

WebSphere®

DEVELOPER'S JOURNAL

The World's Leading Independent
WebSphere Developer Resource

WebSphereDevelopersJournal.com

SEPTEMBER VOLUME 2 ISSUE 9

**REGISTER FOR
WEB SERVICES EDGE
TODAY!**
CALL 201-802-3058
www.sys-con.com/edge
Register before September 26th and
**SAVE
\$100**

FROM THE EDITOR

Lots of Exciting Happenings
in the World of WebSphere

BY JACK MARTIN • PAGE 4

PERFORMANCE

Profiling in
WebSphere Studio 5.0, Part 2

BY ANDREW SONDERGOTH • PAGE 16

FINAL THOUGHTS

Why Java Needs
Rapid Application Development

BY ZACK URLOCKER • PAGE 50

NEWS
PAGE 48

DISPLAY UNTIL NOVEMBER 30, 2003

\$8.99US \$9.99CAN



0 09281 03422 3

**SYS-CON
MEDIA**



A Powerful, Easy-to-Use Logging System

Using log4j from a WebSphere-based application

BY BIBHAS BHATTACHARYA

PAGE 12

J-Unit Testing in WebSphere Studio 5

Stable code is a good thing

BY KULVIR SINGH BHOGAL

PAGE 22

EAI: Demystifying Integration

Quickly link disparate systems

BY TROY HOLMES AND JAY HOTALING

PAGE 26

Skin Deep

A skin is more than just a matter of appearances

BY MARCEL HEIJMANS

PAGE 32

Dealing with Large Database Result Sets

The ValueListHandler design pattern improves
J2EE application performance

BY LLOYD HAGEMO
AND RAVI KALIDINDI

PAGE 36

The Trend Toward Adaptive Software

Making role-based, composite applications a reality

BY JOE LICHTENBERG

PAGE 46

DIRIG SOFTWARE

WWW.EBIZPERFORM.COM

WILY TECHNOLOGY

WWW.WILYTECH.COM

Lots of Exciting Happenings in the World of WebSphere

BY JACK MARTIN

It's been a busy month in the world of WebSphere, with the advent of some very exciting new functions in WebSphere Studio and WebSphere Application Server. The latest version of WebSphere Studio (v5.1) takes a huge step toward helping people build Web applications in Java – without needing to be a Java expert. This version contains several rapid application development features, such as a visual editor for Java and Enterprise JavaBeans, and a wizard for generating Web services. These features enable a whole new class of people to start customizing WebSphere.

WebSphere Studio now enables nonprogrammers to create Web services; soon everyone will be creating Web services. A wizard enables novices to connect a Web page to an application that delivers regular updates of the information of their choice. All the user needs to do is point the wizard to the application, which is registered as a Web service on IBM's UDDI registry, and let the tool generate the necessary code.

UDDI can automatically discover and integrate with services on the Web. Although not widely used today, many companies are experimenting with this technology within corporate intranets, and it should go mainstream within the next couple of years.

IBM now offers a Ready for IBM WebSphere Studio software validation program that opens up WebSphere to practically any third-party vendor, allowing almost any type of add-on functionality. You've got to be a member of PartnerWorld for Developers to participate, but if you're not, it's easy to join. The validation proves that your product works with the WebSphere Studio Workbench and/or the WebSphere Studio Family by correctly using their APIs.

The validation criteria focus on eight key areas of functionality: the ability to launch developer tools from the Workbench; to show and create developer tool views within the Workbench; to manage developer tool attributes and preferences; to connect developer tools to Workbench views and editors; to exchange data with and extend the Workbench Java development tools; to exchange data with and extend WebSphere Studio product components; to create developer tool help information; and to package and install developer tools.



The program is accessed through the Eclipse IDE, which makes adding new functionality to WebSphere Studio a simple process, as developers can plug in their favorite tools from any vendor supporting the open-source platform.

Meanwhile WebSphere Application Server v5.0.2 is more than just a standard fixpack; it's a mini-release. Starting

with support for the Java Development Kit (JDK) 1.4 client container, there are improvements in Web services interoperability and security, performance and application tuning, platform support, systems management, and database integration.

Now there is also support for all of the Web services standards needed for describing (WSDL 1.1) and deploying (JSR 109) applications or services on a network in a consistent way so that they can be discovered (UDDI 1.0 and 2.0) and invoked (SOAP 1.1 and SOAP 1.1 with Attachments, SAAJ 1.1, WSIF, JSR 101), in a more secure (WS-Security, XML Signature, XML Encryption) and reliable manner (SOAP over JMS).

In addition to high-performing support for all of the newly released Web services standards, WebSphere Application Server v5.0.2 is the first production-level application server to support the June 2003 Board Approved Draft of the WS-I Basic Profile 1.0. Developers building Web services applications with WebSphere Application Server v5.0.2 will get a head start on interoperating across heterogeneous environments and enterprise boundaries.

Database connectivity has been expanded with a new, integrated DB2 Correlator with WebSphere Trace, which provides improved serviceability and better debugging when using WAS and DB2 together. It also supports the new JDBC Type 2 driver support for DB2 v8 with continued support for Sybase 12.5 and Oracle 9 R2.

And now, with virtualized backup clusters available, you can automatically configure your system to set up a backup cluster of servers in case your primary cluster fails – without having to write any code. If your primary cluster goes down, the workload is automatically sent to another cluster elsewhere in your network. What more could anyone ask?



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

ADVISORY BOARD

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

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

WRITERS IN THIS ISSUE

Bibhas Bhattacharya, Kulvir Bhogal, Lloyd Hagemo, Marcel Heijmans, Troy Holmes, Jay Hotelling, Ravi Kalidindi, Joe Lichtenberg, Jack Martin, Andrew Sondgeroth, Zack Ullocker

SUBSCRIPTIONS

For subscriptions and requests for bulk orders, please send your letters to Subscription Department. 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
TECHNICAL DIRECTOR ALAN WILLIAMSON
SENIOR VP, SALES & MARKETING CARMEN GONZALEZ
VP, INFORMATION SYSTEMS ROBERT DIAMOND
VP, SALES & MARKETING MILES SILVERMAN
PRODUCTION CONSULTANT JIM MORGAN
FINANCIAL ANALYST JOAN LAROSE
ACCOUNTS RECEIVABLE KERRI VON ACHEN
ACCOUNTS PAYABLE BETTY WHITE
ADVERTISING DIRECTOR ROBYN FORMA
DIRECTOR OF SALES AND MARKETING MEGAN MUSSA
ADVERTISING SALES MANAGER ALISA CATALANO
ASSOCIATE SALES MANAGERS CARRIE GEBERT
KRISTIN KUHNLE
CONFERENCE MANAGER MICHAEL LYNCH
SALES EXECUTIVE, EXHIBITS JAMES DONOVAN
ART DIRECTOR ALEX BOTERO
ASSOCIATE ART DIRECTORS LOUIS F. CUFFARI
RICHARD SILVERBERG
TAMI BEATTY
WEB DESIGNERS STEPHEN KILMURRAY
CHRISTOPHER CROCE
CONTENT EDITOR LIN GOETZ
CIRCULATION SERVICE NIKI PANAGOPULOS
COORDINATORS SHELLA DICKERSON
EDNA EARLE RUSSELL
CUSTOMER RELATIONS MANAGER RACHEL MCGOURAN

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® Developer's Journal (ISSN# 1535-6914)
is published monthly (12 times a year).
Postmaster send address changes to:
WebSphere Developer's Journal, SYS-CON Publications, Inc.
135 Chestnut Ridge Road, Montvale, NJ 07645

WORLDWIDE NEWSSTAND DISTRIBUTION
CURTIS CIRCULATION COMPANY, NEW MILFORD, NJ

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

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

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

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

SYS-CON
MEDIA

PROLIFICS

WWW.PROLIFICS.COM/WEBSERVICES

MQSOFTWARE

WWW.MQSOFTWARE.COM

MQSOFTWARE

WWW.MQSOFTWARE.COM



The Demand for On Demand

A conversation with IBM's Peter Fordyce

— INTERVIEWED BY JACK MARTIN —

WebSphere Developer's Journal Editor-in-Chief

Jack Martin recently spoke with IBM's Peter Fordyce, Worldwide WebSphere Foundation and Tools Sales Manager about how he sees his group's role in the WebSphere community – and what customers are looking for. In this exclusive interview Fordyce discusses the push for on-demand business, the value of legacy code, and the future of WebSphere.

WSDJ: What do you do on a typical day?

Peter Fordyce: I interface with sales folks all over the world, with marketing colleagues formulating our messages and collateral as well, and with the development staff in terms of appropriate features and functions that we need in the product as we go forward.

WSDJ: What kinds of customers do you usually see – big customers, little customers, all kinds of customers?

Fordyce: Typically I get involved through the field sales folks, who either have a strategic initiative with a customer or a large sales opportunity and they want to put the best skills forward. Or it's a challenging situation – maybe a competitive situation – and the rep is seeking unique assistance. Most situations are with large customers, although our small and medium business team has been calling more this year. It's almost always a situation that is either challenging or very important to the local sales teams.

WSDJ: In other words, you're one of the people on the front lines of WebSphere day to day?

Fordyce: In a word, yes. In this position I have the unique opportunity of touching many sales reps and many customers with their most leading-edge business challenges. No one is bashful about expressing what they would like changed or

fixed. I think that I have the best job in IBM, seeing so many different customer situations all over the world. It's amazing to me how businesses are really facing similar problems no matter their culture or geography. What I gain then, is insight into the nuances between different countries, whether it's in the U.S., Belgium, or China.

WSDJ: Since you have a global perspective, where do you see the most activity around the world today? Is it here in the U.S., or elsewhere?

Fordyce: In terms of markets, the U.S. and the European markets are about equal. Those are huge markets with tremendous business activity. Then we look out and see the activity that is going on in the Far East, and there is a tremendous emerging market growing twice as fast as the rest of the world.

WSDJ: I heard that you are doing some significant work in the high-end application server and tools areas right now. Tell me about that.

Fordyce: Our core mission is supporting the multitude of pieces to make our customers more competitive or ready to offer "on-demand" services, as expressed by our chairman [Sam Palmisano] last October. You can summarize the changes needed to become an on-demand business across three areas – people, process, and technology. Each of these components must evolve in concert with the others to achieve the focus that allows for quick change in a robust manner.

Or we could classify the core values necessary in a business to achieving this focus [so it is] built on a solid foundation that is responsive, variable, and resilient. So when I think about the high-end app server, for instance, it is but one of many pieces in the technology infrastructure that allows this evolution. Make no mistake: the process enabled by the technology and utilized by the people also has to change. Thus there are many portions of our software portfolio that when fully deployed complete the technology vision of becoming "on demand."

However, when I think about technology I think of a core technology operating environment; I believe that to be squarely on top of our high-end application server.

The focus of WebSphere Application Server is handling all the repetitive infrastructure services and allowing my customers to concentrate on business development. The more easily I can provide a set of open application and integration services, the faster the business deploying WebSphere can take advantage of new revenue opportunities in their market.

Key then is providing for easy application development. In this world there is no new application development. Application projects invariably need to borrow, interact, or just plain reuse other applications. It's all about providing the integration services for these new project applications that make the developer's and administrator's jobs easier. This includes open connectivity technologies, but also the ability to control the application flow; provide a compensation engine for work that might need to be rolled back; scheduling mechanisms; and automated international services to handle different currencies, formats, time zones, and so forth.

Let's think about an example. If you are a bank and you've been through an acquisition, you're likely to have many different customer systems. If you are building a new application that updates the name and address fields of the customer file, you as a programmer would need to interact with multiple systems. Wouldn't it be nice if the software platform did that work for the programmer? Simply writing a method that updates names and addresses – with the infrastructure taking upon itself the task of coordinating with the different systems, ensuring the updates are indeed done, or backing the transaction out, so to speak, if there is a problem.

That's a lot of low-level code that does not necessarily produce better business results for our clients. Quickly delivering applications to clients in order to service their customers is what we are about. That's what the WebSphere Application Server and associated Studio tool set are all about. I think of the WebSphere Application Server and WebSphere Studio tool set then as providing a set of "build to integrate services" that the developer and administrator can leverage for faster development and deployment.

So if we are about integration, that means connecting different systems with one another. We have always had a base of connectors for CICS, IMS, and SAP or anything you could run over WebSphere MQ or through a JDBC database environment. However, in June we made generally available in excess of 300 connectors on top of the WebSphere Application Server, supplying integration to almost any type of system deployed in our customers' shops today. This is a very exciting time for us. I'm quite proud of the portfolio of servers and tools that allow us to attack highly demanding environments.

WSDJ: I was reading something about a week ago and would like to get your view on it. The average enterprise right now has something in the neighborhood of 13 million lines of COBOL code – it's legacy. That sounds like it plays into a lot of what you are talking about.

Fordyce: It plays into it tremendously. As I said at the beginning, customers face similar challenges the world over. One challenge is to leverage the significant investment they have built up – in many cases over decades of application generation. There are a lot of ways to look at it. You can say that in order to replace the COBOL applications in a typical large enterprise it would take on average a staff of 40 developers 75 years. Or you could look at the average cost of a line of code and determine the average large enterprise has a \$300 million asset. This means that our customers have a significant set of assets that are much like the old factories of the past. The difference today is we don't have to rebuild those factories, thanks to what I refer to as Enterprise Modernization.

The WebSphere software portfolio offers a set of integration facilities that allows for reuse of those application components. And why not – they are tried and true for the business today, having been tested and deployed in production for long periods of time. Furthermore, the cost of leveraging these assets is about one-fifth the cost of developing new assets. So through discovery tools that catalog the individual application components and classify their associated interactions, our customers can easily integrate these assets. So we offer tools to find, and tools to test and simulate the new workload – all deployed on a runtime that handles the high-integrity transactional services our customers are demanding. Enterprise Modernization is the notion of being able to take that significant application investment and extract pieces and parts out of it in a programmatic way so you can build new applications that you couldn't in the past. So, Enterprise Modernization equals reuse; equals leveraging the tremendous assets in each of our large enterprises; equals integration; delivering new applications that couldn't be built before – in the highest-quality way with the least amount of errors. Those old factories then become a gold mine for new, faster development and deployment of revenue-generating applications.

WSDJ: You mentioned that you had categorized your customer input in four common areas.

Fordyce: Another common area I am seeing is that every enterprise wants to simplify the development cycle. They're asking, "How do I manage the entire development life cycle, modeling the actual processes in the business, and quickly delivering the application ideas the business is demanding? Analysis, design, code generation, cataloging for reuse, repository management – all are important elements in the application development process. The question is, "Where I can accelerate the

development over the entire life cycle of program code generation that all large enterprises are faced with?" This is a core theme that I keep hearing over and over again.

We have spoken about Enterprise Modernization as one area to address a common set of challenges our customers are facing. WebSphere provides a unique set of discovery, development, and deployment products to address these critical areas. Let's say number two on my list of common challenges is this notion of accelerating the development life cycle. First I need a common framework that allows for common services like editors, repositories, wizard masks, and so on – a framework that is recognizable whether you are building user interfaces or providing Web services integration. An approach that is recognizable or familiar no matter the development task or role will save training time.

IBM uses such a framework. It's known as Eclipse, which is included in our Studio tools. WebSphere Studio, then, provides a common life-cycle development approach, promotes a structured systematic development approach, and finally optimizes the team development environment. These three areas are oriented at accelerating the development life cycle.

The third area where I am hearing lots of challenges from our customers is in how to absorb and deploy in the most efficient manner all this new application technology that has arisen in the past several years. They are extremely focused on the most efficient methods, configurations, and processes.

WSDJ: Seeing if they can save a lot of money there, I assume?

Fordyce: Sure. All customers want the most cost-effective platform. However, when you start to ask about the platform, then you have to think about a business's most precious process – the application transaction. I try to guide our customers to think about the value of their transaction. Transaction value then allows for a pragmatic discussion on availability and availability costs.

WSDJ: I read something about availability costs – costs for a large stockbroker company could run \$100,000 per minute?

Fordyce: Transaction value varies widely within businesses and across market segments. The important point is that there has to be a value to a transaction or else why do it? Think about a brokerage company that's running financial transactions; they are running thousands of trades a minute. If even one of those trades is a mutual fund composed of several million shares of stock, then the transaction value could be as high as you state. So if their system is not available, meaning that either the code crashed or there is a physical problem with the server – or you can think of 10,000 reasons – then their cost of availability is very high, right? Their business, their factory, has a large dependency upon that application code.

WSDJ: Right. One of those guys goes down, they can lose \$100 million waiting.

Fordyce: Bang! Like that. I suggest that every application that gets produced out there has some sort of availability cost associated with it. Applications are not built today to do nothing; they are built to improve some efficiency or drive revenue to the corporations.

WSDJ: Then you agree that they are spending about \$100,000 a minute to keep the systems running?

Fordyce: The bill probably runs the gamut because it's highly dependent upon the business and the transactions they are driving. Something like a financial transaction for a brokerage house has a lot of value associated with it, but the same brokerage house could have a shipping department that's maybe delivering prospectuses to clients. That transaction has a much lower value, but does have some value associated, so you can range from \$500 a minute to \$1,000 per minute. The absolute value doesn't matter; approximating the transaction value is important.

