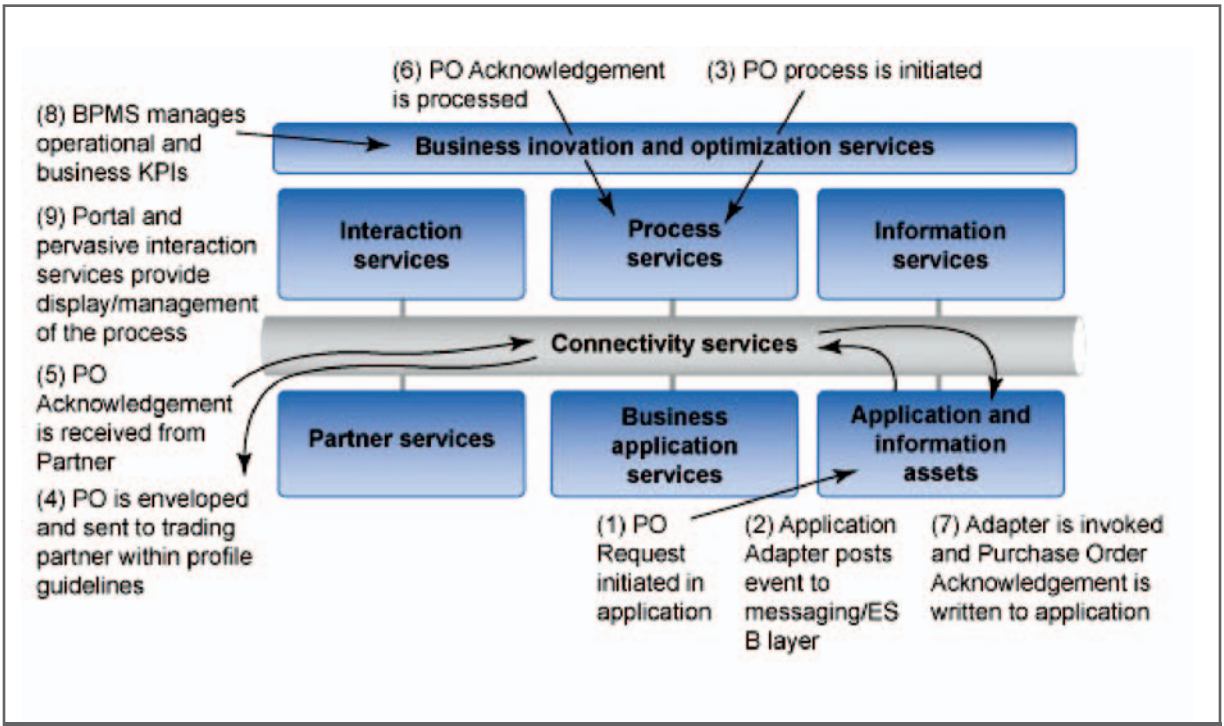


FIG 1: SERVICE INVOCATION IN THE WEBSPHERE INTEGRATION REFERENCE ARCHITECTURE

WebSphere Integration Reference Architecture in Action

In Figure 1, purchase order processing is an end-to-end composite integration process. The solution represents a sequence of services that are orchestrated though the components in the WebSphere Integration Reference Architecture.

The development tools and IT services management aren't shown. Development tools and management services support the development of integration components and the management of the underlying operational runtime framework respectively. Through an SOA-based approach to business integration, individual integration artifacts can be developed independently and then orchestrated to provide an overall solution. This provides a key differentiator to the traditional development of integration solutions as tightly bound applications. More importantly, any of the developed components (such as the adapter services or partner profile services) can be reused without affecting the operation of the purchase order process. The WebSphere Integration Reference Architecture enables organizations to be more strategic in the application of integration techniques to solve the challenges of on demand computing.

As discussed in *Patterns: Service-oriented Architecture and Web Services* (M. Endrei and others, 2004), "an on-demand operating environment provides the infrastructure needed to allow applications to be integrated using common standards and open technologies." This is a key tenet of the integration framework enabled via the WebSphere Integration Reference Architecture. The adherence to common standards and open technologies is the foundation of the IBM integration approach. The table below identifies some of the major standards and technologies in the WebSphere Reference Architecture:

Industry standards such as ACORD, SWIFT, HIPAA, UCCNET, and other key vertical frameworks and initiatives are also reflected in the solution framework. Adhering to standards represents IBM's approach to integration as a key enabler in building flexible and sustainable enterprise integration architectures.

IBM's WebSphere Integration Reference Architecture provides the most comprehensive integration framework in the industry. Besides supporting traditional programming and development solutions, the architecture has extensive support for developing service-oriented solutions for integration. This SOA support begins with the industry-leading support for

Imagine seeing EAI's impact on business process performance in real-time...

Now imagine implementing it in minutes, *instead of* months

To get your free copy of the informative new article

"3 Steps to Uncomplicating Business Process Management"

go to www.nastel.com/uncomplicate
or call 800-580-2344 or
+631-761-9100

Business happens in real time.
If something impacts your business processes, you need to know. *Now.*

AutoPilot™ from Nastel monitors and manages all of the complex events that comprise your business-critical transactions, giving you deeply granular visibility from end to end for all your business processes. AutoPilot lets you define customized, actionable alerts in the context of each user's specific business function to streamline your business processes and eliminate costly failures, bottlenecks and redundancies. With AutoPilot, you can see the impact of EAI on your business process performance in real time, on-demand.

Unlike other monitoring software, though, AutoPilot's unique architecture takes the complexity out of implementation. It doesn't require months of programmers and consultant's time before you reap the benefits. With AutoPilot, you simply point and click, and within minutes you're in total control.

NASTEL™ TECHNOLOGIES, INC.

Put Your Application Infrastructure On **AutoPilot**

MDA in the IBM Software Development Platform and is further implemented in the WebSphere foundation technologies in IBM WebSphere Application Server and IBM WebSphere MQ. The overall WebSphere Integration Reference Architecture is tied tightly to existing and emerging standards that support SOA development. Most importantly, the WebSphere Integration Reference Architecture supports all the major styles of integration, enabling linking people, processes, and information on a standards-based foundation. The solution architecture provides rich support for lifecycle modeling through development and implementation, and through enabling the monitoring of business and IT metrics as part of a complete development platform.

Conclusion

The WebSphere Integration Reference Architecture enables an enterprise to tightly link business requirements and technology solutions through a Service Oriented Architecture foundation. Applying this architecture and using the concepts of "separation of concerns" provides a clear alternative to traditional integration approaches.

Three key concepts – MDA, SOA, and BPM –underpin the WebSphere Integration Reference Architecture. A common framework of role-based development tooling embodying MDA enables the design and development of integration artifacts. These artifacts are tested and deployed in a runtime environment with a communication infrastructure provided through an Enterprise Service Bus architecture. These integration components leverage a common set of core infrastructure services for performance, scalability, security, and so on based on SOA. Finally, overlaying this runtime structure is an extensive monitoring and management environment embodying BPM.

In summary, the WebSphere Integration Reference Architecture provides a complete and comprehensive architecture that covers the breadth of integration needs in an enterprise. Integration services are clearly defined and delivered in a modular way promoting reuse and shared competencies across the organization. As new projects are implemented, services are easily added or extended, enhancing the scope and efficiency of enterprise integration efforts. The WebSphere Integration Reference Architecture lets organizations take a service-oriented approach to integration and avoid the pitfalls associated with traditional integration approaches.

Acknowledgements

The author would like to acknowledge the work of the Worldwide WebSphere technical sales team in creating and evolving the WebSphere Integration Reference Architecture, and the particular contributions of Bill Hassell, Bob Liburdi, and Bob Knaus. The author would also like to acknowledge Dan Wolfson, IBM's CTO for business integration, for his thorough review of this article.

Resources

- M. Endrei (and others), *Patterns: Service-oriented Architecture and Web Services* (Redbook), IBM TSO, 2004.
- C. Perks and T. Beveridge, *Guide to Enterprise IT Architecture* Springer-Verlag, New York, 2003.
- G. Booch et al, *An MDA Manifesto* MDA Journal May 2004.
- M. Keen and others, *Patterns: Implementing an SOA using an Enterprise Service Bus* IBM Redbook SG2463, July 2004.
- *IBM Service-Oriented Modeling and Architecture*, IBM Business Consulting Services Overview, 2004.

SERVICE FUNCTION	RELEVANT STANDARDS
Enterprise Service Bus	JMS, J2EE, SOAP, XSLT, WSDL, UDDI
Development tools	Eclipse, J2EE, J2SE, J2ME, XML, UML, Java Server Faces, SWT, XMI, WS BPEL, SQLJ, JDBC, XSLT, WSDL, UDDI
Business performance management tools	W3 Common Log Format, WS-DM initiatives, CEI/CBE
Interaction services	WSRP, JSR 168, Java Server Faces, VoiceXML, J2EE
Process services	J2EE, BPEL4WS, WSDL, UDDI
Information services	XQuery, SQL, JDBC/ODBC
Partner services	FTP, sFTP, HTTP, HTTP/S, RosettaNet, SMTP, JMS, SOAP/HTTP, WMQ, cXML, EDI (X12, EDIFACT and others)
Business application services	J2EE (JNDI, EJB, JSP, JTA, JAAS, JAXP, JAXR, JMX and others)
Application and information assets	J2C, JMS, IIOP, JDBC, CICS, IMS, 3270/5250

TABLE 1: WEBSPHERE INTEGRATION REFERENCE ARCHITECTURE STANDARDS AND TECHNOLOGIES

How to build and test applications

The Rational Professional Bundle

BY: ERIC NAIBURG

Eric Naiburg is group market manager of desktop products for IBM Rational Software. He is responsible for market strategy, planning, and messaging around Rational's desktop products including requirements, analysis, design, construction, and software quality solutions. Prior to his current position, Eric was manager of product management, focusing on the IBM Rational Rose and IBM Rational XDE product lines. He came to Rational from Logic Works Inc., which is now a part of Computer Associates where he was product manager for ERwin and ModelMart. His published works include *UML for Database Design* and *UML for Mere Mortals*.
ericnaiburg@sys-con.com

The IBM Rational Professional bundle can help your development organization to accelerate software development by integrating systems analysis, design, coding, and developer and systems test activities around a common underlying tool infrastructure. This simplifies the development team's ability to learn the tools and makes it easier for them to share information with the rest of their team. The bundle's built-in version control and integration with other team tools enable them to protect team assets, create and manage requirements, track quality-management efforts, and understand how a change in one project artifact could conceivably impact others.

Creating applications that are optimized to run on WebSphere Application Servers doesn't happen by mistake. The tools in the bundle are optimized for WebSphere software and provide strong support for developing J2EE applications while automating their optimization of WebSphere software. There are built-in test server environments for application servers, databases, portal servers, and more directly on the local machine to help facilitate convenient iterative development for unit testing, debugging, component testing, and functional testing during the development lifecycle. This has tremendous productivity and cost-savings benefits.

The Rational Professional bundle supports geographically distributed teams in many ways:

- Share architectures through visual modeling
- Communicate requirements in a common and standard way
- Version control in a singular manner to provide instant access to the most up-to-date code artifacts
- Change management to assign work across the entire team and understand the project status at any point in the lifecycle
- Flexible license model
  a. Floating licensing gives teams the ability to share licensing around the clock
  b. Authorized user licensing gives the teams that need constant access a license to use it
  c. Fixed-term Authorized User licensing gives the team the ability to purchase a 12-month license that can be renewed depending on the project and organization's needs.

The Tools in the Rational Professional Bundle

There are eight tools in the Rational Professional bundle: design, construction, software quality (testing), and software configuration management (version control). In addition, there are two process guidance configurations inside the Rational Professional bundle. Rational Software Architect includes a configuration for software architects and Rational Application Developer includes one for developers

Conclusion

With the development capabilities the IBM Rational Professional bundle provides, it's easy to see the actual business value derived from the bundle and the IBM Rational Software Development platform. Developers increase their productivity and improve the quality of the software they develop through a single unifying workbench that has a variety of well-integrated development tools. The bundle lowers development and operations costs as well, because the unified workbench reduces administration costs, facilitates skills transfer, and reduces user product training needs. And the open extensible development platform based on Open Source provides even more advantages by reducing vendor lock-in and granting extensibility without vendor dependency.

For more information on the

IBM Rational Professional bundle as well as the IBM Rational Software Development platform, see www.ibm.com/software/awdtools/probundle and http://www.ibm.com/software/info/developer.

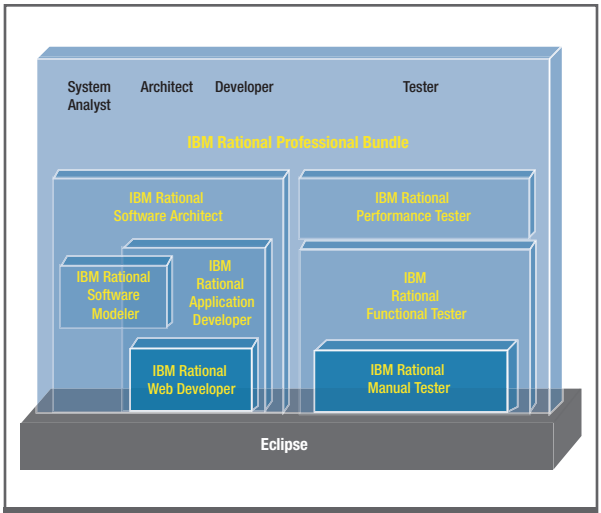


FIG 1: THE IBM RATIONAL PROFESSIONAL BUNDLE HELPS DEVELOPMENT ORGANIZATIONS ACCELERATE SOFTWARE DEVELOPMENT BY INTEGRATING SYSTEMS ANALYSIS, DESIGN, CODING, AND DEVELOPER AND SYSTEMS TEST ACTIVITIES AROUND A COMMON UNDERLYING TOOL INFRASTRUCTURE

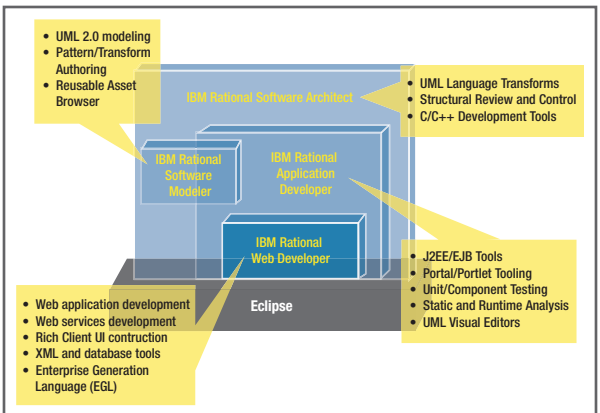


FIG 2: IBM RATIONAL PROFESSIONAL BUNDLE: DESIGN AND CONSTRUCTION TOOLS

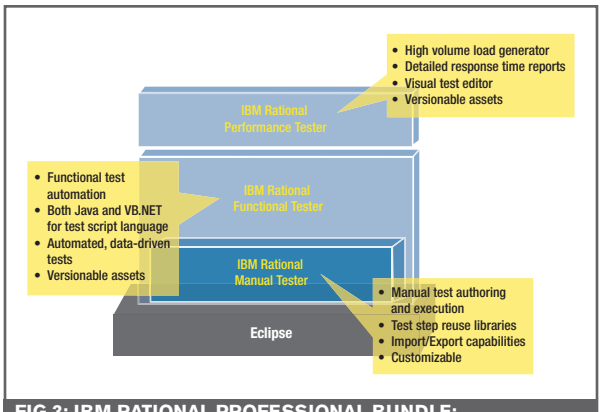


FIG 3: IBM RATIONAL PROFESSIONAL BUNDLE: SOFTWARE QUALITY TOOLS

Interview: IBM's Adam Jollans Outlines Linux Strategy

Cost is very important...but there's something deeper here

INTERVIEWED BY
ROGER STRUKHOFF

Roger Strukhoff, editor-in-chief of ***WebSphere Journal***, is West Coast Bureau Chief for the SYS-CON News Desk, and President of www.wdva.com. He spent 15 years with Miller Freeman Publications and The International Data Group (IDG), then co-founded CoverOne Media, a custom publishing agency that he sold in 2004. His work has won awards from the American Business Media, Western Press Association, Illinois Press Association, and the Magazine Publishers' Association. Read his blog at http://www.rssblog.linuxworld.com. roger@sys-con.com.

Adam Jollans is IBM Software Group's senior Linux strategist. He leads its worldwide Linux marketing strategy. In this role he is responsible for defining Linux marketing activities at the software category level and integrating them with both IBM's corporate Linux marketing strategy and specific DB2, WebSphere, Lotus, Tivoli, and Rational Linux marketing activities.

This role covers a wide spectrum of activities from understanding customer requirements for Linux software to working with technical strategy to define the Linux software strategy and briefing analysts and press on IBM's Linux strategy and offerings. He is now based in London, following a two-year stint in Somers, NY.

He has been involved with Linux since 1998, and prior to his current assignment, he led the European marketing activities of IBM Software on Linux.

Adam joined IBM in 1984, and since then has worked in a range of technical, sales, and marketing roles – most of them with PC and small systems hardware and software. At IBM he has worked with a leading UK Bank on designing and implementing its distributed branch information system. He also spent two years assigned to IBM development in Florida working on advanced operating systems.

Before joining IBM, he worked as an application software designer and programmer, developing commercial graphics and business software.

Adam graduated from Cambridge University in 1980 with a degree in computer science, and is a chartered engineer and a member of the British Computer Society.

He was coeditor of "OS/2 2.11 Power Techniques," published by QUE, and has also written IBM Redbooks and white papers on PC hardware and software.

Adam recently answered some questions for us about IBM and its Linux strategy.

WEBSPHERE JOURNAL: BRIEFLY DESCRIBE IBM'S LINUX STRATEGY TODAY. HOW DOES IT FIT IN WITH IBM'S OVERALL ENTERPRISE IT VISION FOR YOUR CUSTOMERS?

Adam Jollans: Let's start with IBM's overall strategy. This is our On Demand Business vision – how do we help our customers evolve their businesses to be more competitive and more dynamic. This means they have to be able to quickly sense and take advantage of new opportunities, and quickly identify problems and handle them.

"To be an On Demand Business in today's world, you need an On Demand enterprise IT infrastructure "

To be an On Demand Business in today's world, you need an On Demand enterprise IT infrastructure – a dynamic, efficient, and flexible infrastructure that integrates all parts of your business plus your customers, partners, and suppliers. And Linux offers a great way of delivering this – it's cost-effective, it runs on multiple hardware platforms, it virtualizes well, and it's easy to automate.

Our Linux strategy is integrated in every part of IBM – running Linux on our complete range of eServer hardware, enabling all of our key middleware on Linux, offering support, education, and services for Linux, helping to develop the Linux ecosystem of ISVs and business partners, and contributing back to Linux through the hundreds of engineers and programmers in our Linux Technology Center.

WEBSPHERE JOURNAL: WHAT SORT OF CUSTOMERS DO YOU HAVE? ARE YOU CONCENTRATING ON CERTAIN VERTICAL MARKETS? WHAT SORT OF BUSINESS PROBLEMS ARE THEY SOLVING?