Once we all begin to think in terms of transaction value we can start to optimize or put the correct efficiencies in place to service that value. Then a discussion about the kinds of investment needed relative to configurations for high availability or redundancy is easier.

With WebSphere we have a handful of guidelines that will assist with the trade-off that invariably has to be made; it's really working with individual clients and their particular workloads. We've done a lot in version 5 to build in functionality that allows for better management, functionality that will allow for higher availability, and administrative deployment facilities that allow for better change management. It's not just a piece of software that conforms to standards; it also puts in the functionality that allows you to run your business, sleep at night, and know that the transactions – or your core factory – are safe.


WSDJ: And there's one more?

Fordyce: And that brings us full circle to building and deploying on-demand applications. It's where we started our discussion. It's about providing the services in our runtimes and tools that take care of the error-prone application integration tasks. It's providing an infrastructure that allows for rapid change and at the same time provides unparalleled availability. It's about attention to operations and providing a truly on-demand operating environment that then allows our customers to concentrate on their people and business process so they remain competitive.

All of these items – whether it is a configuration decision or it is resiliency to disasters through facilities that allow response to change or the ability to integrate not only internally, but across an entire business exchange – that's what we refer to as our on-demand operating environment.

WSDJ: So Peter, where do you see WebSphere going in 2004?

Fordyce: First and foremost, we will see our customers quickly deploy new applications that generate new revenues for their business. IBM, and WebSphere in particular, will be focused on working with our customers from a holistic point of view to build the appropriate focus to allow our customers to be more competitive than ever before on their core competencies, with the right variable infrastructure that is resilient to unknowns and that adapts quickly to change.

We have an outstanding set of products today, and of course there will be continued investment with new versions in these products. It's guaranteed that those investments will be focused on allowing our customers to take one more step on the evolutionary path toward being an on-demand business. 

KENETIKS

WWW.KENETIKS.COM

Using log4j from a WebSphere-based application

A Powerful, Easy-to-Use Logging System

BY BIBHAS BHATTACHARYA

**ABOUT THE AUTHOR**

Bibhas Bhattacharya is chief technology officer at Web Age Solutions. As head of the consulting division, Bibhas helps companies implement solutions using the WebSphere family of products. Prior to joining Web Age Solutions, Bibhas worked at the IBM Toronto Software Lab as a technical planner for WebSphere Commerce Suite. A huge fan of Sherlock Holmes (the greatest debugger of all time), Bibhas spends his spare moments pondering Victorian England.

E-MAIL

bibhas@webagesolutions.com

The log4j logging system is powerful and easy to use. This article will show you how to best configure and use log4j from a WebSphere-based application. I will show you how to develop a J2EE 1.3 application and test it with WebSphere Application Server v5.

Initializing log4j

The initialization process informs log4j where to log, the log level, and various other configuration settings. The configuration parameters are stored in a Java properties file or an XML file. An application initializes log4j by pointing it to the configuration file.

There is no single entry point in a J2EE application, and applications are generally discouraged from making any assumptions about where files are located in the file system.

Consequently, the issue of where to store the configuration file and exactly when to perform the initialization has been problematic for many J2EE applications. There are two main ways to initialize log4j:

1. Call `PropertyConfigurator.configure(String propertyFileName)`: This function takes the full path name of the properties file that contains the log4j configuration information. It is difficult for an application to call this method without making any assumption about where its files are located in the file system.

2. Do not call `PropertyConfigurator.configure()`, instead allowing log4j to look for a properties file called `log4j.properties`, which is included in a J2EE module's classpath. The lookup takes place when the `Logger.getLogger()` method is called for the first time.

I recommend the latter approach in order to avoid having to write a singleton class to explicitly initialize log4j and make assumptions about the location of the configuration file.

A Typical log4j.properties File

Next, we will take a look at how to create a configuration file. Most applications will need to:

1. Specify the name of the log file.
2. Set the global log level. Optionally, we should be able to set a different log level for a specific component.
3. Rotate the log file.
4. View the timestamp and thread ID in each log entry.

A typical configuration file is shown in Listing 1. In the following lines of

code we set the global log level to ERROR. Other possible values are INFO and DEBUG.

```
#Set the global log level to
ERROR.
log4j.rootLogger=ERROR, ROOT
```

In the following section we set the location of the log file and how the file will be rotated.

```
log4j.appender.ROOT=org.apache.log4j.RollingFileAppender
log4j.appender.ROOT.File=myapplication.log
log4j.appender.ROOT.MaxFileSize=1000KB
#Keep 5 old files around.
log4j.appender.ROOT.MaxBackupIndex=5
```

The next section sets the format of a log entry. The format shown in the following code snippet will show the timestamp, thread ID, class name, log level, and the text message respectively.

```
log4j.appender.ROOT.layout=org.apache.log4j.PatternLayout
#A log format akin to
WebSphere's own
log4j.appender.ROOT.layout.ConversionPattern=[%d] %t %c
%-5p - %m%n
```

Finally, we override the log level for a specific Java package:

```
#Optionally override log
level of individual packages
log4j.logger.com.webage.ejbs=INFO
```

Using log4j from a Web Module

In this section I will show you how to use log4j from an application built entirely as a Web module (i.e., it does not use EJBs).

Start WSAD and create a J2EE 1.3 Enterprise Application Project called LogTest and a Web module within it

called LogTestWeb. Create a file called log4j.properties in the Java Source folder of the Web module. Add the configuration entries as shown in the previous section.

When you build the Web project, the properties file will be automatically copied into the WEB-INF/classes folder. Do not directly create the properties file in the WEB-INF/classes folder. WSAD will delete it the next time the Web project is built.

Download the latest log4j distribution. Extract the JAR file from the distribution (for example, log4j-1.2.8.jar) and copy it to the Web Content/WEB-INF/lib folder of the Web module.

Add a Java class to the Web module called com.webage.model.MyModel. Add a method called checkValid (String name, String value) as shown in Listing 2.

Add a servlet with the class name com.webage.servlets.MyServlet. Set the contents of the servlet's class as shown in Listing 3.

Test

Associate the LogTest project with a WebSphere Application Server v5. Run the servlet. By default, the firstName parameter is absent and the MyModel.checkValid method will throw an exception. The servlet will log the exception. The global log level is set to ERROR and log4j will add the log entry to the log file. The log file (myapplication.log) will be created in the WSAD installation root directory.

Open log4j.properties and enable the DEBUG-level log for the com.webage.model package by adding the following line.

```
log4j.logger.com.webage.model=DEBUG
```

Restart the LogTest project for the change to take effect. Run the servlet again and check the content of the log file. The DEBUG-level log from the MyModel class will be shown.

Troubleshooting

If you do not see the log file (myapplication.log) in the WSAD root installation directory, chances are log4j did not initialize properly. If log4j fails to

locate the log4j.properties file in the classpath, it will put the following message in the standard output.

```
[3/14/03 13:04:05:498 EST]
19daf47c SystemErr R
log4j:WARN No appenders
could be found for logger
(com.webage.servlets.MyServlet).
[3/14/03 13:04:05:498 EST]
19daf47c SystemErr R
log4j:WARN Please initialize
the log4j system properly.
```

In Unix systems, make sure that the spelling of the properties file is correct (log4j.properties is all lowercase).

Using log4j in an EJB Application

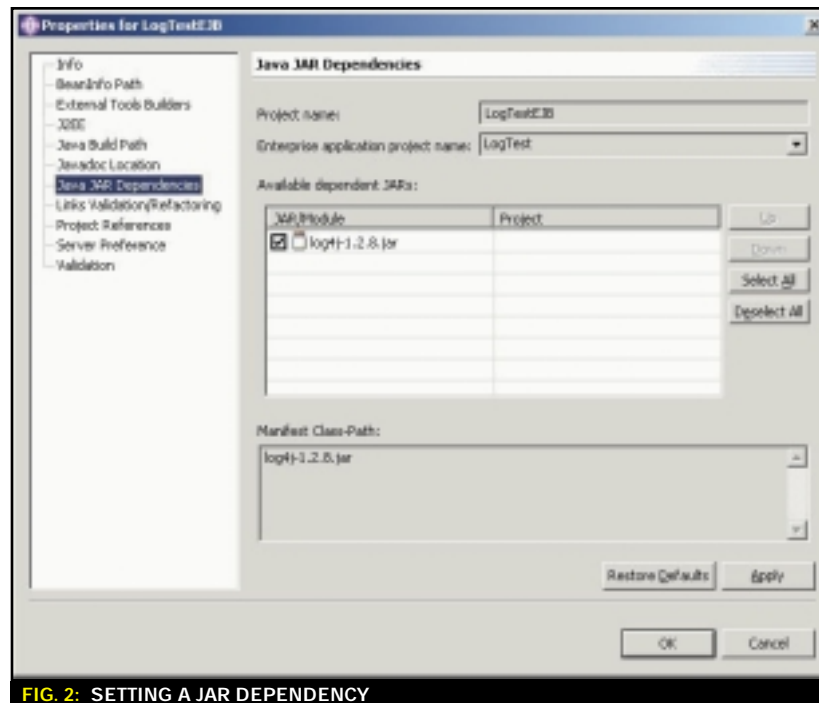
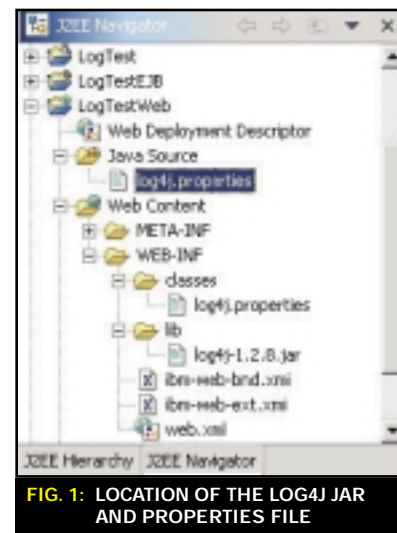
Create a new EJB module called LogTestEJB under the LogTest Enterprise Application module. Move the log4j JAR file from the WEB-INF/lib folder of the LogTestWeb module to the root directory of the LogTest project. Create a folder called lib under the LogTest project and move the log4j.properties file into the newly created directory.

Build a Java JAR dependency from the LogTestEJB project to the log4j JAR file (right-click on LogTestEJB

and select Properties. Select Java JAR Dependency and check the log4j JAR file).

Similarly, build a Java JAR dependency from the LogTestWeb project to the log4j JAR file and the LogTestEJB.jar file.

Add the LogTest/lib directory to the classpath of the server. To do this, switch to the Server perspective and double-click on your server in the Server Configuration view. Click on the Paths tab. Next to the Classpath list click on Add Folder. Select LogTest/lib.



Test

Add a simple session EJB called MySession to the LogTestEJB project. Add logging to the bean class MySessionBean.java, as shown in Listing 4.

Promote aMethod() to the remote interface and use the EJB from the servlet. Finally, test the servlet.

Deploying in WebSphere


Export your application from WSAD as an EAR file. Install the EAR in WebSphere. This will, by default, extract the contents of the EAR into the <WAS_ROOT>/installedApps/ <APP_NAME> folder. Copy the log4j.properties file from there to the <WAS_ROOT>/properties folder. WebSphere automatically adds this folder to the classpath of every application server.

By default, the <WAS_ROOT> directory is set up as the working directory of the application server. To create the log4j log file in the standard <WAS_ROOT>/logs directory, open log4j.properties and change the log filename as follows.

```
log4j.appender.ROOT.File=log
s/extranet.log
```

Conclusion

I demonstrated how to use log4j from an exclusively Web-based application as well as from an application that is both EJB and Web based. The goal was to simplify the process of development and administration. This was achieved in several ways.

1. To do a one-time initialization of log4j requires no additional coding.
2. The log4j JAR file is a part of the J2EE application; there is no need to distribute and install this JAR file separately from the application's EAR file.
3. In the case of a pure Web-based application, there is no additional administration involved. The application will be able find log4j.properties from the WEB-INF/classes folder. For an application that has EJB modules, you need to either change the application server's classpath to add the location of the properties file or store the properties file in a directory that is automatically added to the server's classpath. 

LISTING 1

```
#Set the global log level to ERROR.
log4j.rootLogger=ERROR, ROOT
log4j.appender.ROOT=org.apache.log4j.RollingFileAppender
log4j.appender.ROOT.File=myapplication.log
log4j.appender.ROOT.MaxFileSize=1000KB
#Keep 5 old files around.
log4j.appender.ROOT.MaxBackupIndex=5
log4j.appender.ROOT.layout=org.apache.log4j.PatternLayout
#A log format akin to WebSphere's own
log4j.appender.ROOT.layout.ConversionPattern=[%d] %t %c %-5p
- %m%n
#Optionally override log level of individual packages
log4j.logger.com.webage.ejb=INFO
```

LISTING 2

```
package com.webage.model;

import org.apache.log4j.Logger;

public class MyModel {
    static Logger logger = Logger.getLogger(MyModel.class);

    public void checkValid(String name, String value) throws
    Exception {

        logger.debug("ENTRY");
        logger.debug("Checking parameter: " + name);
        if (value == null) {
            throw new Exception("Parameter is absent.");
        }
        if (value.trim().length() == 0) {
            throw new Exception("Parameter is empty.");
        }

        logger.debug("EXIT");
    }
}
```

LISTING 3

```
package com.webage.servlets;

import java.io.IOException;
import javax.servlet.ServletException;

import javax.servlet.http.HttpServlet;
```

```
import javax.servlet.http.HttpServletRequest;

import javax.servlet.http.HttpServletResponse;
import org.apache.log4j.Logger;
import com.webage.model.MyModel;

public class MyServlet extends HttpServlet {
    Logger logger = Logger.getLogger(MyServlet.class);

    public void doGet(HttpServletRequest req,
        HttpServletResponse resp)
        throws ServletException, IOException {
        logger.debug("ENTRY");

        MyModel model = new MyModel();

        resp.getWriter().println("<h1>Log4J Test
        Servlet</h1>");
        try {
            model.checkValid("firstName",
                req.getParameter("firstName"));
        } catch (Exception e) {
            logger.error("doGet failed.", e);
        }

        logger.debug("EXIT");
    }

    public void init() throws ServletException {
        super.init();
        logger.info("Servlet initializing...");
    }
}
```

LISTING 4

```
import org.apache.log4j.Logger;

public class MySessionBean implements javax.ejb.SessionBean
{
    private javax.ejb.SessionContext mySessionCtx;
    Logger logger = Logger.getLogger(MySessionBean.class);
    //...
    public void aMethod() {
        logger.debug("ENTRY aMethod");
        logger.debug("EXIT aMethod");
    }
}
```

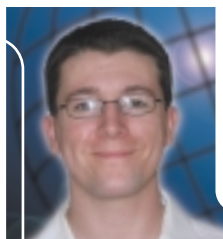
QUEST SOFTWARE

[HTTP://JAVA.QUEST.COM/JPROBE/WDJ](http://java.quest.com/jprobe/wdj)

Part 2: How to get the most out of this powerful toolset

Profiling in WebSphere Studio 5.0

BY ANDREW SONDGEROTH



ABOUT THE AUTHOR

Andrew Sondgeroth has been an instructor and developer for over seven years. He is currently employed with Intertech-Inc. (www.intertech-inc.com), based in Minnesota. Andrew teaches and develops courses on Java-related topics including Enterprise JavaBeans, WebSphere, and Java security. Andrew is visiting faculty on emerging technologies at the University of St. Thomas Graduate School of Business Management Center in Minneapolis/St. Paul, MN. He is also a regular presenter for the Twin Cities Java Users Group.

E-MAIL

asondgeroth@intertech-inc.com

I discussed many of the views in the Profiling Perspective of IBM's WebSphere Studio Application Developer (WSAD) 5.0 in Part 1 of this series, which focused on understanding the information displayed in the different views. In this article I will discuss code optimization and how to use WSAD to pinpoint areas of your applications that need performance tuning.

The Purpose of Profiling

To recap the explanation in Part 1, profiling is used to inspect the performance of code. Profiling allows analysis of application behavior for improving application efficiency. Profiling an application can also provide detection of major architectural problems early in the development life cycle. Profiling helps identify many behaviors and problems, including:

- Memory leaks/inefficient memory usage
- Poor method response times
- Frequent code block usage ("hot spots")
- Threading issues

Optimizing Code

As developers our natural instinct is to make our code as fast as possible. Generally speaking, this is a good rule of thumb. However, keep in mind that "efficiency" does not apply only to how fast code executes. An application requires a balance of efficiency in overall development, application performance, and maintenance. Sometimes the most optimized code is the least flexible and/or hardest to maintain. *Note:* In

most cases, code performance can be increased by using a more efficient algorithm rather than writing efficient (but obfuscated) code.

Below is a list of things to keep in mind when considering changes to code for the sake of optimization. Before making optimization changes to code, consider the following potential pitfalls:

- Modifying stable code opens the possibility of new bugs.
- Optimized code can become obfuscated and harder to maintain.
- Optimized code can be less extensible and/or less reusable.
- Valuable development time can be lost for the sake of minimal increase in performance.

This list is not intended to discourage optimization. It simply summarizes common mistakes made by developers that get overzealous in squeezing performance out an application.

The following sections explain a handful of basic techniques and solutions for performance tuning. Some techniques are supported by WSAD directly (such as refactoring), whereas other techniques can leverage WSAD tools for assistance.

Refactoring/Removing Subexpressions

Refactoring involves changing the flow of an application without affecting the actual behavior. In some cases, refactoring can increase performance as well as readability and maintainability. Renaming variables or turning a reusable snippet of code into a method are examples of refactoring. Removing subexpressions (another form of refactoring) involves rewriting repetitious code that produces the same result so that it is executed as few times as possible. Typically, a subexpression is removed by storing a result in a variable, rather than executing the same code repeatedly.

Whether a new method is created or a subexpression is removed, anything that could be considered "copy-paste" code is a good candidate for refactoring. Consider the following example for recalculating total amounts owed on multiple loans with the same interest rate:

```
double loan1 = 17000.00;
double loan2 = 12000.00;

double interest1 =
(getInterestRate() * loan1);
double interest2 =
(getInterestRate() * loan2);

loan1 += interest1;
loan2 += interest2;
```

Notice that the `generateInterestRate()` method is called more than once. Making the following changes will enable the code to perform more efficiently.

```
double loan1 = 17000.00;
double loan2 = 12000.00;

double interestRate =
getInterestRate();

loan1 += (interestRate *
loan1);
loan2 += (interestRate *
loan2);
```


WSAD has conveniently built in tools for refactoring. The refactoring tools are accessible through many different views (e.g., the Java editor, Outline view, J2EE Navigator). Right-clicking on code snippets, filenames, package names, or members provides a handful of refactoring techniques under the Refactor option (see Figure 1).

Memory Leaks/Inefficient Memory Usage

A memory leak occurs when data is no longer needed but cannot be unloaded from memory due to an error in coding. Mainly, when we talk about memory leaks we are referring to objects that remain in memory instead of being garbage-collected. Arguably, memory leaks are not possible in Java because once an object falls out of scope it will be garbage-collected. Though this is true, coding errors can still prevent unneeded objects from being garbage-collected at the earliest possible time (which essentially is a memory leak). This is an inefficient usage of memory and can possibly cause severe performance problems.

The WSAD profiling toolset has a handful of mechanisms to help identify memory leaks in applications. There are three main views in the WSAD Profiling perspective used to determine if memory leaks are occurring:

1. Instance Statistics view
2. Heap view
3. Object References view

Note: For performance and usability, IBM is adding a Class Statistics view to WSAD 5.0.1, which will replace Heap view.

Each of the three views requires collection of object reference data, accomplished at any point while profiling an application. To do so, in the Profiling Monitor view, right-click on the monitor for your application and select "Collect Object References". Collecting the references provides a snapshot of how objects are referencing each other, which objects have been garbage-collected, and which objects are held in memory. Also, invoke the garbage collector from the same right-click menu by selecting "Run Garbage Collection". *Note:* To identify memory leaks, each of the three views requires enabling the "Collect instance-level information"

option in the final step of the wizard when starting a new profiling process.

When checking for objects that have or have not been garbage-collected, the Instance Statistics view and Heap view provide the information at a glance. The Instance Statistics view provides information on each object instance recorded by the profiler. The "Collected" field displays "false" if the instance has not yet been garbage-collected and "true" if it has. As for the Heap view, the lower-right portion displays instance information graphically. Diamonds represent class objects, solid rectangles represent object instances still held in memory, and empty rectangles represent object instances that have been garbage-collected.

The Object References view provides a more detailed graphical representation of objects, as well as any references pointing to each object instance. By using the Instance Statistics and Heap views, objects that should have been garbage-collected are easily identified. Then, use the Object References

view to locate the exact references that are preventing garbage collection.

When the offending reference is found, simply right-click on the object holding the reference and select "Open Source" to view the code responsible.

After finding a memory leak, examine the code and consider the scope of the object. For example, a static variable could hold an object reference when an instance variable would provide the same functionality.

Memory leaks can be caused and corrected in a number of different ways. The following example demonstrates a simple memory leak:

```
public class Sieve {
    private int[] data;

    public Sieve(int[] data){
        this.data = data;
    }

    public int processData(int index){
        return (data[index] * 2);
    }
}
```

Notice that the data variable is a member variable and will hold a reference to a given object. Now, let's take a look at how a memory leak could possibly occur:

```
int[] array = {13, 45, 21, 93, 20};
Sieve leaky = new Sieve(array);
for(int i=0; i < array.length; i++){
    array[i] = leaky.processData(i);
}
array = null;
//Sieve still holds a
//reference to the same object
```

A simple rewrite of our Sieve class can correct this leak:

```
i
public class Sieve {
    public int processData(int data){
        return ( data * 2 );
    }
}
```



FIG. 1: WSAD SUPPORTS A NUMBER OF REFACTORING OPTIONS

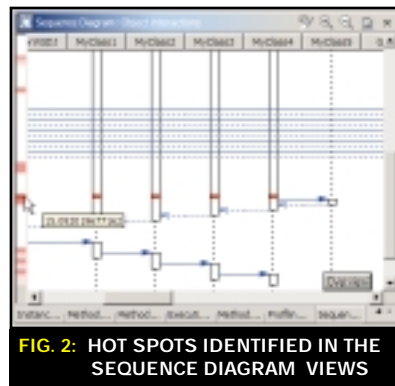


FIG. 2: HOT SPOTS IDENTIFIED IN THE SEQUENCE DIAGRAM VIEWS

Note: The member variable has been removed and information is processed only at a local level. Now when the Sieve class is used it no longer results in a memory leak:

```
Sieve leaky = new Sieve();

for(int i=0; i <
array.length; i++){
    array[i] =
    leaky.processData(array[i]);
}

array = null;
//Sieve does not hold a
//reference to the object
```

For more information on memory leaks and proper coding techniques, see the References section at the end of the article.

POOR METHOD RESPONSE TIMES

Most often, poor method response times are related to object creation or the obtaining of resources (such as opening an input stream or establishing a connection). In some cases, nonintrusive solutions such as changing a JDBC driver are enough to boost performance. Unfortunately, more severe problems or issues beyond the control of the developer can

also cause a performance nightmare (such as network traffic, outdated systems, etc.).

WSAD's Execution Flow, Execution Flow Table, and related views help isolate method or submethod calls responsible for abnormal execution times. Long method calls are easily identifiable using the graphical Execution Flow, Method Invocation, or Method Execution views. Each view represents method calls and times with vertical bars; thus long bars represent long execution times. To examine the statistics of a particular method, right-click on the vertical bar and select "Show Invocation Table" or "Show Execution Table", depending on which view is currently displayed.

The statistical data displayed for each method call or thread can be expanded to examine submethod calls. Doing so will help identify precisely which method or methods need modification to optimize performance. In each of the views mentioned above, right-clicking on any method will give the option "Open Source" to allow prompt examination of the offending code.

HOT SPOTS

Hot spots are areas of code that consume large amounts processing time. Hot spots can be broken down

into two categories: long execution times or frequently accessed code blocks (resulting in large cumulative execution times).

A number of views help identify hot spots. The easiest view to use is the Sequence Diagram. The vertical bar on the left-hand side of the view contains red blocks. Darker red blocks represent methods or code blocks that take the most time to execute. Double-clicking on a red block marks each method involved in the long execution call.

Hot spots can also be identified using the Heap view. As explained in Part I, the Heap view displays method and object information, such as memory usage and execution time. It displays methods with excessive execution times as red blocks.

A hot spot may be the result of poor method performance (as detailed earlier) or it may suggest a candidate for other optimization techniques such as refactoring or implementing a better algorithm.

Figures 2 and 3 illustrate how to use the profiling views to find hot spots.

THREAD ISSUES

As powerful as the WSAD profiling toolset is, threading issues are tricky to diagnose. To test issues such as thread starvation, thread races, and deadlocks requires a thorough understanding of the application, potential pitfalls, and isolated tests focusing specifically on such issues. Graphical views are the best tools to rely on to identify thread-related problems. Graphical views allow finding long- or short-running threads at a glance.

The Execution Flow-type views can identify thread starvation issues. Thread starvation occurs when a thread is unable to perform an adequate amount of work in a given time frame. Possible reasons for thread starvation include higher-priority threads hogging CPU time or staying in a wait or locked state for frequent/long periods of time. When examining Execution Flow-type views, look for threads or method calls that have longer vertical bars (execution time) than expected.

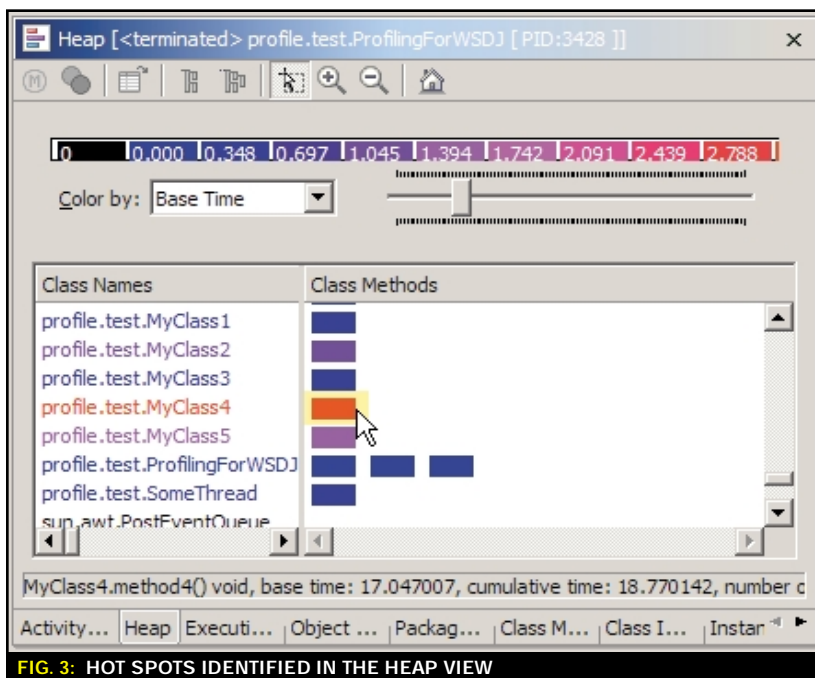


FIG. 3: HOT SPOTS IDENTIFIED IN THE HEAP VIEW

H&W COMPUTER

WWW.HWCS.COM

Deadlocks are more easily spotted with the Sequence Diagram view. Method calls that fail to execute or return in addition to being flagged as a hot spot are potential deadlocks. Figure 4 shows an intentional deadlock.

THE STRINGBUFFER MYTH: SEEDING STRINGBUFFERS PROPERLY

A very common optimization technique for Java is to use a StringBuffer instead of a String when concatenating. The reasoning is that using the + or += operators to concatenate strings causes a new object to be created for each concatenation. On the other hand, the StringBuffer contains an append() method, which allows dynamic string growth without having to create a new String object each time. Generally speaking, this is actually not more efficient. The reason is twofold.

The first reason has to do with the bytecode produced by the Java compiler. The compiler is smart enough to recognize when a number of concatenations are going to be executed and automatically creates a StringBuffer. Each concatenation operation is converted to append() calls behind the scenes. Essentially, manually coding a StringBuffer is unnecessary, unless the following reason is considered.

The second reason is the real kicker. In order for a StringBuffer to truly optimize concatenation, it must be seeded properly. In other words, it needs to be given an appropriate initial size. This is because the StringBuffer keeps the characters of the string it is maintaining in an array. When append() is called, the StringBuffer checks the size of the character array versus the estimated size of the new string. If the estimated size is larger than the actual array, a new array is created and the old

array is copied to the new one. Thus, StringBuffer is not only creating a new object for each concatenation, but it also incurs the overhead of copying all the indexes of the original array.

Using the default constructor of StringBuffer defaults the character array to a measly 16 characters. SQL strings assuredly will exceed 16 characters and constantly cause new character array objects to be created. Even using the StringBuffer's constructor that takes in a String only initializes the character array as 16 characters more than the length of the String argument. The best way to be sure that the StringBuffer will benefit the performance of your application is to give it a large enough initial seed that it will need to create its character array only once.

Symptom Database

One of the most useful tools for troubleshooting application errors is a symptom database. A symptom database is an XML file containing information for specific errors and messages. All the symptoms (errors and messages) have a suggested solution or at least a specified resource to refer to for more help. Some symptoms give a more detailed explanation of what caused the error.

When profiling or running an application, a log file containing messages, warnings, and errors is created by WAS and WSAD. This log file can be used in conjunction with a symptom database to resolve common problems. Preexisting symptom databases can be imported, or you can create your own.

WSAD does not come with a default symptom database but one can be downloaded manually from IBM's public FTP site (see the Resources section). Alternatively, you can use WSAD's import wizard to automatically download IBM's latest symptom database by following these steps:

1. Select File -> Import...
2. Select "Symptom Database file" and click Next.
3. Specify symptom database location and target.
 - Select "Remote Host" and select the appropriate server type.
 - Specify which project will receive the symptom database.
 - WSAD should already be pointing to the appropriate filename.
 - Click Finish when you are ready.

Note: The default symptom database from IBM contains only server-related symptoms and solutions.

To use a symptom database, an application or server log file must be imported (by selecting File -> Import... and choosing either "Logging Utilities XML Log File" or "WebSphere Application Server Log File"). The log file will display log entries that can be more closely examined. Right-click on an entry to display options to load a symptom database or analyze the entry against a symptom database that has already been loaded.


Conclusion

The purpose of profiling an application is to evaluate performance. WSAD's profiling toolset dishes out powerful support for finding efficiency issues. Issues such as memory leaks, hot spots, and threading issues are easier to isolate and identify with WSAD than with standard debugging tools or other mechanisms. Profiling helps identify potentially damaging code and pinpoint sections of code as likely candidates for optimization. Just keep in mind that optimizing performance to a ridiculous degree can possibly hurt code maintainability, flexibility, and/or extensibility.

Resources

- IBM's Symptom Database: <ftp://ftp.software.ibm.com/software/webSphere/info/tools/loganalyzer/symptoms/std/symptomdb.xml>
- Larman, C. (1999). *Java 2 Performance and Idiom Guide*. Prentice Hall.
- *Make Java Fast: Optimize*: www.javaworld.com/javaworld/jw-04-1997/jw-04-optimize.html
- *Java Optimization*: www-2.cs.cmu.edu/~jch/java/optimization.html
- Bentley, J. (2000). *Programming Pearls (2nd Edition)*. Addison-Wesley.
- WSAD 5.0 Help Documentation

Acknowledgments

I would like to thank the following people for help with this article: Jeff Jensen of go-e-biz.com for all his thoroughness; my teammates at Intertech Inc. for their support; and also Mike Bresnahan of Fruition Inc. for his technical insight. 

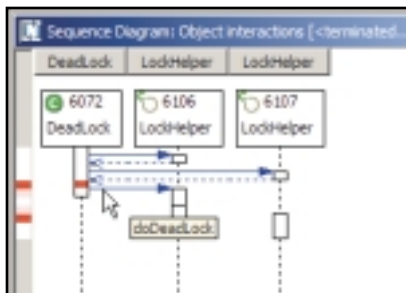


FIG. 4: NO METHOD RETURN PLUS A HOT SPOT INDICATES LIKELY DEADLOCK

VERSATA

WWW.VERSATA.COM/BUSINESSLOGICDESIGNER

J-Unit Testing in WebSphere

Stable code is a good thing

— BY KULVIR SINGH BHOGAL —

WebSphere Studio Application Developer, based on the Eclipse platform, is designed with the Java developer in mind. Extreme programming mandates that Java classes be unit-level tested in an automated fashion. Extreme programming takes a rather pessimistic (though frequently realistic) outlook on code. It assumes that if no automated test case is written for a given code artifact, then the given artifact must be assumed to be broken. Your organization's productivity can be slowed or even paralyzed by an unstable code base. Thoroughly unit-tested code can prevent an ugly situation from showing its face. Though you always have the option to test by hand, doing so can be a tedious and error-prone process. Comprehensively designed, automated test cases can increase code stability significantly.



ABOUT THE AUTHOR

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

E-MAIL

kbhogal@us.ibm.com

JUnit (<http://sourceforge.net/projects/junit>) has become the de facto standard for Java class unit testing. In this article, I'll cover the JUnit framework and describe how it has been integrated into WebSphere Studio Application Developer (WSAD) version 5. I will present an application to which you can apply JUnit's concepts hands-on.

Some JUnit Lingo

There are three major concepts that we need to cover before we dive into our unit-testing endeavors:

- **Test fixture:** Provides resources needed by test cases to run. Later, we'll use our test fixture to set up a database connection that we will use in our example.
- **Test case:** A test case is a collection of tests that are designed to verify the behavior of a unit of your application. In the Java realm, these "units" are typically your

Java classes. The motivation behind testing in such a unit-based, isolated fashion (apart from the full system) is so that you know the individual components that make up your entire application work as expected before moving onto integration. This way, bugs can be identified before the individual code artifacts are integrated.

- **Test suite:** A grouping of test cases.