AJ: We have thousands of Linux customers in every industry and of every size – from Charles Schwab on Wall Street to the city of Bergen in Norway. They're all typically looking to reduce costs, increase reliability and security, and gain flexibility by using Linux. Charles Schwab, for example, is using a Linux supercomputer to do investment calculations faster and answer their client queries on the phone. The city of Bergen is reducing IT costs in its public schools by moving to Linux.

Some industries are moving especially fast in their adoption of Linux – this includes finance, retail, and government. We're seeing Linux being used as banks re-architect their branches and move to multi-channel banking – for example, Banco Popolare di Milano in Italy is using Linux and WebSphere on the server and Linux and Mozilla on the client. We're seeing Linux being used in e-commerce sites and point-of-sale systems in retail – for example, Pioneer Petroleum in Canada is using IBM Workplace on Linux in its stores at its gas stations. And we're seeing Linux being used around the world in local and national governments as they provide e-access to their citizens without increasing costs.

WEBSPHERE JOURNAL: LINUX IS OFTEN MENTIONED FAVORABLY FOR ITS COST. DOES IBM BELIEVE THAT TCO IS A KEY BENEFIT OF DEPLOYING LINUX?

AJ: Yes, definitely. Saving money is often the first reason customers consider Linux. We're seeing customers move both Unix and Windows systems to Linux to save money. A recent report by the Robert Frances Group found that Linux is 40% less expensive than a comparable x86-based Windows solution and 54 percent less than a comparable SPARC-based Solaris solution based on a three-year period of ownership. Some of this is down to the hardware – more efficient use of lower-cost hardware. Some of this is because of reduced licensing costs. And some of this is because Linux is reliable and secure, and so needs fewer systems administrators.

However we're also hearing from customers that they are getting unanticipated second-stage benefits from Linux as well. Another recent report – this time by Pund-IT – conducted in-depth interviews with long-term Linux customers and found that they were gaining more flexibility and better use of skills, as well as the cost savings. In fact, all these customers were moving towards being On Demand Businesses – and Linux was helping them get there.

WEBSPHERE JOURNAL: WITHIN THE CONTEXT OF COST, IT SEEMS THAT MANY E-GOVERNMENT INITIATIVES ARE BEING DEVELOPED AND DEPLOYED WITH LINUX. IS THIS SOMETHING THAT IBM HAS SEEN AS WELL?

AJ: Absolutely. In fact, there are a number of reasons why e-government initiatives are using Linux. Cost is very important – as a government, you're faced with the challenge of providing more (additional e-government applications) without increasing taxes. Linux has definitely helped here – for example, the Regione Lazio regional government, which includes Rome, was able to repurpose an existing IBM zSeries system with Linux and WebSphere to provide its new e-government Web site. And in Brazil, the government is using low-cost Linux PCs to extend access to the Internet to a wider cross-section of society.

However there's also something deeper here. Linux gives you freedom and choice. And that's especially important to governments – they're not locked into

Adam Jollans
IBM Software Group's Senior Linux Strategist

"We have thousands of Linux customers in every industry and of every size… They're all typically looking to reduce costs, increase reliability and security, and gain flexibility by using Linux"

hardware from one software vendor or one architecture. Since Linux source code is freely available, they can examine it for security backdoors – and have their own nationals provide support. Linux gives governments control over their own destiny.

WEBSPHERE JOURNAL: HOW DOES LINUX ADOPTION IN WESTERN EUROPE COMPARE WITH NORTH AMERICA? DO YOU SEE OTHER REGIONS MOVING TOWARD LINUX AS WELL?

AJ: It's both ahead and behind! Or rather, it varies from country to country in Western Europe. Germany, for example, has been an early adopter of Linux – perhaps partly due to the local SuSE Linux distributor being based there. And we're seeing rapid adoption in the Nordics, Italy, France, the UK, and so on.

But the countries where we are seeing Linux moving especially fast are the emerging economies – China, India, Russia, Central Europe, Latin America. They aren't encumbered by the past – and they have a large number of very bright programmers and engineers coming out of their universities all skilled in Linux. Their governments have also seen how Linux can enable them to build their local IT ecosystems – and so help develop their economies.

WEBSPHERE JOURNAL: HOW OFTEN DO YOU SEE PURE-LINUX SHOPS VERSUS ENTERPRISES THAT HAVE A MULTITUDE OF OPERATING ENVIRONMENTS DEPLOYED?

AJ: There's very few 'pure anything' shops these days in IT. What we typically see is a mixture of hardware platforms and operating environments deployed. And even if an IT shop standardized on one operating environment five years ago, today they have to handle a mixed environment since they need to integrate their IT with the their partners, suppliers, and customers – many of which will have made a different decision. The Internet has changed everything.

So we see Linux integrated with Windows, Linux integrated with zSeries mainframes, Linux integrated with Unix, and so on. Sometimes this is even on the same system using virtualization technologies – and this is only going to increase more as IT shops look for better utilization of IT resources and more dynamic computing.

WEBSPHERE JOURNAL: SUPPORT, OR LACK THE REOF, SEEMS TO BE A BIG POTENTIAL DRAWBACK TO LINUX. HOW DOES IBM HANDLE THIS ISSUE?

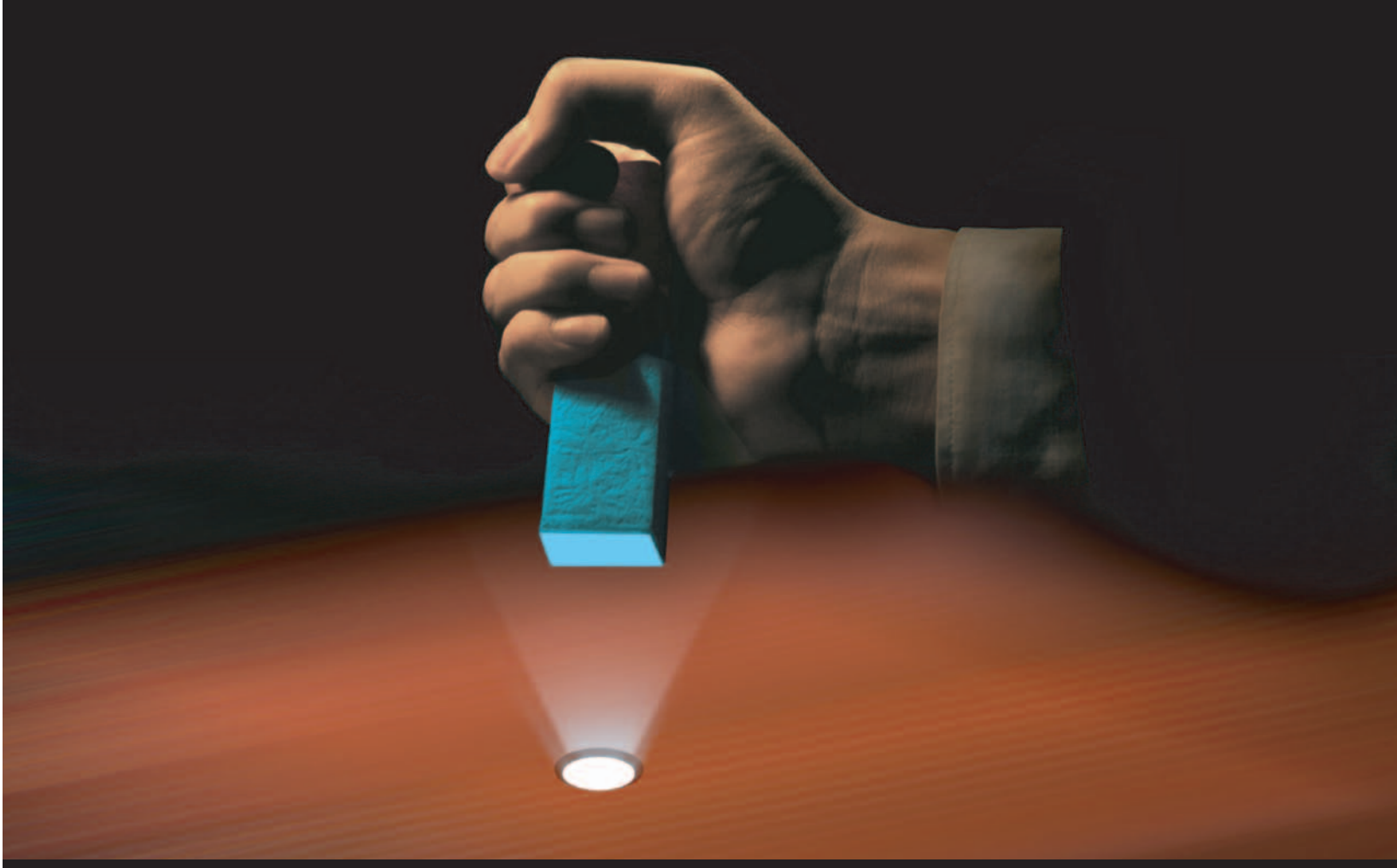
AJ: I think this was an issue in the past, but most customers have moved on from there. Today you can purchase 24x7 support contracts for Linux from IBM and others, just as you can purchase 24x7 AIX or DB2 support contracts. In fact we've been offering these for five years now. Enterprise customer requirements don't change in this sense – they want to be confident their mission-critical systems are going to be available when they need them.

We offer first- and second-level support, we link with Red Hat and Novell SuSE on third-level support, and we have very deep skills inside IBM on Linux through the hundreds of our programmers and engineers who are contributing to Linux and its development.

WEBSPHERE JOURNAL: DOES IBM SUPPORT OSDL, AND IF SO, HOW?