Getting Ready

For educational purposes, we'll be testing a small Java application that reads and writes data from an IBM DB2 database. Of course, I am assuming that you have DB2 UDB (Universal Database) installed on your machine. If not, you can download a trial of DB2 UDB from www-3.ibm.com/software/data/db2/udb/downloads.html. Alternatively, you can modify the code presented in this article to work with a database that you have access to. Our database will consist of a simple database with one table. To prepare our database, let's hit the DB2 command line (Start> Programs> IBM DB2> DB2> Command Line Tools>Command Line Processor) after the database manager is started. (Use db2start to ensure this.)

```
db2 => create db junitdb
```

Then we connect to the database:

```
db2 => connect to junitdb user db2admin
using db2admin
```

In the foregoing example it is assumed that you have an ID of db2admin created on your system with a password of db2admin.

Next we create a table:

```
db2 => create table junittable (lastname
varchar(25) not null, firstname varchar(25)
not null, email varchar(25) not null primary
key)
```

As you can see, the table we created is a simple one that allows you to house a last name, first name, and e-mail address. We have added a primary key for the e-mail column, which controls the entry of duplicate records (differentiated by the e-mail address).



Our Test Program

The code for the program we are going to test is shown in Listing 1. As you can see, the code simply takes a database connection object and performs either an insert or delete on the JUNITTABLE using JDBC.

To get things set up, create a Java project in your WSAD workspace called JUnitExample. Right-click on the Java project to get to the properties of the Java project. We need to add a couple of files in the Java build path (to be more specific, the Libraries section). Add the external JAR of db2java.zip (which should be located in the [db2install\dir] \sqlib\java directory if you are using DB2 8.1), where [db2install\dir] is the directory where DB2 was installed (typically c:\Program Files\IBM). Also, we need to add the JAR file, junit.jar. This JAR file contains the JUnit classes we need. The JAR file should be located in the [WSAD 5 Install Dir]\eclipse\plugins\org.junit_3.7.0 directory (see Figure 1).

Next, in your Java project create a package named com.ibm.junitexample and import the JUnitExample.java file (which you can obtain from www.sys-con.com/web-sphere/sourcec.cfm), shown in Listing 1. After doing this, your project structure should look like that seen in Figure 2.

Ready, Set, Test!

At this point, we are ready to create our tests. Create another package, com.ibm.junitexampletest, under the JUnitExample project. We'll house our tests in this new package. Such package separation is a good idea so you can easily distinguish your test code. That way, when you move on to a production system, you can easily leave the test code behind.

Under the project, create a Java class called JUnitExampleTest and have it extend junit.FrameWork.TestCase as shown in Figure 3.

It is important to note that the specification of the TestCase superclass is essential. It is from this superclass that we will inherit the magic we need to unit-test. Also, be sure to select the "Constructors from superclass" option, as well as the "Inherited abstract methods" option for method stubs you would like created, as shown in Figure 3.

Let's fast-forward and show you how the final product of JUnitExampleTest.java should look (see Listing 2). Again, this code can be found in the source code zip file. Take a quick look at Listing 2. Throughout the rest of this article I'll dissect the code to make sure you have a grasp of what's going on.

Setting Things Up and Tearing Things Down

Let's start with the setUp() method shown in Listing 2. If you recall from our earlier JUnit lingo discussion, fixture objects are those resources needed by our tests to run. We establish these so-called fixture objects in the setUp() method. Note that you must use this exact method name, as we are overriding an inherited method.

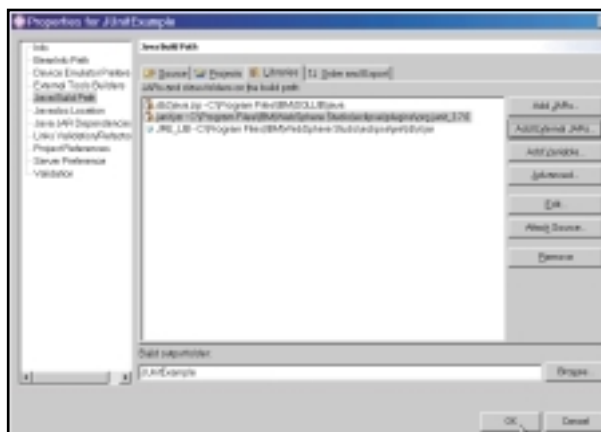


FIG. 1: ADDING APPROPRIATE EXTERNAL JARS

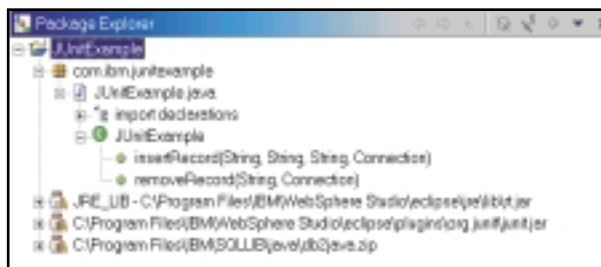


FIG. 2: JAVA PROJECT STRUCTURE IN PACKAGE EXPLORER VIEW



FIG. 3: CREATING YOUR NEW TEST CLASS

In this method, we establish our DB2 connection using the appropriate URL, username, and password. We also create a JUnitExample object, which is an instantiation of the class we want to test.

If you peruse the code, you'll also notice that a tear-

Down() method is paired with the setUp() method (see Listing 2). This method is used to release resources we allocated in our setUp() method. Accordingly, since we established a database

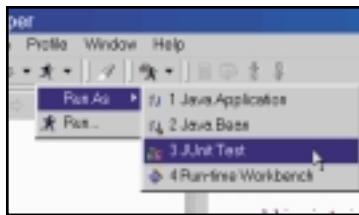


FIG. 4: RUNNING YOUR JUNIT TEST

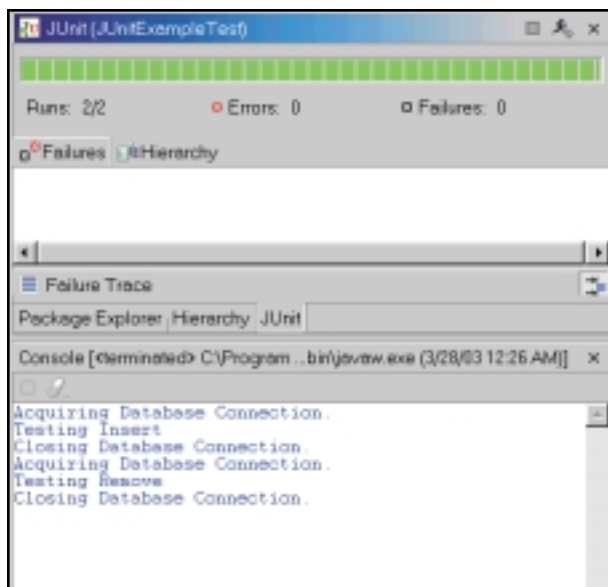


FIG. 5: THE FRUITS OF YOUR TESTING LABORS

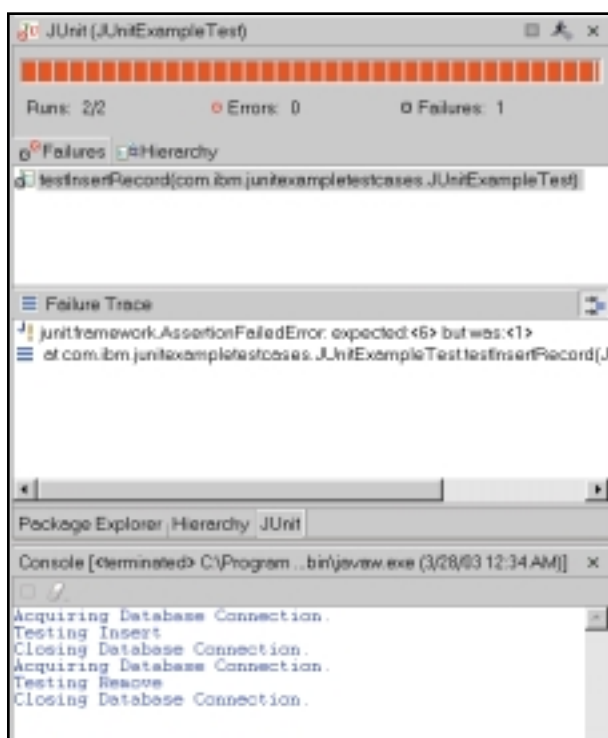


FIG. 6: A TEST GONE BAD

connection, we will close the connection to be database Good Samaritans. We don't worry about the JUnitExample object we created, since it will be picked up by the garbage collector when it gets around to it.

It is important to note that the setUp() and tearDown() methods are run before and after each of the test methods that you will build just a little later.

Creating Some Test Methods

Test methods in the JUnit world must be prefixed by the word "test". This is necessary so the JUnit framework can figure out which methods of your test class can be added to your testing suite. testInsertRecord() and testRemoveRecord() (see Listing 2) are a couple of test methods that are used in JUnitExampleTest.java.

In testInsertRecord(), a record is inserted into the database to test the insertRecord method of the JUnitExample class. As the insertRecord method ends up calling the executeUpdate method, the laws of JDBC let us know that a successful insert should return a value of 1. Accordingly, we tap into the assert features of JUnit to check if our insert occurred properly. If our assertion fails, then the assert method will throw an AssertionError. This is a (deliberately) unchecked exception that will cause our test to fail. The code for testRemoveRecord() is very similar to that of testInsertRecord(), except that here we expect a value of 0 for a successful database record removal.

Building the TestSuite

The static suite method is where we actually define our test suite and specify the tests that we want performed. Using this JUnit technique, we can also specify the order in which the tests are performed:

```
public static Test suite()
{
    TestSuite mySuite = new TestSuite();
    mySuite.addTest(new
        JUnitExampleTest("testInsertRecord"));
    mySuite.addTest(new
        JUnitExampleTest("testRemoveRecord"));
    return mySuite;
}
```

Alternatively, we could have the code:

```
public static Test suite()
{
    return new
        TestSuite(JUnitExampleTest.class);
}
```

In the code segment above, the JUnit framework would use reflection behind the scenes to look for methods in the JUnitExampleTest class that are prefixed by "test", and in turn will use them as test cases. This methodology is rather powerful and automated; however, it doesn't give the user the granularity of being able to pick the order in which the tests are run and which tests the user wants to run. Since in our particular setup it is important to run the insert test before the remove test, the explicit approach is more desirable.

Running the Suite

To run our JUnit tests, click on the run icon on your toolbar and choose Run As>JUnit Test, as seen in Figure 4.

After you run the JUnit test, you should see information written to your console, as well as a JUnit window like that shown in Figure 5.

In the case depicted in Figure 5, both of the tests were successful. This is indicated by the green status bar, as well as the fact that no information is written to the Failures section. Now if we change our assertion in the testInsertRecord method to assertEquals(6, c), which we know is not supposed to happen, and we run our JUnit tests, we'll get something similar to the screen shown in Figure 6.

What JUnit Doesn't Do for You

It is important to note that JUnit does what its name implies – unit testing. It does not cover load/stress testing. Other open source testing tools such as JMeter can be leveraged to do this. Similarly, in the specific arena of testing

J2EE components, tools such as HttpUnit and Cactus can be leveraged. Check out my article about HttpUnit at: www.7b.boulder.ibm.com/dmdd/library/techarticle/0303bhogal/0303bhogal.html

Conclusion

In this article, you learned how to leverage JUnit to unit-test your Java artifacts within WebSphere Studio Application Developer version 5. Unit testing is essential to maintaining code stability. As you have seen, creating unit tests in WSAD 5 is not too difficult. Sure, it's a bit of a pain. However, the value-add of unit testing can save your organization a huge chunk of change, quite possibly deterring the deployment of bad code to production.

Acknowledgment

The author wishes to thank Matt Oberlin and Kwang S. Kang for their review of the article and their help in making it better. 

LISTING 1: JUNITEXAMPLE.JAVA

```
package com.ibm.junitexample;
import java.sql.*;

public class JUnitExample {

    /**
     * Inserts a database record.
     */
    public int insertRecord(
        String firstName,
        String lastName,
        String email,
        Connection db2Conn) {
        int returnMe = -1;
        try {
            String myInsertStatement =
                "insert into JUNITTABLE values ('"
                    + firstName
                    + "','"
                    + lastName
                    + "','"
                    + email
                    + "')";

            Statement st = db2Conn.createStatement();
            returnMe = st.executeUpdate(myInsertStatement);
        } catch (Exception e) {
            e.printStackTrace();
            System.out.println("An Error has occurred." + e);
        }
        return returnMe;
    }

    /**
     * Removes a database record.
     */
    public int removeRecord(String email, Connection db2Conn) {
        int returnMe = -1;
        try {
            String myInsertStatement =
                "delete from JUNITTABLE where email = '"
                    + email + "'";

            Statement st = db2Conn.createStatement();
            returnMe = st.executeUpdate(myInsertStatement);
        } catch (Exception e) {
            e.printStackTrace();
            System.out.println("An Error has occurred." + e);
        }
        return returnMe;
    }
}
```

LISTING 2: JUNITEXAMPLETEST.JAVA

```
package com.ibm.junitexampletest;

import java.sql.Connection;
import java.sql.DriverManager;
import java.sql.SQLException;
import com.ibm.junitexample.JUnitExample;
import junit.framework.Test;
import junit.framework.TestCase;
import junit.framework.TestSuite;

import junit.textui.TestRunner;
```

```
public class JUnitExampleTest extends TestCase {
    Connection db2Conn = null;
    JUnitExample junitExTest = null;
    protected void setUp() {
        try {
            junitExTest = new JUnitExample();
            System.out.println("Acquiring Database
Connection.");
            Class.forName("COM.ibm.db2.jdbc.app.DB2Driver");
            String dbName = "junitdb";
            String url = "jdbc:db2:" + dbName;
            String userName = "db2admin";
            String password = "db2admin";
            db2Conn = DriverManager.getConnection(url,
userName, password);
        } catch (Exception e) {
            e.printStackTrace();
            System.out.println("An Error has occurred." + e);
        }
    }

    protected void tearDown() {
        try {
            System.out.println("Closing Database Connection.");
            db2Conn.close();
        } catch (Exception e) {
            e.printStackTrace();
            System.out.println("An Error has occurred." + e);
        }
    }

    public JUnitExampleTest(String arg0) {
        super(arg0);
    }

    public static Test suite() {
        TestSuite mySuite = new TestSuite();
        mySuite.addTest(new JUnitExampleTest("testInsertRecord"));
        mySuite.addTest(new JUnitExampleTest("testRemoveRecord"));
        return mySuite;
    }

    /**
     * Tests insertion of database record.
     */
    public void testInsertRecord() {
        System.out.println("Testing Insert");
        int c =
            junitExTest.insertRecord(
                "Kulvir",
                "Bhogal",
                "someone@somewhere.com",
                db2Conn);

        assertEquals(1, c);
    }

    /**
     * Tests removal of database record.
     */
    public void testRemoveRecord() {
        System.out.println("Testing Remove");
        int c =
            junitExTest.removeRecord("someone@somewhere.com",
db2Conn);

        assertEquals(1, c);
    }
}
```



ABOUT THE AUTHOR

Troy Holmes has been working in the IT industry for 14 years and is currently a J2EE architect using WebSphere 5.0. He has completed several large-scale J2EE applications using WebSphere. Troy is a certified Java programmer who has been working in the Java environment for five years. He also has more than five years of experience with Oracle and two years of experience with Informix. His professional background ranges from system administrator to system architect.

E-MAIL

troy.holmes@prizum.com

Demystifying the Art of Integrating Enterprise Applications

Quickly link disparate systems

— BY TROY HOLMES AND JAY HOTALING —

Enterprise Architecture Integration (EAI) is the process of integrating enterprise software that enables reuse of existing applications and allows for quick integration of best-of-breed, custom off-the-shelf (COTS) products. This article will attempt to demystify the art of integrating enterprise applications and to supply an approach that can be used to quickly link disparate systems.

EAI Primer

In today's IT environment, it is rare to work on a project that does not require integration with a COTS product or legacy system. Corporations are often unable to commit the vast sums of money and resources required to rewrite existing systems to produce a new "integrated" solution. Instead, many CIOs have opted to integrate their existing legacy systems rather than attempt a high-risk all-or-nothing approach, often resulting in an overall decrease in development costs through the integration of existing applications with new COTS products. This enables companies to leverage the existing legacy code while adding new innovative technology solutions to enhance their business processes. It is like putting together a puzzle. We'll look at reporting tools, ETL (Extract, Transform, Load) products, and middleware products and try to determine the fastest, least disruptive approach to integrating them with existing legacy systems in a logical manner.

There are many EAI products on the market today and we as architects are challenged to determine which is the best. The main question to answer is what type of integration are we trying to accomplish? The two choices that stand out are messaging and custom services. Messaging is the process of sending a message from one application and receiving it with another application. The message is then

transformed into a format that the receiving application can use to complete a business process. Custom services, on the other hand, involve creating services or methods that can be accessed by an external system. These services receive invocations and complete business processes.

Obviously, messaging is the lowest common denominator for communicating between disparate systems; therefore if you are connecting legacy applications, messaging would be the easiest approach. In this case we would most likely select WebSphere MQSeries (WMQ) – which has become the de facto standard for messaging in today's market – as the messaging infrastructure. The WMQ product is widely recognized by customers as the key enabling technology for business integration.

The next step would be to select a product to provide a "central hub" mechanism to allow the decoupling of applications. This movement away from direct point-to-point connections between systems reduces the total number of interfaces. Such a hub can also provide routing and transformation of message data, reducing the need for each system to know how to map its data to each target system.

One of the products IBM introduced to provide these features is WebSphere Business Integration Message Broker, which offers the ability to create message flows that define rules for the routing of messages, transform message formats, augment message data using external data sources, propagate or aggregate messages, and support publish/subscribe functionality.

Another key concept to understand is how the entities in each system are related to each other. In a perfect world with perfectly integrated systems, a common key would be used by all systems in your enterprise. However, most of us attempting to integrate our enterprise are not so lucky. Since different systems usually have different keys, it is necessary to define the dynamic relationship metadata used to map the corresponding entities from one system to another. While this is not a trivial endeavor, a complete discussion of how to achieve it is outside the scope of this document. For our example we have defined a cross-reference (XREF) table in a DB2 database that contains this legacy system mapping.

JMS Primer

The next issue is how to integrate this messaging system into a J2EE application. Here we have two choices. Vendors provide proprietary APIs that enable the use of their products. However, the Java community, recognizing the need for a standardized messaging service, has created the Java Messaging Service (JMS) API. A primary goal of enterprise application development is to limit the exposure to change within key components of the architecture; therefore, JMS would be the preferred method in a J2EE application. This service enables applications to communicate to messaging architectures without worrying about low-level socket coding.

In addition, JMS provides a common API and provider framework that enables development of portable enterprise applications regardless of the underlying messaging architecture. JMS provides an infrastructure that isolates changes made in the messaging environment. Therefore, if the enterprise decided to use a different message service the Java code would require only minor modifications. The addition of the JMS API enhances the J2EE platform by enabling loosely coupled, reliable, asynchronous interactions among J2EE components and the messaging architecture. Figure 1 is a visual representation of the architecture outlined thus far.

Now that you have a basic understanding of the design for an architecture that supports messaging, it is time to get into the details of how to do it. We will first discuss how to configure WebSphere Business Integration Message Broker and WMQ to support messaging between a J2EE application and a legacy application; next, we will outline how to configure JMS in a J2EE environment. Finally, we will discuss some options for designing legacy applications to communicate with WMQ.

For our example, we have chosen to solve a common problem (mentioned earlier in the article) that exists in many organizations – point-to-point (P2P) interfaces between existing systems. Let's say that a company's busi-

ness infrastructure consists of a central reservation system, a rewards (frequent traveler) system, a profiling system (exposed via the Web to allow customers to update their own travel profile information), and a front office property management system (PMS). Each of these systems has its own flavor of a "customer" database. These databases are kept in sync via P2P batch interfaces or through direct system-to-system calls. This problem is illustrated in Figure 2.

How then, do we integrate business processes that span these legacy systems without creating/maintaining the numerous P2P interfaces required to interconnect all the systems? Each system now has knowledge of the other systems that must participate in keeping the customer data in sync. Due to the batch nature of some of the interfaces, it is possible that there will be data latency/consistency issues and each system must also have specialized routines to prepare the data to send to the receiving systems or to accept inbound data itself.

If a customer comes to the company's Web site and updates his or her profile (a change of address in the online Web-based profiling system), then calls to verify a reservation, the customer would expect the information to have been updated and available.

We want the update made via the profiling system to update the various legacy systems, utilizing the unique business logic of each of the systems, thus syncing the customer profile across the enterprise. Of course, the company could create a centralized customer database and modify all the existing systems or develop a new system that includes the functionality of the four existing applications, but as you can guess, that would be a very expensive/risky proposition.

To illustrate our approach to this problem, we will assume that the profiling system is a J2EE application connected to an Oracle database and that the remaining systems are legacy applications that communicate via XML over MQ. Based on our requirements, the profiling system will need to update the PMS, reservation, and rewards systems with any profile information the customer has modified or created. We input the profile change request via a Web site and then propagate the change to the legacy applications. Before we can describe the design of the profiling system, we will discuss the WMQ environment required to support this integration effort.

Definition of the WMQ Environment

A complete description of how to install and configure WebSphere MQ is beyond the scope of this document. However, we will review some of the important concepts involved.

- **Queue manager:** Performs connection and queue management functions. You must first connect to a queue manager before accessing a queue. The queue manager also handles the routing of messages to other queue managers in the topology through the use of channels and listeners.

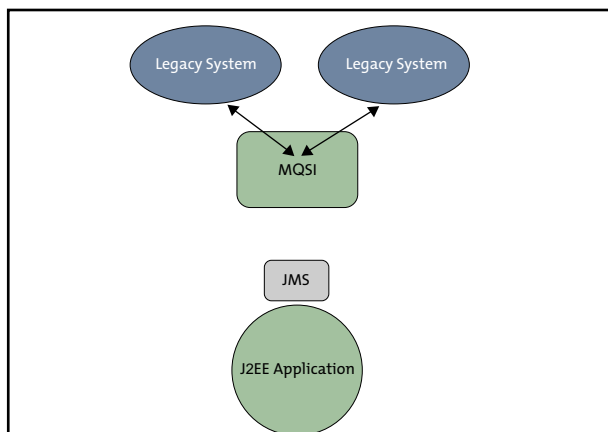


FIG. 1: THE ARCHITECTURE SO FAR



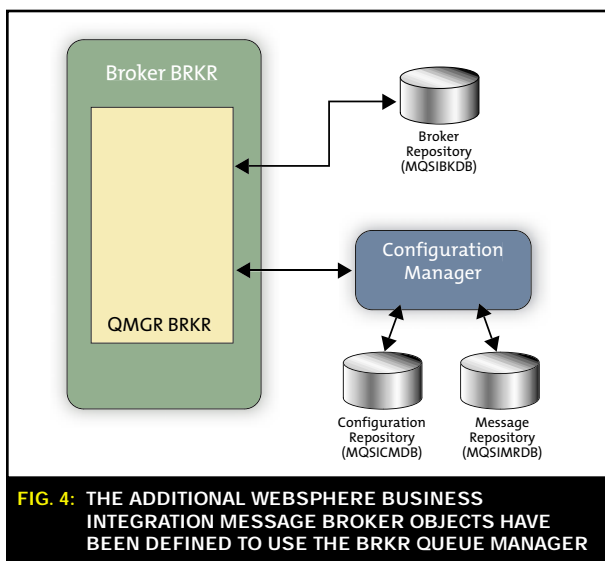
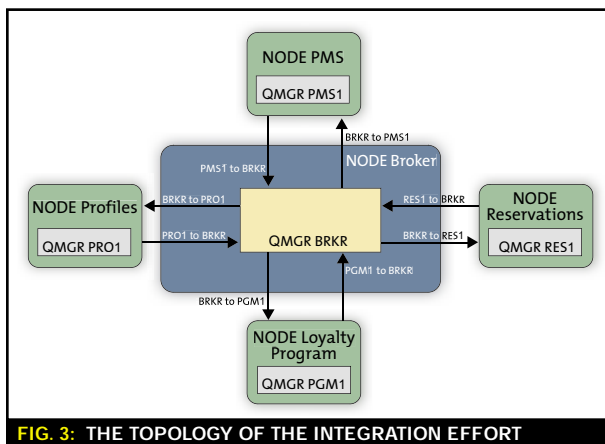
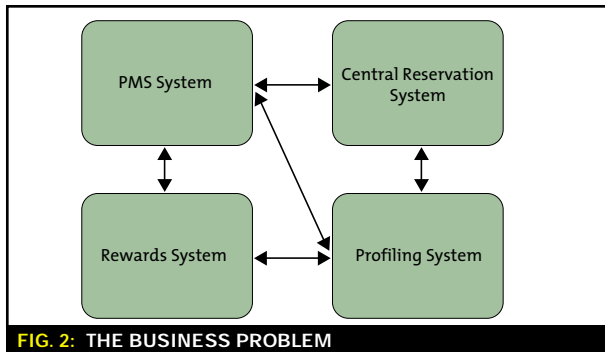
ABOUT THE AUTHOR

Jay Hotaling has been working as a consultant in the IT industry for over 13 years and is currently working on J2EE/EAI technology projects involving WebSphere MQ Integrator and WebSphere Application Server.

E-MAIL

jay.hotaling@prizum.com

- **Channels:** Before queue managers can exchange messages, communication channels must be defined. A “sender” channel is defined on the source queue manager and a corresponding “receiver” channel is defined for the target queue manager.
- **Listeners:** To enable this channel communication, a “listener” must be defined on the receiving queue manager. This listener detects incoming network requests and must be active for the receiver channel to activate in response to a sender channel startup request.
- **Queues:** A queue is simply a resource that is defined to retain received messages until they are read or the messages expire.



The topology we have defined to support this integration effort is represented in Figure 3. As you can see, each system to be integrated has its own WMQ installation. Even though WMQ allows applications to connect to a queue manager as a “client” (a remote connection), we have chosen to utilize local or server connections for our EAI effort for the following reasons:

- Global transaction coordination is not supported over a client connection.
- A client connection approach can result in the application being unsure its WMQ request was successfully processed. This could occur if the connection to the queue manager was lost before a response was received.
- Local/server connections obviously offer performance advantages over client connections.
- Local/server connections are required if WebSphere Application Server is to act as the transaction coordinator.

Definition of the WebSphere Business Integration Message Broker Environment

WebSphere Business Integration Message Broker leverages and extends the messaging infrastructure by providing a business rules-based message brokering facility that includes:

- Message routing (to one or more destinations) based on the application of business rules and/or message content
- Message transformation to allow systems to exchange messages with other systems using incompatible message formats

As with WMQ, a complete description of how to install and configure WebSphere Business Integration Message Broker is beyond the scope of this document. There are, however, some concepts that you should be familiar with:

- **Broker:** A WebSphere Business Integration Message Broker resource that is the host for your EAI business logic and processes. A broker must be associated with a queue manager. It uses a persistent store to manage its data. (We used IBM's DB2, although other database vendor products could be used.) The broker connects to the database via ODBC (in the Windows 2000 version of the product). The broker is the hub of your EAI effort, allowing you to decouple all the point-to-point interfaces that bind your legacy applications together and move to a “hub-and-spoke” connection strategy.
- **Configuration manager:** Provides central management functions for your broker domain. It is used to maintain broker configuration information and manages the deployment of message flow-processing objects. It also uses a database to maintain its configuration information. (Again, we used DB2.)
- **Control center:** Provides the user interface (GUI) to the configuration manager functions.

The additional WebSphere Business Integration Message Broker objects have been defined to use the BRKR queue manager, as shown in Figure 4.

Configuring the EAI Business Rules

To get started configuring your broker, first make sure the MQSeries service (on Windows 2000) is started. Then start the MQSeries Broker ConfigMgr and the MQSeries Broker BRKR services. Once all are running, we can begin using the Control Center. To begin using the Control Center, click Start -> Programs -> IBM WebSphere Integrator 2.1 -> Control Center. The first time you run this program, you will

be prompted to fill in the Host Name (the server the broker process is running on), Queue Manager Name (associated with the broker when it was created), and the Port (the listener associated with the queue manager).

After the main screen opens, select the tab labeled Message Flows. This is where we will define a message flow, build the routing logic using IBM primitive nodes, and deploy the flow to our broker. First, a brief description of the terms:

- **Message flow:** A series of processing steps defining the routing and transformation of messages between queues using provided IBM primitive nodes and node connections.
- **IBM primitive nodes:** Basic processing units that allow the input/output of messages, database access and update, message transformation, aggregation of messages, and other process-related logic. You can see the list in Figure 5. Each node has different configuration parameters, such as queue manager name, queue name, transactional support, message type, etc. Certain nodes allow the use of ESQL.
- **ESQL:** A language derived from SQL. It is used to manipulate both message and database data. An ESQL program is simply a series of statements that are executed in the order they're written. A complete reference to the ESQL language is provided with WebSphere Business Integration Message Broker.
- **Node connections:** The "lines" you draw between the primitive node input and output terminals, defining the routing of the message from node to node.

Figure 5 represents the message flow we created for the profiling update process. The process is as follows:

1. Capture the profile update request from the profiling system using the MQInput node. Message processing flows always start with this type of node. It performs an MQGET to read the message from the queue name you defined in the node properties. We have configured it to accept XML messages, to read from the PROFILE.REQUEST.INPUT queue, and renamed it "ProfilingRequest" to assist in flow documentation. The input XML document is shown in Listing 1.
2. Propagate the request to the appropriate systems, using metadata obtained from the XREF database table. This is achieved through the use of a compute node, which allows the construction of an entirely new message based on the input message, modification of a message (headers, header fields, body data), and enhancement of message body data using information retrieved from databases. In our example, we used the Profiling key to retrieve corresponding target system keys from the database and appended that data to the message. The message is then routed to other nodes downstream for processing. Figure 6 shows the ESQL used to retrieve the data and create a different outbound message. (Only the inbound message headers were copied into the outbound message.)
3. Before the message is routed to any legacy system, it must be transformed into a format the target system can understand. For this example, we will transform the message into various XML formats. (Fixed-record and delimited-record formats can also be generated.) Figure 7 represents transformation for the reservation system as an example, with the XML structure changed and an output field created from two input fields. Once again, we use a compute node to perform the transformation.

That concludes our basic message transformation and propagation example. All that remains is to assign the flow to our BRKR broker and deploy it using the Control Center.

Design of Profiling System JMS Functionality

By now, it should be clear that we intend to use JMS to support messaging applications in J2EE environments. JMS is composed of four main components. These are JMS clients, non-JMS clients, messages and providers. Clients are the Java programs that send and receive messages. Non-JMS clients are the legacy systems that use the native provider APIs. A message is the data that is communicated between clients and nonclients. Providers are the actual messaging systems.

A JMS application can use one of two messaging styles. These are referred to as the point-to-point application model and the publish-subscribe application model. For the purposes of this article, we will use the point-to-point application model. This model consists of several interfaces that are required for a successful implementation:

- **QueueConnectionFactory:** Used to create QueueConnections
- **QueueConnection:** The interface that connects to the point-to-point provider APIs
- **Queue:** Encapsulates the provider-specific queue name
- **QueueSession:** The interface used to create QueueSenders and QueueReceivers

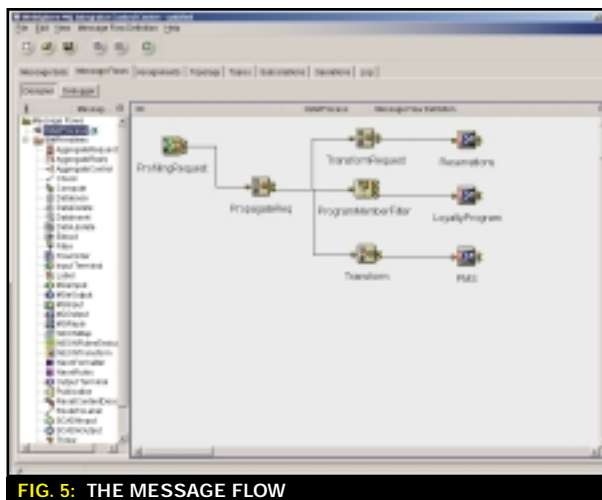


FIG. 5: THE MESSAGE FLOW

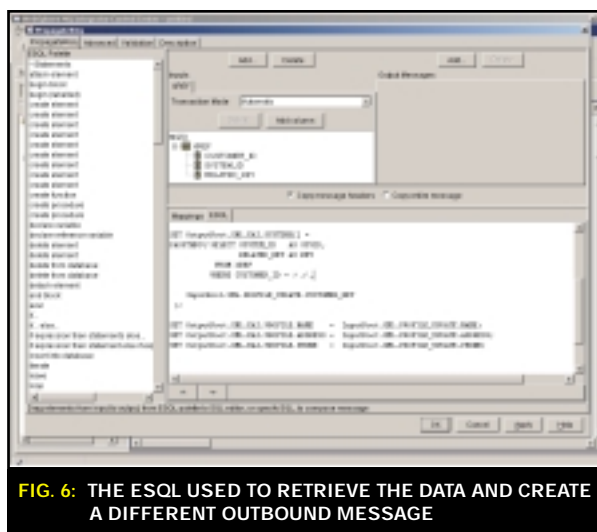


FIG. 6: THE ESQL USED TO RETRIEVE THE DATA AND CREATE A DIFFERENT OUTBOUND MESSAGE

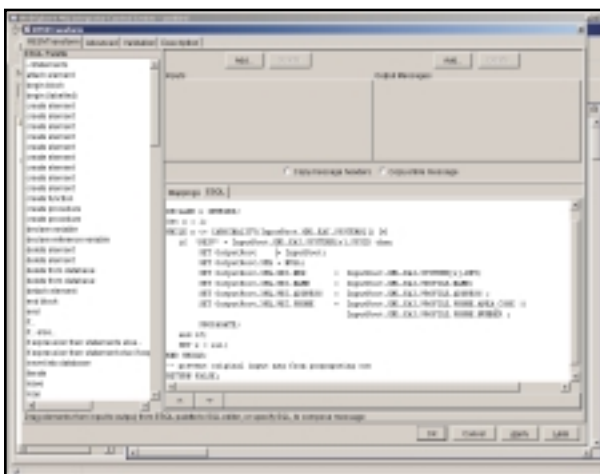


FIG. 7: TRANSFORMATION FOR THE RESERVATION SYSTEM

- **QueueSender:** The interface used to send messages to a queue
- **QueueReceiver:** The interface used to receive messages from a queue

In addition to JMS, we have implemented the ServiceLocator API recommended in the Sun blueprints. This API encapsulates JNDI lookups. The example code (see Listing 2) will use the ServiceLocator implementation for JNDI lookups. For more information on ServiceLocator, refer to the following link: <http://java.sun.com/blueprints/code/jps131/src/com/sun/j2ee/blueprints/servicelocator/ejb/ServiceLocator.java.html>.