AJ: Yes – IBM was one of the founding members of OSDL, w e're on the board, and we're working with OSDL on a variety of Linux activities. This ranges from our involvement in its working groups such as Desktop Linux…to providing enterprise hardware to help their developers…to working with OSDL and other vendors on joint marketing activities. A couple of months ago, OSDL hosted a joint press announcement about Linux adoption in the retail industry, which brought together IBM, HP, Red Hat, Novell, and BakBone Software. How often do you see competitors do that?

One of the most important characteristics of Linux is the way it has opened up possibilities for collaborative innovation, between companies, universities, and customers. And OSDL is a key group in enabling this to happen.



Think Crystal is a good fit for your Java application? Think again.

Think JReport.

Forcing a Windows reporting solution into a J2EE environment is never going to result in a perfect fit. Only JReport is built from the ground up to leverage J2EE standards and modular components for seamless embedding in any Web application.

JReport is ready out-of-the-box, with all the tools necessary to empower your application with any reporting requirement. From reusable, shared report components to flexible APIs, JReport is the most complete embedded reporting solution available.

With the ability to scale to multi-CPU and server clusters, JReport is a perfect fit for any reporting workload. Load balancing and failover protection provide peak performance and uninterrupted access to critical business data. In addition, JReport integrates with any external security scheme for single sign-on.

JReport's ad hoc reporting lets users access and analyze data on demand, from any browser. And, with cascading parameters, embedded web controls for dynamic sorting and filtering, drill-down and pivot, JReport ensures users get the information they want, when they want it, and how they want it.

See for yourself why over half of the world's largest organizations have turned to JReport to enable their J2EE applications with actionable reporting.

When it comes to embedded Java reporting, JReport is the perfect fit. Download a FREE copy of JReport today at www.jinfonet.com/wj4.

JReport
JINFONET SOFTWARE

IBM Roundtables Its Linux Strategy

Finds advantages beyond TCO

IBM has developed an active Linux strategy for its enterprise IT customers based on its belief that "over the last decade, Linux has seen increasing international adoption as the platform of choice both for cost savings and broader business benefits like flexibility, scalability and security," according to a company statement.

Company management also believes that "as these Linux deployments start to mature, enterprises are reaping unexpected bonuses. Initially considered for the lower TCO, which is still a driving force, these companies are now enjoying consolidated application workload, reduced systems management investment, and optimized cost performance across the value chain."

So IBM recently held what it called a roundtable teleconference in which it presented recent findings about Linux and how it fits into customers' enterprise IT strategy. Adam Jollans (interviewed in this issue of *WebSphere Journal*) and Scott Handy, vice-president, Worldwide Linux, IBM Systems and Technology Group, represented IBM during the call.

Also present were Charles King, principle analyst, Pund-IT Research, and two members of the Robert Francis Group, Cal Braunstein, CEO and executive director of research, and Chad Robinson, principle analyst.

The event was called "TCO Benefits of Linux." The discussion elicited several overarching messages that these executives share about the Linux operating system today. These findings included:

- Linux continues to lead in total cost of ownership over Windows x86 (40%) and SPARC Solaris (54%) according to Robert Francis Group research
- Production results show Linux administrators can often manage more systems that Windows administrators in a given amount of time, resulting in reduced management costs and less overall complexity in management activities
- Linux can reduce on-going licensing and maintenance costs, especially where per-processor licensing models are used
- Linux also provides a number of strategic benefits not available in Solaris or Windows. It runs on a broad range of hardware platforms and is cost-competitive when scaled well both horizontally and vertically
- It enjoys solid ISV support, and is available from a number of vendors in several licensing and support models
- Linux benefits include minimal initial acquisition costs, as well as significant reduction or elimination of ongoing software licensing costs compared to proprietary systems
- Linux also allows users to choose lower-cost hardware or extend the life of existing systems, potentially changing hardware ROI
- Linux also helps free businesses to choose the hardware solutions that best meet the needs of their business applications and workloads
- Second-stage benefits arise from longer-term tactical and strategic issues such as Linux adoption for or adaptation to new IT business processes
- Linux can also help simplify a variety of IT management efforts; enhancing IT staff performance and lowering system maintenance and management costs

A pair of online white papers was also made available to developers and IT managers who are interested in this topic. They can be found at:

- Robert Francis Group: http://www1.ibm.com/linux/whitepapers/robertFrancesGroupLinuxTCOAnalysis05.pdf
- Pund-IT: http://www1.ibm.com/linux/whitepapers/pundITLinuxSecondStageBenefits.pdf

ENGAGE AND EXPLORE...

The Technologies, Solutions and Applications that are Driving Today's Initiatives and Strategies...

CALL FOR PAPERS NOW OPEN!

10th International SOAWebServicesEdge conference+expo

edge 06

The Sixth Annual SOA Web Services Edge 2006 East - International Web Services Conference & Expo, to be held May 23, 24, 25, 2006, announces that its Call for Papers is now open. Topics include all aspects of Web services and Service-Oriented Architecture

Suggested topics...

- *Transitioning Successfully to SOA*
- *Federated Web services*
- *ebXML*
- *Orchestration*
- *Discovery*
- *The Business Case for SOA*
- *Interop & Standards*
- *Web Services Management*
- *Messaging Buses and SOA*
- *Enterprise Service Buses*
- *SOBAs (Service-Oriented Business Apps)*
- *Delivering ROI with SOA*
- *Java Web Services*
- *XML Web Services*
- *Security*
- *Professional Open Source*
- *Systems Integration*
- *Sarbanes-Oxley*
- *Grid Computing*
- *Business Process Management*
- *Web Services Choreography*

CALL FOR PAPERS NOW OPEN!

2006 ENTERPRISE> OPENSOURCE CONFERENCE+EXPO

The first annual Enterprise Open Source Conference & Expo announces that its Call for Papers is now open. Topics include all aspects of Open Source technology. The Enterprise Open Source Conference & Expo is a software development and management conference addressing the emerging technologies, tools and strategies surrounding the development of open source software. We invite you to submit a proposal to present in the following topics. Case studies, tools, best practices, development, security, deployment, performance, challenges, application management, strategies and integration.

Suggested topics...

- *Open Source Licenses*
- *Open Source & E-Mail*
- *Databases*
- *ROI Case Studies*
- *Open Source ERP & CRM*
- *Open-Source SIP*
- *Testing*
- *LAMP Technologies*
- *Open Source on the Desktop*
- *Open Source & Sarbanes-Oxley*
- *IP Management*

Submit Your Topic Today! www2.sys-con.com/events

Sponsored by

WebServices JOURNAL · XML JOURNAL · .NET DEVELOPER'S JOURNAL · eclipse developer's journal · WebSphere JOURNAL · information STORAGE+SECURITY journal · wldj · JDJ · LINUXWORLD MAGAZINE · MX developer's journal · asp.netPRO · SD Times · CoDe · Software Test & Performance

*Call for Papers email: jimh@sys-con.com

Attention Exhibitors:
An Exhibit-Forum will display leading Web services and OpenSource products, services, and solutions

For Exhibit and Sponsorship Information – Call 201 802-3066

Produced by SYS-CON EVENTS

Performance Consideration

Balance trade-offs and maximize performance

BY RAINER DZIERZON, KLAUS NOSSEK & MICHAEL MENZE

Rainer Dzierzon is team lead of the WebSphere Portal performance team located in the IBM Development Laboratory in Boeblingen, Germany. Since 1990, he has worked on numerous software projects for several companies in the area of database performance tools, text search, SyncML, online banking, and financial services architecture and standards. He and his team work as part of the WebSphere Portal development community to provide developers with the necessary insights of performance behaviour, and to consult customers and solve their performance issues. He holds a diploma in Computer Engineering from the University of Cooperative Education, Stuttgart.

This article was first published on developerWorks WebSphere at www.ibm.com/developerWorks/websphere.

This article provides general guidance for creating well performing custom code for IBM WebSphere Portal. Custom code does not only refer to portlets (although they are the most common programming model for portals), but also includes code for WebSphere Portal themes and skins. Since these are implemented using the same basic technologies used for portlets, many performance considerations are applicable in the same way.

Regarding portlets, this article focuses on standardized portlets following the Java Portlet Specification, *JSR 168,* and the corresponding implementation in WebSphere Portal. The basis for this article is WebSphere Portal V5.1 or higher, although most guidelines and recommendations presented here will apply regardless of the version of WebSphere Portal you are running.

This series of articles explains how to set up and exploit the deployment parameters of a portlet application to optimize portal and portlet performance, since this is the last step in creating custom portal code. In contrast, general tuning of WebSphere Portal (that is, administrative actions that are performed after custom code has already been created and deployed) will not be covered here. (Look for another series of articles, beginning in the October issue of *WebSphere Journal*, that covers *WebSphere Portal performance tuning.* Those articles, combined with this series, will provide an excellent resource on portals and performance.)

This article is intended for programmers, designers, and architects involved in building custom portal applications and want to improve their understanding of potential performance issues with regard to custom code.

WebSphere Portal Environment Overview

IBM WebSphere Portal is built upon the IBM WebSphere Application Server product. As a result, the programming environment for custom portal code is threefold, with important corresponding implications:

- **WebSphere Portal and all its components are Java-based programs.** Thus, in general, best practices with regard to programming high-performance Java code should be followed.
- **WebSphere Portal is a J2EE application running on top of an application server platform.** J2EE embraces multi-threading; J2EE containers typically take a thread-per-request approach to handle the request burden. Any implementation or performance considerations inherent with using this mechanism should likewise be taken into account.
- **WebSphere Portal provides APIs to expand portal functionality.** Many tasks can be programmed in many different ways. Differences that affect performance should be addressed as a priority.