The first step to setting up our example is to create the ConnectionFactory. This is the main controller of the JMS model. The connection factory enables the creation of queue connections. Next, we will need to create the queue receiver and queue sender queues. As seen in the code example, the ServiceLocator class encapsulates these methods. This is because each requires a JNDI lookup to access the individual class. Once we have completed our JNDI lookups we can use the classes to access the JMS APIs.

The next step is to create a QueueConnection. This interface enables the JMS application to communicate to the provider's point-to-point APIs. After we have created the QueueConnection we can create a session that will enable the creation of the QueueSender

Interface, which is responsible for sending messages to the queue. This example has encapsulated the JMS functionality into an implementation class that could be accessible from a session bean or a Web component.


Legacy System Integration

The final step in our project is to integrate the legacy applications. As we illustrated in Figure 3, this requires the installation of the WMQ software into each system's environment, and the definition of the WMQ queues and channels needed to connect the servers hosting the various legacy systems to the WebSphere Business Integration Message Broker server.

The details concerning the design of the application programs used by each legacy system to access the message queues are outside the scope of this article. In general terms, each legacy program written to process incoming messages should provide the following functions:

- **Read message queue for received messages:** The specifics of how to do this depend on the programming language and platform the system is running on. Refer to the WMQ documentation for further information.
- **Parse message, if required:** Since WebSphere Business Integration Message Broker allows for the use of both XML messages and "fixed-format" message types, this step may be optional.
- **Process update to the system:** This can be done via direct updates to database tables or by invoking legacy programs. Obviously, we want to leverage existing business rules, so calling legacy programs directly or inserting the message data into some sort of staging table to be processed later by existing batch processes is desirable.

Conclusion

Once we have completed the coding of legacy components, we have achieved a level of integration that replaces any existing point-to-point batch interfaces, and provided the additional benefits outlined earlier. Now, because we have implemented WebSphere Business Integration Message Broker, the business rules for integration are centrally located and easily modified. In addition, this integration insulates the legacy systems from further code changes. 

LISTING 1

```
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<PROFILE_UPDATE>
  <CUSTOMER_KEY>100</CUSTOMER_KEY>
  <NAME>
    <FIRST_NAME>BOB</FIRST_NAME>
    <LAST_NAME>JONES</LAST_NAME>
  </NAME>
  <ADDRESS>
    <NUMBER>600</NUMBER>
    <STREET>PALM BLVD</STREET>
    <CITY>FAIRFAX</CITY>
    <STATE>VA</STATE>
    <ZIP>22030</ZIP>
  </ADDRESS>
  <PHONE>
    <AREACODE>703</AREACODE>
    <NUMBER>5551212</NUMBER>
  </PHONE>
</PROFILE_UPDATE>
```

LISTING 2

```
import javax.jms.JMSEException;
import javax.jms.Message;
import javax.jms.Queue;
import javax.jms.QueueConnection;
```

```
import javax.jms.QueueConnectionFactory;
import javax.jms.QueueSender;
import javax.jms.QueueSession;
import javax.jms.Session;
import javax.jms.TextMessage;
import java.text.SimpleDateFormat;
import java.util.Date;
import
sun.j2ee.blueprints.servicelocator.ServiceLocatorException;
import sun.j2ee.blueprints.servicelocator.ejb.ServiceLocator;

/**
 * The MessageService is used to handle all Messaging
 * requests.
 */
public class MessageService {

    private ServiceLocator locator;

    /**
     * MessageService constructor sets up the Service
     * Locator
     */
    public MessageService () throws MessageServiceException
```

```

{
    try {
        this.locator = new ServiceLocator();
    } catch (ServiceLocatorException ex) {
        throw new MessageServiceException(key, ex);
    }
}

/**
 * Submit an update of profile to external systems
 *
 */
public synchronized String updateProfile(Customer customer,
String[] legacySystem) throws MessageServiceException {

    // Initialize local variables
    String request = null;
    String response = null;
    String connectionFactory =
        "java:comp/env/jms/connectionFactory";
    String updateRequestDestination =
        "java:comp/env/jms/updateProfileRequest";
    String updateReponseDestination =
        "java:comp/env/jms/updateProfileResponse";

    // Prepare the Request XML String
    request = generateUpdateRequestXML(customer);

    response = processUpdate( connectionFactory,

        updateRequestDestination,
        updateReponseDestination,
        request );

    return response;
}

/**
 * Send an update Message -
 *
 */
private String processUpdate( String    connectionFactoryName,
    String updateRequestDestination,
    String updateReponseDestination,
    String messageText)

    throws MessageServiceException {

    String reply = null;
    QueueSession queueSession = null;
    QueueSender queueSender = null;
    QueueConnection queueConnection = null;

    try {
        // First get the QueueConnectionFactory
        QueueConnectionFactory queueConnectionFactory = loca
            tor.getQueueConnectionFactory(connectionFactoryName);

        // Next get the Request(SendQueue)
        Queue requestQueue =
            locator.getQueue(updateRequestDestination);

        // Next get the Reply(ReceiveQueue)
        Queue replyQueue =
            locator.getQueue(updateReponseDestination);

        // Get the Connection
        queueConnection =
            queueConnectionFactory.createQueueConnection();

        // Get the Session
        queueSession = queueConnection.createQueueSession
            (false, Session.AUTO_ACKNOWLEDGE);

        // Create the Sender
        queueSender = queueSession.createSender(requestQueue);

        // create a text message from the message that was
        passed in

        Message reqMessage = queueSession.create

```

```

        TextMessage(messageText);

        // Send the message
        queueSender.send(reqMessage);

    }

} catch (ServiceLocatorException ex) {
    String key = this.getClass().getName() ;

    throw new MessageServiceException (key, ex);
} catch (JMSEException ex) {
    String key = this.getClass().getName() ;

    throw new MessageServiceException (key, ex);
} finally {
    try {
        queueConnection.close();
        queueSession.close();
        queueSender.close();

    } catch (JMSEException ex) {
        // do something nice here
    }

}

// Return response
return reply;
}

/**
 * Create the UpdateCustomer request XML message text
 *
 */
private String generateUpdateRequestXML( Customer customer,
    throws MessengerException {

    StringBuffer xml = new StringBuffer();
    xml.append("<?xml version='1.0' encoding='UTF-8'
        standalone='yes'?"");
    xml.append("<PROFILE_UPDATE>");
    xml.append("<CUSTOMER_KEY>");
        xml.append(Customer.getCustomerKey());
    xml.append("</CUSTOMER_KEY>");
    xml.append("<NAME>");
        xml.append("<FIRST_NAME>");
            xml.append(Customer.getName().getFirstName());
        xml.append("</FIRST_NAME>");
        xml.append("<LAST_NAME>");
            xml.append(Customer.getName().getLastName());
        xml.append("</LAST_NAME>");
    xml.append("</NAME>");
    xml.append("<ADDRESS>");
        xml.append("<NUMBER>");
            xml.append(Customer.getAddress().getNumber());
        xml.append("</NUMBER>");
        xml.append("<STREET>");
            xml.append(Customer.getAddress().getStreet());
        xml.append("</STREET>");
        xml.append("<CITY>");
            xml.append(Customer.getAddress().getCity());
        xml.append("</CITY>");
        xml.append("<STATE>");
            xml.append(Customer.getAddress().getState());
        xml.append("</STATE>");
        xml.append("<ZIP>");
            xml.append(Customer.getAddress().getZip());
        xml.append("</ZIP>");
    xml.append("</ADDRESS>");
    xml.append("<PHONE>");
        xml.append("<AREACODE>");
            xml.append(Customer.getPhone().getAreaCode());
        xml.append("</AREACODE>");
        xml.append("<NUMBER>");
            xml.append(Customer.getPhone().getNumber());
        xml.append("</NUMBER>");
    xml.append("</PHONE>");
    xml.append("</PROFILE_UPDATE >");

    return xml.toString();
}

```


A skin is more than just a matter of appearances

Skin Deep

BY MARCEL HEIJMANS



ABOUT THE AUTHOR

Marcel Heijmans is a senior software engineer and founder of Mnemonics. He created the "J2EE Development Coaching" concept, which trains and supports novice developers and architects within their projects while minimizing the project risks.

E-MAIL

marcel.heijmans@mnemonics.nl

The look and feel of a WebSphere Portal site is determined by the definition of themes and skins. This might give you the impression that themes and skins are mere window dressing and hence the domain of interaction designers and graphic designers. The WebSphere Portal page renderer, however, constructs its portal pages on the logic defined in the JavaServer Pages (JSPs) of the themes and skins. This means that themes and skins are capable of more function than just determining the look and feel of a portal page through cascading style sheets and images. WebSphere Portal developers can utilize the themes and (in particular the) skins to implement functionality that will influence the behavior of the portlet container (window and title bar) without modifying the portlets involved.

Themes and Skins

The visual appearance and the navigational structures of all the portal pages within a "Place" are determined by a *theme*. Themes determine the overall look and feel of the portal, including colors, images, and fonts, which will ensure consistency and provide company-specific branding. A theme supports multiple skins. A *skin* defines the look and feel of the window around the individual portlets (see Figure 1). Each portlet on a page can have its own skin, configured by the Portal administrator.

The themes and skins are the main parts of the Portal page construction process. They can be found in the directory WebSphere/PortalServer/app/wps.ear/wps.war and the more specific subdirectories. During the construction of the page, the portal server searches for theme and skin

components, starting with the most specific subdirectory (country, locale, and client) and moving up to the more general, higher-level directories.

For example, when a user requests a page from a client using Internet Explorer version 5 with the locale set to en_US and the users' skins set to Shadow, WebSphere Portal will search for the skin resource file Control.jsp in the following order:

1. /skins/html/Shadow/ie5/en_US/Control.jsp
2. /skins/html/Shadow/ie5/en/Control.jsp
3. /skins/html/Shadow/ie5/Control.jsp
4. /skins/html/Shadow/en_US/Control.jsp
5. /skins/html/Shadow/en/Control.jsp
6. /skins/html/Shadow/Control.jsp
7. /skins/html/ie5/en_US/Control.jsp
8. /skins/html/ie5/en/Control.jsp

9. /skins/html/ie5/Control.jsp
10. /skins/html/en_US/Control.jsp
11. /skins/html/en/Control.jsp
12. /skins/html/Control.jsp
13. /skins/Control.jsp

Portal Page Construction

Figure 2 depicts the internal flow of the Portal page construction. Every page within WebSphere Portal is built according to this JSP scheme, except for some specific screens such as the registration, login, and error pages.

The starting point of the Portal page construction process is Default.jsp in the Themes directory. This page includes the other JSPs of the theme. Together these theme JSPs compose the header and navigation (places and pages) parts of nearly every WebSphere Portal page. Modifications to the Portal header, toolbar, place bar, or page bar are made within the corresponding JSP file. At the bottom the Default.jsp holds the <wps:screen Render> tag, which will include the Home.jsp of the Screen directory. The Home.jsp represents the content of a Portal page and triggers the composition render process of the skin-related JSPs.

Control.jsp

The portlet render process is executed in the Control.jsp page of the skin. This means that every portlet is rendered according to the Control.jsp defined by the portlet skin for each page request. Interaction designers and graphic designers modify the images and cascading style sheets that are used by the Control.jsp to determine how a portlet (window) looks. The Control.jsp itself is less frequently altered, because it defines the title bar icon placement (title, minimize, maximize, etc.) of the portlet window (see Figure 3). The <wps:portletRender> tag will be replaced by the actual content of the corresponding portlet.

WebSphere Portal provides a tag

library (enige.tld) that is used in the JSPs of the themes and skins to provide the construction and decision logic. These tags can somehow determine which portlet is to be rendered. Consequently, the portlet is accessible from within the Control.jsp before it is rendered. This means that it is possible to create skin-based portlet window behavior without modifying portlet code. Since many portlets are provided by third parties (www.ibm.com/websphere/portal/portlet/catalog), no source code is available. Being able to change the behavior of portlets by selecting a custom skin (Control.jsp) offers a few interesting opportunities. The following example shows the type of solutions that can be built with skin-based portlet functionality, and how to create them with WebSphere Studio and the Portal Toolkit plug-in. The source code (included in the WAR files) can be downloaded from www.sys-con.com/websphere/sourcecfm. The example code was tested on WebSphere Portal 4.1 and 4.2.

Personalization and Authorization

One of the core features of WebSphere Portal is the configuration (Configure mode) and personalization (Edit mode) of portlets. The portlet developer determines what is to be configured by the Portal administrators and what is to be personalized by a portlet user. Extending the configuration or per-

sonalization of portlets requires extra development per portlet and of course the source code must be available. The latter is often a problem for third-party portlets, not to mention decompilers.

Other related Portal features are access control (security/authorization) and custom pages. The Portal administrator determines which Places, Pages, and Portlets are accessible by which users and groups. When given the proper permissions, a user can edit his or her personal pages, determining which portlets should be on the page and their placement.

Now suppose that an extra feature is needed, functionality that provides conditional rendering (i.e., showing or hiding a portlet) based on a user profile setting. Personalization is not the solution because it controls the behavior of the portlets' content and not the window. Once the doView() method is invoked it is too late to hide a portlet, because parts of the portlet window (the title bar and border) are already created by the Control.jsp; however, skipping the doView() would render an empty portlet window. Neither is access control applicable. The authorization mechanism of WebSphere Portal is not advanced enough to handle conditional authorization based on user settings.

Conditional Rendering

If a Portal requires conditional portlet rendering, the best place to intervene is where the portlet window and content are (about to be)

rendered. Hence, the Control.jsp of the selected skin is the best place to handle this kind of requirement because it contains the portlet window (title bar and border) and the <wps:portletRender> tag. This solution involves no modification of the portlet itself. The Portal administrator can just add an extra parameter to the portlet settings of all portlets that require custom rendering. The Control.jsp can access the parameters before portlet rendering and provide the required behavior. Because portlet parameters can be added, modified, and deleted at runtime by the Portal administrator, it is the ideal way to trigger specific skin functions.

Making rendering dependent on a user profile means that a personalized profile setting is exploited. In this example, the preferred language setting of the user is utilized. Of course, any other user profile setting – including a custom setting – is viable. The preferred language was chosen for reasons of simplicity (it is the default for WebSphere Portal installation). Now the modified Control.jsp compares the user profile setting with the portlet parameter and renders it when there is a match (see Figure 4).

The user information is available in the Control.jsp by invoking the getUser() method on the RunData



FIG. 1: THEMES AND SKINS IN WEBSPHERE PORTAL

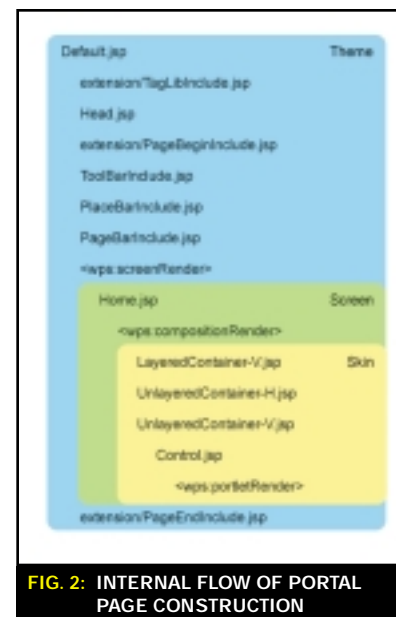


FIG. 2: INTERNAL FLOW OF PORTAL PAGE CONSTRUCTION

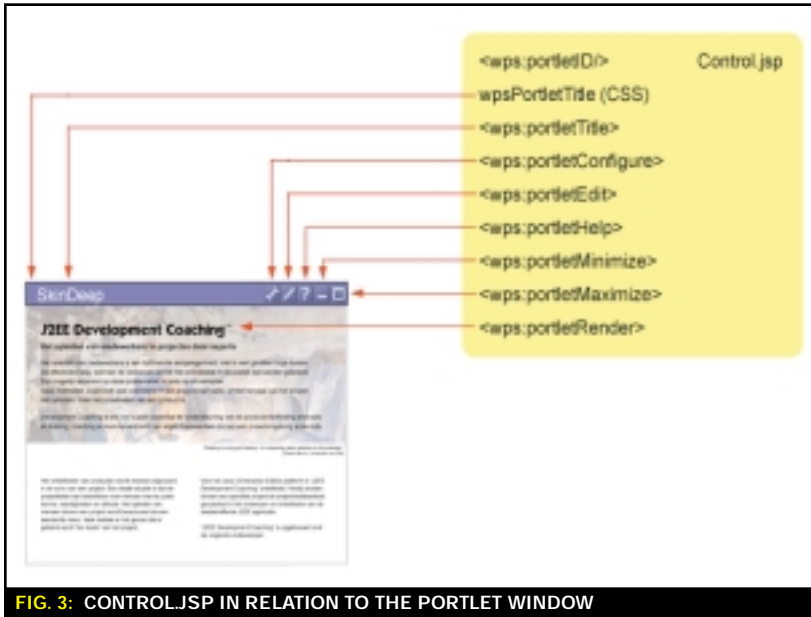


FIG. 3: CONTROL.JSP IN RELATION TO THE PORTLET WINDOW

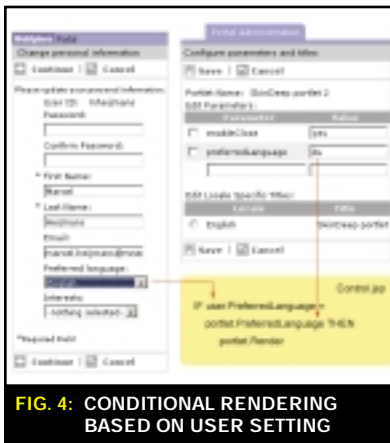


FIG. 4: CONDITIONAL RENDERING BASED ON USER SETTING

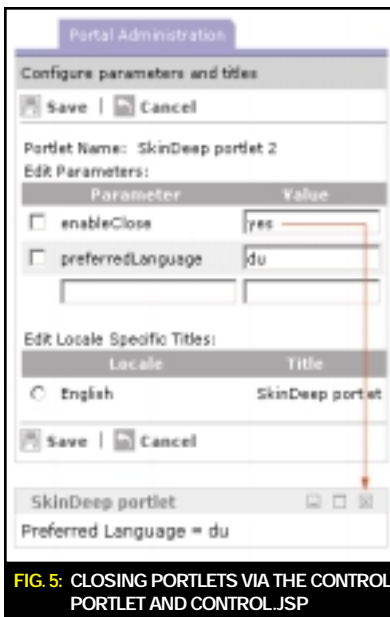


FIG. 5: CLOSING PORTLETS VIA THE CONTROL PORTLET AND CONTROL.JSP

object. `RunData` is an interface to run-time information that is passed within WebSphere Portal (actually it is JetSpeed, or more accurately, Turbine). The `com.ibm.wps.engine.RunData` can be retrieved from the request object. Acquiring portlet parameters in the `Control.jsp` is more complex. First, the `PortletDescriptorID` that identifies the portlet to be rendered is retrieved from the `PortletHolder`. The `PortletHolder` is available in the `RunData` through an attribute (name differs per WebSphere Portal version). Finally, with the `PortletDescriptorID` the `ConcretePortletEntry`, which holds the portlet parameters (as a map), can be found in the `PortletRegistryAccess`.

For technical details, refer to the `Control.jsp` of the source code zip file. Be careful to check the WebSphere Portal version, because there are differences between the `Control.jsp` (Portlet API) of WPS 4.1 and WPS 4.2. These differences are clearly marked by comments within the `Control.jsp` source files.

PortletSession vs HttpSession

In this example the conditional rendering can be activated or deactivated with an additional `Control` portlet that is available in the WAR file. This `Control` portlet also

demonstrates how to communicate between a portlet and the `Control.jsp` using session variables. This might seem trivial, but there is a catch. A portlet can retrieve the session by calling `getPortletSession()` from the request object. However, this `PortletSession` is confined to a subset of the `HttpSession` used by this portlet. It is hard to retrieve (Portlet) session variables in a JSP page because the variable names are prefixed with a portlet ID. Here is a trick to circumvent this problem: cast the `PortletSession` to a `PortletSessionImpl` and use the `getServletSession()` method of this class to obtain the complete `HttpSession`. Be aware that this is outside the published Portlet API.

Closing Portlets

Although available in the (JetSpeed) Portlet API and a valid value according to the DTD, the portlet `Close` function is not supported by WebSphere Portal. `Control.jsp`, in conjunction with the `Control` portlet, can offer this close functionality. The Portal administrator again needs to add an extra parameter (`enableClose`) to all portlets that require closing. The `Control.jsp` can now display the `Close` icon in the title bar of the portlets with this parameter set to "yes". When a user clicks the `Close` icon, an action event is sent to the `Control` portlet, which sets a session variable to indicate that the corresponding portlet must be closed. Closed means that the `Control.jsp` does not render the portlet (and the window), which also implies that the portlet content is not aggregated (see Figure 5).

The `Control` portlet has an extra function that can restore all closed portlets to normal again (see Figure 6). For technical details, refer to the `Control.jsp` and `SkinDeepControl.java` of the source code WAR file.

Conclusion

Using the skin's `Control.jsp` to provide specific user interaction functionality with respect to portlet rendering is a solution that will work for all portlets, without any

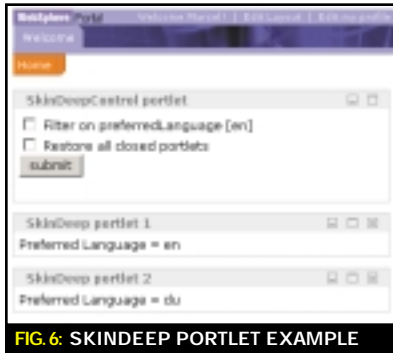


FIG. 6: SKINDEEP PORTLET EXAMPLE

modification to the portlets involved. Although this is a big advantage, there are limits to the applicability of a pure Control.jsp solution. Sometimes a combination of special portlets (Control portlet) and a custom Control.jsp is necessary to solve specific problems.

There are also certain drawbacks attached to the advanced Control.jsp modifications described here, the most important one being the Portlet API differences between the versions

of WebSphere Portal (4.1, 4.2, and future versions). Another shortcoming is that the Java code in the Control.jsp can be visible to graphic designers and may make them uncomfortable while modifying the look and feel of the portlet window. It is also error prone because of the imports in the page directive. Nine out of ten times when there is an internal translation/compilation problem with the Control.jsp the problem is a missing import. The solution to all these problems is to use custom tags that implement the Java part of the Control.jsp. With custom tags only the tag library needs to be modified in case of a Portlet API revision, the HTML tags are not cluttered by Java code, and there are no imports, except for the inclusion of the tag library.

The example presented here shows a few possible functions that can be implemented with a custom skin (Control.jsp). Of course there are

many more applications in which the use of a custom skin, possibly in combination with special portlets, can be a (partial) solution.

Customizing the skin might be a good answer, especially when specific portlet window user interaction or particular portlet rendering is required.

References

- *IBM WebSphere Portal V4.1 Handbook Volume 2, IBM Redbook, SG24-6920-00:* <http://publib-b.boul.der.ibm.com/Redbooks.nsf/CDAbstracts/sk3t8282.html>
- *Developing and Debugging Portlets Using the IBM Portal Toolkit Plug-in for WebSphere Studio Application Developer:* www7b.software.ibm.com/wsdd/library/techarticles/0210phillips/phillips.html
- *Creating a Distinctive Look and Feel for Your Portal:* www7b.software.ibm.com/wsdd/library/techarticles/0307bartek/bartek.html





SEPT. 30 - OCT. 2, 2003

REGISTER TODAY!

CALL 201-802-3058

www.sys-con.com/edge

Register before September 26th and

SAVE \$100



FREE IBM TUTORIAL

WHEN YOU REGISTER FOR A VIP PASS!

Guarantee your seat when you register for a Full Conference Pass

How to Develop, Deploy, and Manage Web Services Using IBM Tools
September 30, 2003

Are you a developer, product manager, or software architect interested in learning how to develop, deploy, and manage Web services? If so, this technical seminar – in which IBM experts will review the standards initiatives behind some of these technologies, the latest developments and their future roadmap – is for you!

Course Highlights

- Real-world implementation
- Developing and deploying SOAP-enabled Web services
- Registry operations and programming with UDDI4J version 2
- IBM WebSphere Studio Application Developer
- RAS and Web services
- Web services stack and WebSphere Application Server version 5.0
- Emerging Web services technologies - WSFL, WSIL, WSUI, etc.
- Developer support - IBM Web Services Toolkit version 3.0, resources, tools, products, Web sites, Business Partner support, education, etc.

Prerequisites

Participants should have basic skills in Java technology and HTML, and experience in core XML technology.

*Seating is limited.
Full Conference Attendees will receive Priority Seating for all Tutorials; all other seating is on a first come, first served basis.

PRODUCED BY
SYS-CON
EVENTS

The ValueListHandler design pattern improves J2EE application performance

Dealing with Large Database Result Sets

BY LLOYD HAGEMO AND
RAVI KALIDINDI



ABOUT THE AUTHOR

Lloyd Hagemo is a senior director for Candle Corporation's Application Infrastructure Management Group. He is responsible for WebSphere tools development. Lloyd has led the successful development of more than 20 products for the WebSphere environment, including operating system utilities, network performance and tuning products, WebSphere MQ configuration and management tools, and application integration solutions.

E-MAIL

lloyd_hagemo@
candle.com

When designing J2EE (Java 2 Platform, Enterprise Edition) applications, developers often find themselves challenged to create a display for large database result sets. Improper treatment of the large result set display can lead to poor response time and, ultimately, lost productivity and sales.

Developers must be cautious when they deal with large results, particularly when they choose entity beans as a data access layer. Based on our experience, entity beans negatively impact performance when returning large result sets, due to the Enterprise JavaBean (EJB) container's extensive built-in capabilities, such as transactional controls, security, resource pooling, remote interface, thread safe, and database synchronization. The use of a Data Access Object (DAO) along with Java Database Connectivity (JDBC) is the best practice for managing large database result sets while maintaining a high level of performance. The DAO JDBC queries can be written relatively easily to retrieve select records based on user requests.

This column discusses strategies and tools for designing an application that properly handles large database result sets and effectively displays them to end users. We will focus on the use of a specific design pattern, the ValueListHandler, as an indispensable tool for addressing the challenge of displaying large database results. The ValueListHandler design pattern is used to control the size and

caching of the result set returned to the client application while avoiding the resource overhead required to process data using J2EE entity beans.

A Case Study

We begin with a case study. A retailer wanted to develop an online catalog containing several thousand products. As part of the initiative, the retailer wanted to create a standard storefront application from which customers could browse the catalog, select items to place in a shopping cart, and, ultimately, purchase the items selected. The retailer estimated that more than 200 users could be browsing the catalog at any given time.

The development team used Container Managed Persistence (CMP) entity beans to enable database access to the catalog, inventory items, customer information, shopping cart, and order form. Figure 1 shows an overview of the application architecture.

The system was easy to develop using tools that generate the database interfaces and require little or no Java coding. The entity bean deployment descriptors were built to match the

database schemas and then modified to support the "get-and-set" data access methods. As recommended by design best practices, all of this was wrapped in session beans to provide a solid verb-noun relationship – that is, a solid relationship between the session bean providing the transaction business logic and the entity bean providing the data content. The development team also implemented the user interface via a servlet with JavaServer Pages (JSPs) to display the results. This methodology allowed the retailer to bring the application online quickly.

The Web opened the retailer to a broader market, which increased the company's visibility and created heavy in-store traffic. All was not well, however, with the online outlet. Customers soon began to experience poor response times. In addition, the system's throughput performance was unacceptable. The retailer traced the problems to the database implementation, which used CMP entity beans for reading the catalog. The following XML code was used to retrieve data from the catalog.

```
<query>
<query method>
  <method-
    name>findAllProducts</
    method-name>
    <method-parms>
    </method-parms>
  </query-method>
<ejb-ql>
<![CDATA[SELECT * FROM
  Catalog]]>
    </ejb-ql>
  </query>
```

This implementation resulted in the entire product catalog being read for each user. Each "enter" resulted in several thousand "reads" of the catalog records. The system used the same methodology to serialize catalog records for transfer back to the servlet and to display a subset of records on the user's terminal using HTML.

The application architecture had several other deficiencies. The database implementation, however, was by far the most serious problem. It was the root cause for the poor response time and excessive resource consumption by the application server.

ValueListHandler Pattern Implementation Value

J2EE applications must be designed to handle database-processing requests that have the potential to return large quantities of data. As important, access to these lists is usually bidirectional, and the records are often discarded after use.

In this pattern, the term “ValueList” represents a collection class that holds a set of database records. The term “Handler” represents a class that iterates the ValueList collection. Accessing a large catalog represents an ideal situation for utilizing the ValueListHandler design pattern, which was created to provide a more efficient way to manage large lists. The design pattern is used to cache and provide results to the client using a lightweight mechanism. Rather than returning the entire list, the design pattern retrieves only a small subset of data. As a result, only data specific to the user’s request is retrieved and transferred to the client.

Listings 1 and 2 present a simple ValueListHandler implementation that uses simple Java classes and JDBC to read a SQL database. The ValueListHandler class implements

the HandlerIterator interface. Listing 1 provides the code for the HandlerIterator interface, and Listing 2 shows the code for ValueListHandler class.

In a production application, a developer would need to extend this implementation to meet specific application requirements. The ValueListHandler design pattern is architected to access the database directly from the Web container. This design pattern uses servlets and JSPs instead of EJB entity beans to read and return large catalog list result sets.

Figure 2 shows the ValueListHandler pattern and its dependent classes. ValueList is a collection that holds the results, which can be cached to avoid unnecessary database access. The ValueList represents each record in the result set. A DAO is used to encapsulate the JDBC code that reads the catalog records and returns the results to the ValueList object. The ValueList represents each record in the result set. The DAO class gets the data from the database and sets the results into a ValueList object using a setList(List list) method. From that point onward, enterprises simply need to iterate through the ValueListHandler class, using getNextElements(int count) and getPreviousElements(int count).

The green section in Figure 2 represents the Java classes for the ValueListHandler design pattern. The classes in this example are imple-

mented in the Web container. The ValueListHandler will iterate through the ValueList, returning the exact number of database records required to fill the user’s request based on the browser’s page size. A decision is made by the ValueListHandler based on the display count integer to call the DAO to retrieve additional database records. This strategy eliminates all serialization and network traffic between the Web and EJB containers. The end result is improved application performance.

It is important to note that the ValueList object is not shared with other user sessions. Each session has its own copy of the catalog entries it is browsing. Therefore, to reduce memory overhead, the design pattern can be modified to limit the number of result sets held in the ValueList object. While this approach reduces the memory overhead within the Web container, it also reduces the efficiency of this design pattern. The strategy you select will depend on your performance requirements, the available resources, and the number of users accessing the system.

Conclusion

In the case study the customer used a set of design patterns and J2EE application server capabilities to implement a Web-based storefront application. A number of application development books use similar examples to illustrate the capabilities of a J2EE application. It is important, however, to remember that the development tips included in most books serve as learning tools and rarely reflect the variables and considerations associated with real-world business requirements. When faced with production requirements, application architecture must provide high performance and deliver the flexibility to respond to application changes.

We believe that the ValueListHandler enables J2EE developers to quickly search catalog categories and return lists of database records to users as required. The design pattern improves the performance of database operations that return large result sets by reducing resource

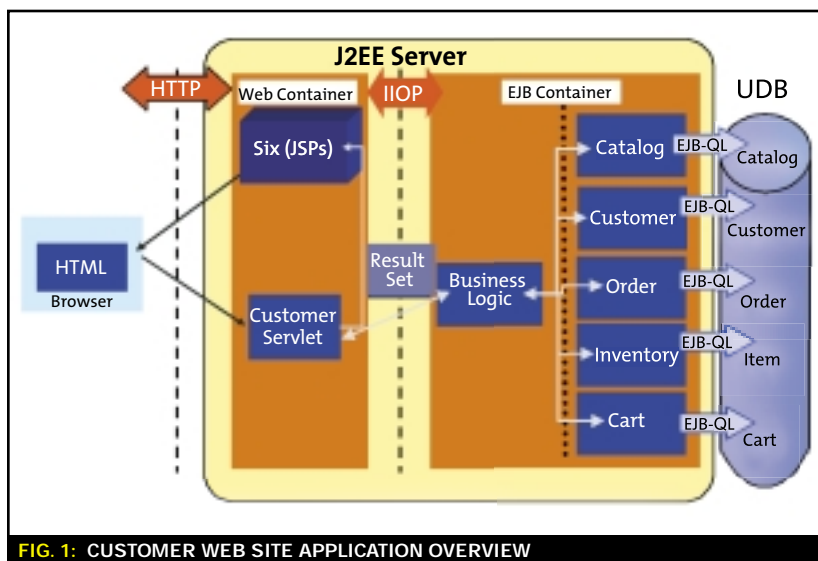


FIG. 1: CUSTOMER WEB SITE APPLICATION OVERVIEW



ABOUT THE AUTHOR

Ravi Kalidindi is a senior software engineer in Candle's Application Infrastructure Management Group. Ravi has worked with Java since its inception and has published several articles that focus on J2EE best practices.