The next sections introduce some general performance considerations for different parts of the portal programming environment.

Java

Clearly, this section is not intended as a complete treatment of Java performance. Instead, we present here the items we have found most useful when dealing with WebSphere Portal development, and suggest some resources that can help provide an indepth understanding of Java performance.

Basic Java Performance

In this section, we will cover some general performance items that will apply to most Java programs. Although these recommendations may not yield order of magnitude performance improvements, they can make you aware of the importance of low-level program execution performance while in the development phase.

- **Use java.lang.StringBuffers instead of java.lang.String instances when the modification of strings is required.** String objects in Java are immutable, whereas StringBuffer objects are mutable. Whenever text is to be appended to or deleted from a String, a new object is created under the covers and the old object is discarded. Hence we prefer this:

```
StringBuffer sb = new StringBuffer("Hello ");
sb.append(var).append(" World");
```

over this string concatenation:

```
String s = "Hello" + var + " World";
```

You can sometimes further improve performance by setting the initial capacity of a StringBuffer; the class is designed to automatically grow when it can no longer hold all its data. However, there is a performance penalty here, since a StringBuffer must work to transparently increase its size and shift the data around. For example, if a StringBuffer is used as a collecting parameter (meaning that more and more data is added to it), you should compute the appropriate buffer size before instantiating it so that it never needs to grow in size.

- **Avoid costly I/O operations in server-side programs.** Execution of at least this thread is blocked during the I/O operation; if other threads also must wait for the disk, then system response times will easily and rapidly increase. Unless logging is being performed (for example, of exceptions or site access information), WebSphere Portal does not cause any disk access on its own. We will discuss I/O more later.
- **Minimize the number and length of synchronized code blocks.** The synchronized keyword lets only one thread enter a code block at a time. The longer a synchronized code block requires for execution, the longer other threads wait to enter that block. We will discuss synchronization more later.
- **Avoid expensive calculations and method calls.** For example, retrieving the current time information using System.currentTimeMillis() is rather expensive. If you need time information in your code, verify whether you need the exact time in every case, or (for example) if accuracy to the nearest second would be sufficient. If you have many get-time calls in your code path, but millisecond-accuracy is not mandatory, an alternative could be to determine the time at the beginning of a request and simply reuse this information throughout the request.
- **Restrict your use of exceptions.** In general, exceptions in Java should be used to indicate a faulty situation. Do not use exceptions to indicate the success of an operation, especially because creating the exception stack traces is a time-consuming effort for the JVM, and because the traces can be very deep in a WebSphere Portal system.
- **Take care when using the Java Reflection API.** This API adds powerful options to dynamic code execution, but also imposes severe performance penalties in terms of method execution times in exchange for this flexibility. In general, try to avoid the use of the Java Reflection API in portal code. However, if it is necessary to have reflection calls, try to have them in init methods so that they are not executed during every request.

Memory Consumption and Garbage Creation

While memory often is not a predominant performance issue for Java client software, it is a major concern for J2EE applications, mainly because enterprise applications are typically accessed by many users at the same time. For an application server to be efficient, the available resources, including memory, CPU and bandwidth, are shared among the clients' requests. There are three major memory issues we want to mention:

- **Keep the amount of temporary objects to a minimum.** That means, try to reuse objects as often as possible, and do not create new object instances too often. The more objects that are created, the more frequently the JVM garbage collector has to reclaim memory and, at least partially, interrupt request handling at that time. Creating many objects also tends to increase heap fragmentation, which leads to even more garbage collection cycles. For example, do not create objects prematurely:

```
String logData = "Parameter 1: " + param1;
if (logger.isLogging(DEBUG)) {
logger.log(logData);
}
```

In this example, logData should only be created after evaluating the condition. Caching and pooling can be helpful techniques to reduce temporary object creation.

- **Keep your permanent memory footprint low.** Do not read too much information into memory; rather, use caches to hold important information. Sometimes it is possible to change the data type for a piece of information. Date information, for example, can be held within a java.util.Date object or in a long variable. Objects typically are larger and somewhat slower to process than primitive data types. It might depend on the surrounding APIs and data structures

Klaus Nossek is a performance analyst on the WebSphere Portal team at the IBM Development Laboratory in Boeblingen, Germany. He received a diploma in Computer Science from the University of Stuttgart in 1996. After working for several years on various software projects, he joined the IBM WebSphere Portal division in 2003. Currently, he belongs to the WebSphere Portal performance team. His main working areas are developing of performance scenarios for WebSphere Portal, and Java code profiling for detailed performance analysis.

Michael Menze joined IBM development in 2002 after studying with IBM for three years. He works in the Boeblingen development team for WebSphere Portal and is responsible for WebSphere Portal performance analysis, and the introduction of performance improvements.

which data type is preferred. In general, a higher memory footprint leads to higher garbage collection rates and additional pause times during request processing.

- **Check your application for memory leaks.** When leaks occur, they typically occur within Java collection classes. For example, you have a java.util.Map and, under certain conditions, data is added to the map but never removed from it. Memory leaks lead to more and more consumption of memory that is reserved by the Java heap, and the garbage collector will be able to free up less memory over time. Thus, garbage collection will occur more frequently, and, finally, the portal system will become unresponsive. To complicate matters, memory leaks are often uncovered by long-running tests only, but there are tools around that can assist you with this analysis.
- **Code design for performance and scalability.** There is much to remember when designing and developing code for scalability. Three things of particular importance are: Caching, pooling, and information pre-fetch.
- **Caches store already computed results.** For example, you can retrieve information from a backend system, but rather than copy every possible object from the store to memory, just load small pieces and put them into a cache. This way, the information is available for later reference, possibly in another request, or even for another user.

  Caches almost always take the form of object maps with an upper size limit. A cache also has to have a way of knowing when something is unlikely to be asked for again so that it can be removed from the cache when appropriate. Such evictions are typically determined by a "time-to-live" (TTL) or "least-recently-used" (LRU) algorithm. Furthermore, the client using a cache cannot be confident that it will successfully retrieve an object from the cache; it must check for its existence, and then create the object if it is not found:

```
Mail mail = myCache.get("myMail");
if (mail == null) {
mail = readMailInformation();
myCache.put("myMail", mail) ;
}
...
```

  (In some cases application-specific caches can be designed in such a way that they lookup the required data from some data source that is transparent to the client.)
- **Object pools are used to restrict the number of instances of a certain class.** Often, a request requires an instance of a certain class, but this object does not (and should not) need to be recreated in every request. This is especially true in cases where object creation and initialization are expensive. Rather than accepting the performance hit, clients can request objects from a pool and then return them to the pool after finishing their use case.

```
PooledObject po = myPool.get();
...
// use the PooledObject
...
myPool.put(po);
```

- **A simple form of pool is to canonicalize an object.** This means that all different instances of an object are created during the program initialization phase and reused and referenced later on. The class java.lang.Boolean provides an example of an object that is canonicalized. There need to exist only two different Boolean objects, preferably accessible as constants. The same can be done with other objects with fixed sets of read-only internal state.
- **Do not fetch more data as you currently want to process.** For example, in your portlet you could provide a list of e-mails; the portlet displays the subject, date, sender, and other important information. When the user selects a particular e-mail, the body of the e-mail displays. The body is not needed before the specific item was selected from the portlet, so retrieving it any earlier would be a waste of execution time and memory resources. This pattern applies to many situations. The general rule is to only compute and retrieve those pieces of information that have direct significance for the current request and response.

J2EE

IBM WebSphere Application Server is the J2EE implementation upon which WebSphere Portal is built. Many of the performance considerations in the subsequent section apply due to the J2EE runtime context. The following sections describe performance items that are only applicable to IBM WebSphere Application Server. The items listed below are described here at a high level, and will be explained in more detail later in this article. For a more general discussion see Resources.

J2EE Standard

The J2EE standard specification contains a number of items with performance implications:

- init methods, available for many J2EE resources -- and also for portlets -- should be used to pre-calculate everything that will be used later on and that will not change. For example, JNDI lookups for common resources such as a data source should be performed only once at initialization time. Also, reading data from certain read-only files should be done just once during initialization of a portlet. You can scan your portlet service methods for any code that is executed in the same manner on every request and move it to the init method to reduce the run time cost for the service methods.

Visit the *New* www.SYS-CON.com Website Today!

The World's Leading *i*-Technology News and Information Source

24/7

JUMP TO THE LEADING i-TECHNOLOGY WEBSITES:

IT Solutions Guide
Information Storage+Security Journal
JDJ
Web Services Journal
.NET Developer's Journal
LinuxWorld Magazine
Linux Business News
Eclipse Developer's Journal
MX Developer's Journal
ColdFusion Developer's Journal
XML Journal
Wireless Business & Technology
Symbian Developer's Journal
WebSphere Journal
WLDJ
PowerBuilder Developer's Journal

FREE NEWSLETTERS
Stay ahead of the i-Technology curve with E-mail updates on what's happening in your industry

SYS-CON.TV
Watch video of breaking news, interviews with industry leaders, and how-to tutorials

BLOG-N-PLAY!
Read web logs from the movers and shakers or create your own blog to be read by millions

WEBCAST
Streaming video on today's i-Technology news, events, and webinars

EDUCATION
The world's leading online i-Technology university

RESEARCH
i-Technology data "and" analysis for business decision-makers

MAGAZINES
View the current issue and past archives of your favorite i-Technology journal

INTERNATIONAL SITES
Get all the news and information happening in other countries worldwide

- EJB components and sessions are important and powerful concepts within J2EE, but either of these can have severe performance implications if not used wisely. For example, applications should not put too much data into sessions to reduce the memory footprint of the server and to make session persistence easier and faster. Regarding EJB components, you should become familiar with the different persistence types associated with remote and local invocation, and so on. Some features available for EJBs come with a high performance penalty.

WebSphere Application Server

WebSphere Application Server products provide features to assist developers and architects in designing high-performance systems. (See the Information Centers for WebSphere Application Server and WebSphere Business Integration Server Foundation in Resources.

- As mentioned earlier, database connections are expensive to create. As defined in the J2EE standard, application servers can provide a pooling facility so the connections do not need to be recreated with every incoming request. WebSphere Application Server provides such a pooling facility together with some additional performance helpers, like a statement cache for frequently executed SQL statements. However, failing to return the connection immediately after completing the database interaction results in making the connection unavailable to other requests for significant periods of time. Using the WebSphere Application Server administrative console, you can control connection pools as properties of data sources to a JDBC database, and can define, for example, a minimum and maximum number of connections for the pool.

  For example, database connections are expensive to create. In this case, it is possible to make use of JDBC connection pooling and leverage the prepared statement cache as provided by WebSphere Application Server:

```
. . .
public class IDontCare extends GenericPortlet {

   private javax.sql.DataSource ds;

   public void init() throws javax.portlet.
     PortletException {

      try {
         Hashtable env = new Hashtable();
         env.put( Context.INITIAL_CONTEXT_
FACTORY,
            "com.ibm.ejs.ns.jndi.
CNInitialContextFactory" );
         Context ctx = new InitialContext( env
);
         ds = (javax.sql.DataSource)ctx.lookup(
          "jdbc/MYSHOES" );
         ctx.close();
      } catch (Exception any) {
         // handle exceptions here
         . . .
      }
   }

   . . .
   public void processAction (
      ActionRequest request,
      ActionResponse response
   ) throws PortletException, IOException {
      . . .
      try {
         Connection con = null;
         ResultSet rs = null;
         PreparedStatement pStmt = null;
         con = ds.getConnection ( dbuser,
           dbpasswd );
         pStmt = con.prepareStatement(
            "select * from myscheme.
              size_of_shoes");
         rs = pStmt.executeQuery();
         . . .

         // release the resource when it is
          no longer used
         if (pStmt != null) pStmt.close();
         if (con   != null) con.close();
      } catch (Exception any) {
         // handle exception here
         . . .
      }
   }
}
```

- WebSphere Application Server also supports the general concept of object pools, with each pool consisting of a pool manager providing access to pools for different class types. Such pools can be queried for an instance of the class type, as given in the previous example about pools.
- WebSphere Application Server also provides a "general purpose" cache. In the admin console, you can define cache instances that your applications can use to store, retrieve, and share data. Unlike the default shared dynamic cache, which is used by the portal to cache objects, a cache instance is only accessible to those applications that know its JNDI name. The DistributedMap interface is the programming interface that your applications work with, enabling the application to get and put objects into the cache instance, as well as invalidate them.

If portlets make use of a caching implementation, they should look up or instantiate a cache instance in

Reach Over 100,000

Enterprise Development Managers & Decision Makers with...

Offering leading software, services, and hardware vendors an opportunity to ***speak to over 100,000 purchasing decision makers*** *about their products, the enterprise IT marketplace, and emerging trends critical to developers, programmers, and IT management*

Don't Miss Your Opportunity to Be a Part of the Next Issue!

Get Listed as a Top 20* Solutions Provider

For Advertising Details Call 201 802-3021 Today!

*ONLY 20 ADVERTISERS WILL BE DISPLAYED. FIRST COME FIRST SERVE.

SYS-CON MEDIA

The World's Leading i-Technology Publisher

their initialization phase and keep a reference to that cache so that cache entries have a lifetime that is potentially longer than a single request. While processing the action and the render phase of a portlet, entries can be put into and retrieved from a cache. The portlet implementation needs to make sure that there is proper backend access and cache update handling if the cache does not return the data when queried with a certain key.

Also, be aware that scoping of keys into the cache (for example, per user session) might be required for proper function of the intended design. A cache is typically a self-managed unit that can evict or invalidate entries dependent on the cache implementation. Note that, for the same reason, a cache is not suitable to communicate information back and forth between several pieces of code. A cache should also maintain a reasonable upper size limit to avoid memory over-utilization in custom code.

Portal APIs

WebSphere Portal supports two different portlet APIs:

- IBM Portlet API, which extends servlets
- JSR 168 Portlet API, which is defined by the Java Community Process (JCP)

In this article, we focus on the JSR 168 Portlet API. WebSphere Portal provides a variety of interfaces for integrating your portlets into the WebSphere Portal environment. As such, portlets should be carefully designed to take advantage of portal features. Be sure to apply best practices (such as those listed in *Resources*) so that your application of WebSphere Portal APIs is appropriate.

Common Implementation Considerations

In this section, we look at performance topics that are relevant for theme and skin programming, as well as portlet development.

JSPs

JavaServer Pages (JSPs) are one of the cornerstones of portlet programming. In most portlets, using Model View Controller (MVC) terminology, JSPs function as the view component. JSPs are composed of a mixtures of HTML (or other markup languages) and Java code; their processing output is also a markup language, in most cases HTML. In its purest form, JSPs do not contain any Java code, but only custom tags that are called to perform non-HTML operations. (Conversely, it is also possible to have virtually no HTML content in a JSP file.)

- Upon the very first access to a JSP file, the file is parsed and translated into a regular Java servlet source file, which is then compiled into byte code. Hence, the very first request to a JSP is typically slow due to the two subsequent conversions (from JSP to Java source to byte code), but then it works like any other servlet for all future requests.

  This is different from another approach for generating HTML content: XML and XSLT. In this case, upon every request the XML has to be parsed and style sheet transformations have to be applied. Only a good approach of caching the results and not re-running the transformations upon every request can save the performance here. Hence, from a performance point of view, JSP should be preferred over XML/XSLT. Furthermore, the portal infrastructure is optimized around JSPs, enabling easy expansion into other markups, languages, and browser support.
- Application servers execute JSPs similarly to how they execute regular servlets. Nonetheless, servlets resulting from JSP compilation contain generated code, which is, in general, less optimized for performance than handcrafted code. If performance is very important for a certain JSP and you cannot achieve your goals with generated code, consider writing the markup into the output stream by yourself.
- Java code fragments in JSPs are called scriptlets. Since JSPs are converted to Java source code anyway, there is no real performance penalty associated with using scriptlets. Some optimizations in the latest version of WebSphere Application Server apply to JSP files in cases where a JSP file does not contain any scriptlets. In general, you should not put scriptlet code into your JSPs, and instead use tags for those tasks.
- JSPs can include other JSPs. That means that a single JSP does not have to answer the complete request; you can split the response into multiple JSPs and include others from a parent JSP. There are two different forms of inclusion, static and dynamic:
  - Static JSP includes are resolved at compile time. The JSP compiler includes the referenced file instead of the include statement. This option is generally very fast and adds no run time overhead at all.

```
<%@ include file="include2.jsp" %>
```

  - Dynamic JSP includes are resolved at run time, which is not an inexpensive undertaking. Resolving the correct JSP to which to dispatch is quite expensive in terms of garbage creation and execution time. For example (inside a JSP):

```
<jsp:include page="include2.jsp" flush="true" %>
```

  - Dynamic inclusion in JSPs is similar to using javax.servlet.RequestDispatcher when including other files from servlet code. Therefore, wherever possible, you should use static includes. Dynamic includes offer the highest flexibility, but come with a significant performance overhead if used too often.

EJB Usage

Enterprise JavaBeans (EJB) define a component-based architecture for building scalable, transactional,

distributed, and multi-user business applications. EJB components are designed to encapsulate the business logic while hiding all complexity behind the bean and built-in EJB container services.

The support for the variety of functions frequently required by enterprise applications does not come for free, as there is a certain amount of performance overhead that needs to be taken into account when using EJBs.

- A portlet can obtain EJB references through JNDI lookups, which tend to be expensive with regard to performance. For example, if a portlet does not cache the reference to the EJB home interface somewhere, then every logical call to the EJB requires two remote calls: one to the naming service, and one to the actual object. To rectify this situation, use caching techniques to reduce or eliminate repeated lookups of EJB home references.
- EJB components expose remote and local interfaces. EJBs that are location-independent use a remote interface. Method parameters and return values are serialized over RMI-IIOP and returned by value. Remote methods must be designed to satisfy data needs according to the usage pattern of the API. Use a granularity of methods and data types in the API that fits well with the use cases of the interface to minimize serialization costs.
- Minimize the number of remote calls to reduce the overhead imposed by remote calls in your code path. Use session beans acting as remote facades to wrap complex interactions and to reduce remote calls between portlets and domain objects. A portlet accessing a remote entity bean directly usually results in several remote method calls. If you use entity beans in this context, avoid giving them a remote interface. Instead, session beans acting as facades can access entity beans via their local interfaces, gather data from them, and then return this information to the calling application.

  The concept of local interfaces works when the calling client (such as a session facade) shares the same container as the called EJB. The use of local interfaces reduces inter-process communication costs by eliminating the overhead of a distributed object protocol. Local calls do not go through the communication layer and any objects can be passed by reference.
- Transaction management supported by the EJB container can also affect performance. After developing an EJB, the programmer must set deployment descriptors that define characteristics, such as transaction support and isolation levels for the EJB. Set the transaction type to NotSupported if no transaction is required.
- The transaction isolation level is the degree to which the underlying database exposes changed but uncommitted data to other transactions. For the best performance, use a liberal isolation level. Letting uncommitted data be seen by other transactions, however, can result in unexpected side effects, such as clashing updates and inconsistent reads.
- See the IBM white paper WebSphere Application Server Development Best Practices for Performance and Scalability (www-306.ibm.com/software/webservers/appserv/ws_bestpractices.pdf) and the IBM Redbook IBM WebSphere V5.1 Performance, Scalability, and High Availability WebSphere Handbook Series (www.redbooks.ibm.com/abstracts/sg246198.html?Open) for additional recommendations, as well as the justification for each recommendation.

Markup Size

Markup size refers to the number of bytes to be transferred from the portal server to the client for a completely rendered portal page. From a portal server point of view, the most important part is the size of the HTML page containing the resulting markup. Other files, like stylesheets, images, or JavaScript, must be transferred to the client as well. Since the static files are typically stored outside the portal system on HTTP servers or proxy caches, we will concentrate here on the "real" HTML markup size.

Why do we bother at all about markup size? Within company intranets, there are probably lesser issues with network bandwidth, but if users are connected via modem or other low-bandwidth network connections, lengthy download times for large HTML responses can be very annoying.

Here is a short calculation. Let's assume that a server or cluster is processing up to 100 requests per second. The average HTML page size should be 100 KB, which may seem like a lot, but can easily be reached if you have a complex theme and several portlets on the page. For the server, this means that it has to put about 10 MB/sec on the wire (100 KB * 100 pages/sec). This is about the maximum traffic a 100 MB network can handle. (You cannot expect an Ethernet to support 100% of its potential capacity of 100 MB/sec and the inbound traffic is typically not negligible. For users connected to the portal via 56K modem, the download time for one page would be in the area of 15 seconds!)

How large is too large? This is difficult to answer in general. However, more than 100 KB per HTML page is probably too much. Also, keep in mind that smaller devices have restrictions on the markup size they can handle per request.

The main contributors to markup size are the theme and the portlet output. Since all portal JSPs are customizable, you can influence how compact your markup is at the end. Here is what you can do to limit the size of your markup:

- **Use JSP comments instead of HTML comments inside your JSPs.** Comments of the form <%– ... –%> will be deleted by the JSP compiler, while comments of the form <!– ... –> are kept and transported over the wire.
- **Try to reduce the amount of white space, tabs and line breaks inside your JSP source files since these are retained by the JSP compiler.** This may reduce legibility of the code. It can be helpful to have code in development that is nicely laid out, but it is processed and stripped off its formatting with a tool before the JSP files are applied to your production environment.
- **Try to avoid sending the same information to the client several times.** For example, style definitions should go into separate CSS files. The same is true for JavaScript code. Furthermore, these separate files, as they typically do not change, can be cached in browser or proxy caches, reducing network traffic even further.
- **If your environment setup for it, you can also send compressed markup using HTTP compression to the client.** Refer to your Web server's and your clients' documentation for more information.

"While memory often is not a predominant performance issue for Java client software, it is a major concern for J2EE applications, mainly because enterprise applications are typically accessed by many users at the same time"

Logging, Tracing, and I/O

Logs usually end up on the hard disk at some point. From a performance perspective, anything interacting frequently with the disk presents a potentially expensive operation, so it is best to minimize the use of the Java I/O library in the production environment. Since the I/O is usually provided through the use of some native library layered beneath Java programming, there is some default overhead incurred. An operation like System.out.println synchronizes processing during file I/O, which significantly impacts performance.

In development and test mode, you may want all the logging and debugging to be active, since it could be essential for finding errors. When you deploy your application in a production environment, leaving extensive logging on is just not a viable option. It is a good practice to guard log statements so they are turned on during error and debugging situations only. Do that by using a final Boolean variable, which, when set to false, effectively tells the compiler to optimize out both the checking and execution of the logging code:

```
static final boolean LOGGING = false;
if (LOGGING) {...}
```

The Java language provides two types of streams: readers/writers and input/output:

- Readers and writers are high level interfaces to support unicode characters in I/O operations.
- Input/output streams provide data access mechanisms at a much lower level; at the byte level.

There is a performance overhead involved with readers/writers because they are intended for character streams, and they encode data to bytes under the covers. Instead, you should use input/output streams whenever you want to manipulate binary data.

To maximize I/O performance, you should buffer read and write operations. If you want to write large amounts of data from a portlet, it is usually a good approach to flush the buffered data partly rather than

"Keeping and maintaining data for a portlet that should have a lifetime longer than a single request is a typical portlet programming task"

flushing them completely in a single blow. On the other hand, do not flush the buffer too often.

Synchronization and Multi-Threading

The Java mechanism that is used to coordinate access to shared objects is called synchronization. The synchronized statement enbales only one thread to enter a code block at a time.

- During the lifetime of a portlet, the container sends service requests in different threads to a single portlet instance. Avoid synchronization within a portlet because it has a significant performance impact: synchronization reduces concurrency, since only one thread is allowed to run at a time in a synchronized block of code and all the concurrent threads are queued. There is also performance overhead caused by managing the monitors that the Java virtual machine uses to support synchronization. Besides the performance impact, there is also the possibility of deadlocks that can potentially freeze the single portlet or – even worse – the whole portal. Deadlock prevention is the programmer's responsibility, because the monitors do not support any deadlock resolution.
- In cases where synchronization is necessary, the synchronized code block should be minimized. It is crucial to accurately identify which code block truly needs to be synchronized and to synchronize as little as possible. If it is not small enough, you should analyze your code and refactor it in such a way that anything that could run asynchronously is located outside of the synchronized block.
- Some Java J2SE functionality indirectly uses synchronization. Java collection classes, like Vector or Hashtable are fully synchronized. Java programs pay the costs associated with thread synchronization even when they are used in a single-threaded environment. Newer collections introduced in Java 1.2, like ArrayList are not synchronized. This provides faster access to the data. In situations where you know you need thread safety, use a thread-safe view. Thread-safe views are wrapper classes that add functionality to synchronize the methods of the standard collection. The factory method of the Collections class returns a thread-safe collection that is backed by the instance of the specified collection type:

```
List list = Collections.sychronizedList(new
ArrayList());
```

- Another example of indirect synchronization is the Java I/O libraries. Minimize the use of Java I/O library methods (for example System.out. println())to reduce unnecessary performance overhead.
- Do not spawn unmanaged threads from portlets. Currently J2EE strongly discourages from trying to spawn new threads in the container. In fact, J2EE specification 6.2.1 programming restrictions states:
  "If application components contain the same functionality provided by J2EE system infrastructure, there are clashes and mis-management of the functionality. For example, ... to manage threads, ..."

A practical reason for not trying to spawn new threads is that new threads do not have full access to the J2EE context. Further, newly created unmanaged threads undermine the goal of WebSphere Portal to achieve a stable, optimized and scalable run time environment. Therefore, use the asynchronous bean feature (See WebSphere Application Server Enterprise V5 and Programming Model Extensions WebSphere Handbook Series: www.redbooks.ibm.com/abstracts/sg246932.html?Open) in WebSphere Application Server. An asynchronous bean is a Java object or enterprise bean that has the ability to submit code to be run on a separate thread – and asynchronously – using the J2EE context.

Portlets

The portlets programming model enables developers to create Web applications that can become part of an aggregated view of several of such applications in a client browser. In WebSphere Portal, such applications can not only coexist on a page (that is, the aggregated view), they are also able to communicate with each other while constructing the page. Thus, the implementation of a portlet can influence the overall perceived performance of a page. If a specific "critical" portlet resides on a page, then it may be worthwhile to invest some effort in optimizing this specifc portlet. You should also have a deeper look at portlets that reside on performance-critical pages.

Backend Access

Totally self-dependent portlets are a rare case in real world portals. Such portlets are typically used as an addition to or helper tool for a Web site; for example, a pocket calculator. Such portlets can be optimized in their local code execution path only and should not put too much burden on a running portal system.

A more typical usage of portlets is to offer application functionality that requires access to other data sources or transaction systems that also require execution resources apart from the system where the portlets are originally running. Data is potentially retrieved from and stored on other backend systems over a network. The transaction length, isolation level, and data locking that may occur on the backend system needs to be considered in the overall system design.

Be aware that a single portlet is probably not the only client of a backend system. In fact, there are many clients to such a system in the real world -- and even a single portlet can access the same backend system several times simultaneously. A portlet might execute its code in separate application server threads for different user requests. Therefore, it is worth investigating the access patterns and how transactions and locks acquired by a portlet or other clients can potentially influence the average response time of such a backend system.

If a portlet requires intensive backend system access during the action or render phase, the response time (for finishing these phases) more and more depends on the responsiveness of the backend system. (Waiting for responses from outside the portal server to satisfy incoming requests will introduce latency that cannot be recovered by optimizing the execution path of the portlet code.) A good design for communicating with the backend and an understanding of transactional behavior is often more promising.

To avoid times when a portlet – and the page on which it appears – fails to respond because of a collapsed backend system, it can be a good practice to incorporate timeouts in your code; be aware, though, that managing and keeping track of time stamps introduces some processing overhead. If the parallel portlet rendering feature in WebSphere Portal is used (discussed later), time outs are configurable for the parallel render threads.

It is also a good programming practice to reduce the interaction and data traffic to such external backend systems, where possible. To achieve this, portlets can cache information if the freshness criteria of the information permits you to do so. This may reduce the roundtrips for fetching the same data over and over again for each incoming request to WebSphere Portal. It also helps to lower the load on the backend system, since it is not involved in providing the same information so many times. Also, if the data does not need to be transferred over the network, the portlet can potentially be rendered more quickly.

Another way of avoiding roundtrips to the backend system can be to retrieve more data than is actually required to fulfill the current request, but that is known to be required in a request that would otherwise follow. With this approach, however, we still advice against

general prefetching if it is not known what prefetched data is actually needed in subsequent requests. A good understanding of typical user interactions with the portlet application is needed for a proper design of this nature. Keep in mind that retrieving data in advance has an impact on memory usage of the portal JVM. (See Code design for performance and scalability: www-128.ibm.com/developerworks/websphere/techjournal/0508_dzierzon/0508_dzierzon.html#cpi.) Such design approaches might require an interface change of the backend system, but it could save considerable processing time that might make such a change worth it.

For caching, WebSphere Application Server offers a dynamic caching feature with its DistributedMap interface to portlets. (See WebSphere Application Server 5.1 Information Center: http://publib.boulder.ibm.com/infocenter/wasinfo/v5r1/index.jsp for more information.)

The Session and Other Data Stores

Keeping and maintaining data for a portlet that should have a lifetime longer than a single request is a typical portlet programming task. Using PortletSession is often the first approach that is considered. While PortletSession is convenient to use from a programmer's perspective, it requires resources for managing sessions from an application server perspective. The problem can get worse if the session contains more and more data and thus requires more memory.

If the session is configured to be persistently stored in a database, or is configured for memory-to-memory replication (that is, WebSphere Portal is configured for failover in a clustered environment), then the session may become serialized any time its contents changes.

The time it takes to serialize and deserialize session data when they are being written to a remote copy can become considerably large. In rare cases, some objects stored in a session can be marked transient. This will reduce the serialized size of a session, but will not change the in-memory size, which also has an impact on how efficiently an application server can handle sessions.

(Parts Two and Three of this article will appear in future issues of WebSphere Journal.*)*

Resources

A number of resources to this article can be found online. You can visit IBM Developer Works to find the original version of this article along with the following resources.

- JSR 168: Portlet Specification: www.jcp.org/en/jsr/detail?id=168
- Best practices: Developing portlets using JSR 168 and WebSphere Portal V5.02: www-128.ibm.com/developerworks/websphere/library/techarticles/0403_hepper/0403_hepper.html
- Meet the Experts: Marshall Lamb on WebSphere Portal Programming: www-128.ibm.com/developerworks/websphere/library/techarticles/0401_lamb/lamb.html
- Portlet Development Best Practices and Coding Guidelines: www-128.ibm.com/developerworks/websphere/zones/portal/portlet/portletcodingguidelines.html
- WebSphere Application Server Development Best Practices for Performance and Scalability: www-306.ibm.com/software/webservers/appserv/ws_bestpractices.pdf
- IBM WebSphere Portal for Multiplatforms Version 5.0 Performance Tuning Guide: www-1.ibm.com/support/docview.wss?rs=688&context=SSHRKX&q1=tuning&uid=swg27004492&loc=en_US&cs=utf-8&lang=en+en
- WebSphere Application Server 5.1 Information Center: http://publib.boulder.ibm.com/infocenter/wasinfo/v5r1/index.jsp
- WebSphere Business Integration Server Foundation 5.1 Information Center: http://publib.boulder.ibm.com/infocenter/wasinfo/v5r1/index.jsp
- WebSphere Portal 5.1 Information Center: http://publib.boulder.ibm.com/infocenter/wp51help/index.jsp
- Redbook: IBM WebSphere V5.1 Performance, Scalability, and High Availability WebSphere Handbook Series: www.redbooks.ibm.com/abstracts/sg246198.html?Open
- Redbook: WebSphere Application Server Enterprise V5 and Programming Model Extensions WebSphere Handbook Series: www.redbooks.ibm.com/abstracts/sg246932.html?Open
- *Java Performance Tuning* by Jack Shirazi, O'Reilly & Associates (2003), ISBN: 0596003773
- *Java Performance and Scalability* by Dov Bulka, Addison Wesley (2000), ISBN: 0201704293
- *Java Platform Performance: Strategies & Tactics* by Steve Wilson, Addison-Wesley (2000), ISBN: 0201709694
- *Performance Analysis for Java Websites* by Stacy Joines, Addison Wesley (2002), ISBN: 0201844540
- IBM Rational Application Developer: www-306.ibm.com/software/awdtools/developer/application/
- Eclipse Test & Performance Tools Platform: www.eclipse.org/tptp/
- JVMPI documentation: http://java.sun.com/j2se/1.4.2/docs/guide/jvmpi/jvmpi.html
- IBM Java Diagnostics Guides: www-128.ibm.com/developerworks/java/jdk/diagnosis/
- IBM WebSphere Developer Technical Journal: Develop high performance Web sites with both static and dynamic content using WebSphere Portal V5.1: www-128.ibm.com/developerworks/websphere/techjournal/0506_liesche/0506_liesche.html

Get Products and Technologies

- IBM Pattern Modeling and Analysis Tool for Java Garbage Collector: www.alphaworks.ibm.com/tech/pmat
- Diagnostic Tool for Java Garbage Collector: www.alphaworks.ibm.com/tech/gcdiag
- heapRoots: alphaworks.ibm.com/tech/heaproots

The IP Battles Continue

BY ROGER STRUKHOFF

The battles over intellectual property continue. Google is the company-of-the-day in the headlines, with its plan to make available excerpts and whole text from several university libraries under legal attack from the New York-based Author's Guild, which represents 8,000 authors.

This is a copyright issue. Earlier headlines this year, involving all major U.S. technology providers, on the other hand, involved patents: how they're granted and adjudicated in the U.S., and how they should be granted and adjudicated within the European Union.

Meanwhile, Google and Microsoft are tussling over the hiring of a key employee – about how much of what he knows in the burgeoning area of search belongs to Microsoft, which is the latest issue to threaten Microsoft's long dominance on the desktop – and whether he can be held to the non-compete he signed.

US business people in the entertainment, publishing, software, and clothing industries will flat out tell you they won't do business in China until that country cleans up its act as a global leader in pirated IP. This as China moves rapidly toward full integration in the world economy and prepares to host the 2008 Summer Olympics.

We're clearly not at "the end of history," as one unfortunate school of thought maintains, nor are we in a "clash of civilizations," as another book and its acolytes claim. One could certainly think either of these thoughts by superficially examining current events.

Certainly capitalism is in the final rounds of a resounding victory over communism on a global scale, as the former theory posits. And there are certainly religious and political leaders throughout the world making points with their constituents by painting a simplistic portrait of one divine over another.

Yet although communism appears to be gasping its last breath, it is capitalism, not religion, that remains under attack by various individuals and groups throughout the world.

Dig down into the debate over globalization, for example, and one finds an elite group that is mainly disenchanted with the efforts of the World Bank and the IMF, but remains solidly capitalist in its worldview. This view dramatically differs from that of the anarchists and nihilists who demonstrate at global economic summits; these people object to capitalism, period, and find the two institutions mentioned above the most convenient targets.

Meanwhile, people usually characterized by Americans as terrorists and insurgents use religion to justify their agenda, which beyond removing U.S. troops from certain places is also focused on removing what they see as corrupting capitalist inroads into their societies.

Capitalism is the Big Story right now. And capitalism can't function – and so can't be reformed – in a world that lacks intellectual property protection. Just as the semi-mythical Wild West of the American 19th century was not truly "won" until a semblance of a legal code was instituted and enforced, there will be little advance in the average wealth of the average citizen of the world until a semblance of a fair intellectual property code is instituted and enforced throughout the world.

Roger Strukhoff, editor-in-chief of ***WebSphere Journal***, is West Coast Bureau Chief for the SYS-CON News Desk, and President of www.wdva.com. He spent 15 years with Miller Freeman Publications and The International Data Group (IDG), then cofounded CoverOne Media, a custom publishing agency that he sold in 2004. His work has won awards from the American Business Media, Western Press Association, Illinois Press Association, and the Magazine Publishers' Association. Read his blog at www.rssblog.linuxworld.com. Contact him at roger@sys-con.com.

YOUR TESTING SUCKS.

1. Writing test code isn't testing – it's coding.
2. Your developers aren't making testable software.
3. It's impossible to get the business side involved in quality.
4. Your existing testing tools have become shelfware.
5. Your customers have become the primary testers.

LISA from iTKO.

Because everyone should own quality.

iTKO Try a free eval of LISA 2.5 at **www.itko.com**

We understand.

You've probably invested considerable time and money trying to automate your testing. But it just didn't work.

You were sold a tool that let you point and click to test on Day 1, and on Day 2 you were maintaining a test script - inserting bugs into your tests at the same rate as you inserted bugs into the code during development.

The tools were too hard for the QA teams to use, they required intense training and developer knowledge, and the seats were more expensive than a presidential fundraiser.

It's OK. We've seen this scenario at every customer we talk to.

LISA is different.

LISA is a comprehensive, no-code automated testing solution ideal for distributed applications and teams.

Using "inline testing" technology, LISA allows even non-developers to instrument and test web services, J2EE apps, databases and websites.

LISA automates test design, staging and monitoring for unit, regression, load and performance tests. And since LISA is pure Java, she's equally stylish on Linux, OS X, Unix or Win clients.

But don't take this ad's word for it. Try LISA for free and talk to the experts at iTKO. We'll show you how everyone can own quality.

IBM WEBSPHERE PRESENTS

YOU VS APPLICATION INTEGRATION COMPLICATION

FEATURING: **POWER TO CONNECT ANY APPLICATION ON ANY PLATFORM**
UNMATCHED EXPERIENCE BUILDING COMPOSITE APPLICATIONS
INCREASED EFFICIENCY, ULTIMATE FLEXIBILITY

PLUS HEAVYWEIGHT GUESTS: **OPEN STANDARDS ★ EASE ★ AFFORDABILITY**

IBM MIDDLEWARE.
POWERFUL. PROVEN.

RE-USE YOUR EXISTING I.T. INVESTMENTS
FIGHT BACK AT
WWW.IBM.COM/MIDDLEWARE/CONNECT