E-MAIL

ravi_kalidindi@candle.com

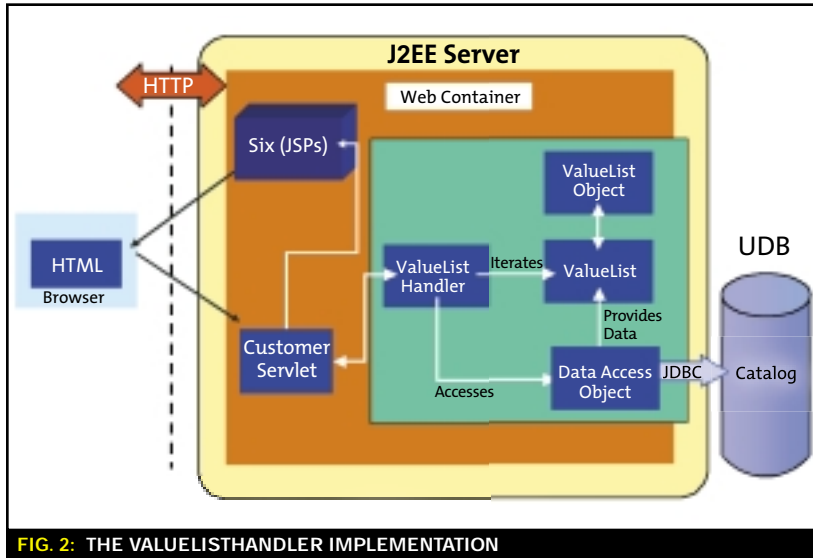


FIG. 2: THE VALUELISTHANDLER IMPLEMENTATION

contention and network traffic. In addition, the ValueListHandler pattern provides a Java class that can act as a single point of control for the client as the user iterates through catalog entries. This provides a common process for developers to make changes within the application to meet new requirements. Most important, the ValueListHandler design pattern meets the retailer's requirements for a Web-based shopping solution that responds quickly to users' catalog lookup requests.

Reference

- *The Source for Developers:* www.sun.com/developers/blueprints

LISTING 1: HANDLERITERATOR INTERFACE CODE:

```
import java.util.List;
public interface HandlerIterator {

    public int getSize() throws ValueListHandlerException;
    public List getPreviousElements(int count) throws
        ValueListHandlerException;
    public List getNextElements(int count) throws
        ValueListHandlerException;

}
```

LISTING 2: VALUELISTHANDLERCLASS CODE:

```
import java.util.*;
public class ValueListHandler
implements HandlerIterator {

    protected List list;
    protected ListIterator listIterator;

    public ValueListHandler() {
    }

    protected void setList(List list)
    throws ValueListHandlerException {
        // validate
        if(list == null)
            throw new ValueListHandlerException("List is
            null");

        this.list = list;
        listIterator = list.listIterator();
    }

    public List getList(){
        return list;
    }

    public int getSize() throws ValueListHandlerException{
        // validate
        if(this.list == null)
            throw new ValueListHandlerException("List is
            null");

        int size = 0;
```

```
size = list.size();
return size;
}
```

```
public List getPreviousElements(int count) throws
ValueListHandlerException{
    // validate
    if(this.list == null)
        throw new ValueListHandlerException("List is
        null");
```

```
int i = 0;
Object object = null;
ArrayList elements = new ArrayList();
if (listIterator != null) {
    while (listIterator.hasPrevious() && (i < count)){
        object = listIterator.previous();
        elements.add(object);
        i++;
    }
}
```

```
return elements;
}
```

```
public List getNextElements(int count) throws
ValueListHandlerException{
    // validate
    if(this.list == null)
        throw new ValueListHandlerException("List is
        null");
```

```
int i = 0;
Object object = null;
ArrayList elements = new ArrayList();
if(listIterator != null){
    while( listIterator.hasNext() && (i < count) ){
        object = listIterator.next();
        elements.add(object);
        i++;
    }
}
return elements;
}
```

DELIVERING .NET, JAVA, MAC OS X, AND XML TECHNOLOGIES

International Web Services Conference & Expo

Web Services Edge

SEPT. 30 -- OCT. 2, 2003

3rd Annual

web services **EDGE**
conference & expo

- Take in tutorials covering .NET & Web services
- Listen to success stories
- Evaluate case studies & best practices
- Experience hands-on labs



REGISTER TODAY!

CALL 201-802-3058

www.sys-con.com/edge

Register before September 26th and

SAVE \$100

2003 WEST



Santa Clara, California

KEYNOTE SPEAKERS



Vermeulen

CTO



Magee

VP, Oracle 9i



Litwack

Senior VP



Schmidt

VP, Systems Integration



Owned and Produced by:
SYS-CON EVENTS

Event Sponsors: **ORACLE** **Novell**

Education Sponsors: **Microsoft**



Media Sponsors:

WebServices

JAVA DEVELOPERS JOURNAL

WebSphere DEVELOPERS JOURNAL

XML JOURNAL

.NET JOURNAL

WebLogic

LINUXWORLD MAGAZINE

wireless BUSINESS TECHNOLOGY

LINUXWEEK

CF Advisor

ColdFusion Developer's Journal

PowerBuilder Developer's Journal

market WIRE

HSP STREET 2003

OASIS

SDTimes

LINUXWORLD.COM

SAMS

IBM BUSINESS JOURNAL

**web
services**
conference & expo

EDGE
conference & expo



Conference & Expo
**THE LEADING
EVENT**
for i-Technology Professionals



SEPT. 30 — OCT. 2, 2003
Santa Clara Convention Center

FEATURES & ATTRACTIONS

- 3 Days Packed with Education and Training
- Keynotes & Panel Discussions from Industry Leaders
- 60 Hard-hitting and Informative Seminars
- **FREE** Web Services Workshop Presented by Oracle
- **FREE** .NET Tutorial with Microsoft's Russ' Tool Shed
- Java University Certification Training
- Industry-Leading Certification Programs
- **FREE** IBM Web Services Tutorial
- "Birds of a Feather" Discussions
- Round Table Discussions
- Opening Day Welcome Reception
- SAMS Meet the Authors Hot Topics Lounge
- Compelling Case Studies & Best Practices
- Hands-On Labs
- Featured Product Demonstrations
- Exhibit Floor featuring more than 40 companies and hundreds of products
- Real-time SYS-CON Radio Interviews

For more information visit
WWW.SYS-CON.COM
or call
201 802-3069

**WHO
SHOULD
ATTEND**



CEO
CTO
CIO
IT Director
Project Manager

- ▶ Software Developer
- ▶ Software Engineer
- ▶ Development Manager
- ▶ Application Developer
- ▶ Technical Director
- ▶ Analyst/Programmer
- ▶ IT Manager
- ▶ Technical Architect
- ▶ Team Leader
- ▶ Software Consultant

KEYNOTES & HIGHLIGHTED SPEAKERS

**Allan Vermeulen**

CTO, Amazon.com

Sept. 30 10:00 a.m.

"Web Services Foundations"

Allan Vermeulen, CTO and vice president at Amazon.com, directly oversees the Platform Technologies group. This group is responsible for guiding Amazon.com's technology architecture, including building and acquiring foundational components. Prior to his move to Amazon.com, Vermeulen was CTO and vice president of development at Rogue Wave Software. He holds a PhD in systems design engineering from the University of Waterloo.

**John Magee**

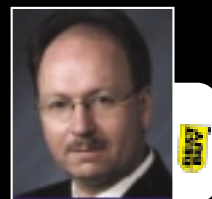
Vice President,

Oracle9i Application Server, Oracle
Oct. 1 10:00 a.m.**"J2EE Development on the Grid"**

John Magee is vice president of Oracle9i Application Server and Oracle9i Developer Suite at Oracle. Mr. Magee has over 14 years of experience in the enterprise software industry and has held positions in product development, product management, and product marketing. In his current role, he manages technical product marketing for Oracle's application server and development tools products, and is responsible for evangelizing Oracle technology initiatives around J2EE, XML, and Web services.

**David Litwack**Senior Vice President, Web
Application DevelopmentProducts, Novell
Sept. 30 2:00 p.m.**"Business Integration****and IT" Keynote Panel**

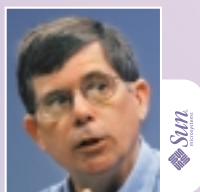
David A. Litwack is senior vice president of Web Application Development Products, responsible for the development and advancement of Novell's secure Web services strategy. Mr. Litwack assumed his current position in July 2002 following Novell's acquisition of SilverStream Software, a company for which Litwack had served as president and CEO since 1997.

**John Schmidt**Leader of Systems Integration
and Middleware, Best Buy Co.

Sept. 30 2:00 p.m.

"Business Integration**and IT" Keynote Panel**

John Schmidt is the chairman of the Methodology Committee for the EAI Industry Consortium and leader of systems integration and middleware at Best Buy Co., a leading specialty retailer of consumer electronics, personal computers, entertainment software, and appliances.

**Jon Bosak**Distinguished Engineer, Sun
Microsystems

Jon Bosak organized and led the W3C working group that created the XML specification and then served for two years as chair of the W3C XML Coordination Group. At Sun, where he holds the title of Distinguished Engineer, Mr. Bosak sponsors projects intended to advance XML technology. He is currently chair of the Universal Business Language (UBL) Technical Committee of OASIS.

**Dave Chappell**VP, Chief Technology Evangelist,
Sonic Software

Dave Chappell is the vice president and chief technology evangelist for Sonic Software. He has more than 18 years of industry experience building software tools and infrastructure for application developers, spanning all aspects of R&D, sales, marketing, and support services. Dave has also been published in numerous technical journals, and is currently writing a series of contributed articles for *Java Developer's Journal*.

**Anne Thomas
Manes**

Research Director, Burton Group

Anne Thomas Manes is a research director at Burton Group, a research, consulting, and advisory firm. Anne leads research for the Application Platform Strategies service. Named one of NetworkWorld's "50 Most Powerful People in Networking" in 2002, and one of Enterprise Systems Journal's "Power 100 IT Leaders" in 2001, Anne is a renowned technologist in the Web services space. Anne participates in standards development at W3C and OASIS.

**Marc Fleury**

President, JBoss

Marc Fleury, PhD, is chief technical officer for Telcel, Inc. He is the leader of the JBoss project (www.jboss.org), which is an open source EJB server. Marc is based out of Silicon Valley and founded the project upon leaving Sun Microsystems. He was one of the main developers behind JBoss 1.0 and 2.0. Marc is the "keeper" of the project. He founded the JBoss Group, a company regrouping the elite developers of JBoss to consult around Jboss.

Hotel & Travel

**Reserve Your Hotel Room Now
At The Westin Santa Clara!**The Official Conference Hotel of
Web Services Edge West 2003The Westin Santa Clara
5101 Great America Parkway
Santa Clara, CA 95054

Arrangements have been made with the Westin Santa Clara, which is conveniently located at The Santa Clara Convention Center. Specially reduced rates have been secured at this luxury, full-service hotel.

Single Occupancy Room: \$165.00
Double Occupancy Room: \$165.00

All rooms are quoted exclusive of applicable state and local taxes which are currently 9.5% as well as the California State Tourism Tax of 0.045%. The above rates are group rates and are available for Web Services Edge 2003 delegates, over the show dates of September 28 - October 3, 2003, only.

To learn more about The Westin Santa Clara you can contact the hotel directly or you can make your reservations by calling Expo Travel International at (800) 829-2281 or (201) 444-0060 (direct). Fax reservations to (201) 444-0062. Credit card information is required to guarantee reservations and expedite confirmation. Confirmations will be mailed directly from the hotel,

time permitting. All changes and cancellations should be made directly through Expo Travel International.

To make online reservations:

www.expotravel.com by September 12, 2003.

Reservations received after September 12, 2003, will be accepted on a space-available basis only, at the special rate, if available.

Contact Information:

The Westin Santa Clara Reservations:
Tel: 408 986-0700
Fax: 408 980-3990

Hotel Arrangements Are Easier Than Ever! You have your choice - contact the hotel directly or call us. The Official Conference travel agent, Expo Travel International.

Official Conference Travel Agent:

Expo Travel International
Toll Free: (800) 829-2281
Tel: (201) 444-0060
Fax reservations to (201) 444-0062

Driving Directions to Westin Santa Clara from San Jose Airport:

Highway 101 North. Exit at Great America Parkway/Bowers. Turn Right onto Great America Parkway; hotel is about 1.5 miles down on the right side.

**SPECIAL
DISCOUNTS
AVAILABLE**

Take advantage of the Early Bird and Preregistration values available right now, or save even more with a group of 5 or more. For special group discounts contact Michael Lynch at mike@sys-con.com, or by phone at (201) 802-3058.

PRODUCED BY
**SYS-CON
EVENTS**

SPECIAL INSERT: WEB SERVICES EDGE 2003

TUESDAY, SEPTEMBER 30

	JAVA	.NET	WEB SERVICES	
8:00AM – 4:00PM	REGISTRATION			
9:00AM – 9:50AM	The Next Phase in Evolution of J2EE	Using WSE 2.0	Web Services Management	
10:00AM – 10:50AM	Keynote - "Web Services Foundations" - Allen Vermeulen, CTO and Vice President, Amazon.com			
11:00AM – 6:00PM	EXPO OPEN			
2:00PM – 2:50PM	Keynote Panel Discussion - Business Integration and <i>i</i> -Technology			
3:00PM – 3:50PM	Ant Applied in "Real World" Web Services	Smart Devices in the Enterprise	Building Interoperable Web Services Using WS-I Basic Profile	
4:00PM – 4:50PM	Developing Applications with SWT	Using the Mobile Internet Toolkit	Web Services Orchestration	
5:00PM	OPENING NIGHT RECEPTION			

WEDNESDAY, OCTOBER 1

8:00AM – 4:00PM	REGISTRATION			
9:00AM – 9:50AM	Empowering Java and RSS for Blogging	Introduction to ROTOR	ID, Please. The Case for Giving Web Services an Identity	
10:00AM – 10:50AM	Morning Keynote - "J2EE Development on the Grid" - John Magee, Vice President, Oracle9i, Oracle			
11:00AM – 4:00PM	EXPO OPEN			
2:00PM – 2:50PM	Keynote Panel Discussion - Interoperability: Is Web Services Delivering?			
3:00PM – 3:50PM	JUnit: Testing Your Java with JUnit	Using Portable .NET	WS-BPEL	
4:00PM – 4:50PM	JDK1.5: The Tiger	ASP.NET with Mono	UDDI: Dead or Alive?	
5:00PM – 6:00PM	Squeezing Java	Using WSE with IBM's Web Services Tool Kit	Web Services Choreography, Management, and Security - Can They Dance Together?	

THURSDAY, OCTOBER 2

8:00AM – 4:00PM	REGISTRATION			
9:00AM – 9:50AM	Leveraging AOP in JBoss	Success Story: Eiffel, .NET, and Design by Contract for the Financial Industry	Strategies for Securing Web Services	
10:00AM – 10:50AM	Technical Keynote			
11:00AM – 11:50AM	Apache Axis	.NET IDE's	Web Services Progress Report	
12:00PM	LUNCH			
1:00PM – 1:50PM	Meeting the Challenges of J2ME Development	Windows SharePoint Services	The Seven Habits of Highly Effective Enterprise Service Buses (ESBs)	
2:00PM – 2:50PM	Keynote Panel Discussion - Summit on Web Services Standards			
3:00PM – 3:50PM	Simplifying J2EE Applications	BizTalk 2004	See www.sys-con.com for more information	
4:00PM – 5:00PM	Integrating Java + .NET	See www.sys-con.com for more information	See www.sys-con.com for more information	

REGISTER BEFORE SEPTEMBER 26th – SAVE \$100

XML

MAC OS X

Introduction to Xforms

Introducing OS X
(Panther) What's New?Securing Your XML and
Web Services InfrastructureProgramming Rich User
Interfaces Using CocoaUBL - The Universal Business
LanguageQuick Applications
Using AppleScriptStandards-Based Enterprise
Middleware Using XML/Web
Services

Java and OS X: A Perfect Marriage

XML and Enterprise
Architecture: Technology
Trends

Enterprise Java and OS X

Using XML Schemas
Effectively in WSDL Design

Developing Web Services Using WebObjects

Canonical Documents for
Your Business: Design
StrategiesCocoa, Carbon, Java: Application Frameworks
for OS X (When to Use What)

XML and the Fortune 500

Securing OS X Applications

XML at Work in 'Fortune
500' CompaniesXserve: Ease of OS X
and Power of Unix

XML Schema Best Practices

OS X for the Unix Developer

See www.sys-con.com for
more informationIntroducing Quartz: 2D
Graphics for AppleSee www.sys-con.com for
more informationSee www.sys-con.com for
more information**FREE** Web Services Workshop
presented by**ORACLE®****October 1, 2003****SHARPEN
YOUR
SKILLS,
DEVELOP
YOUR
CAREER**

Web services? You've read all the ins and outs about it. You think you have the concepts pretty well figured out. Now you are not sure where and how to start developing your first Web service. Get the answer at this free Web services workshop offered by Oracle as part of its Oracle Developer Days roadshow!

Oracle's workshop is specifically designed to get you started with your first Web service project, with a combination of presentations and hands-on labs that take you deep into the technology and let you put in action what you've learned. Oracle's experts will be available throughout the workshop to answer all your questions and assist you while you are going through the labs.

The workshop gives tips and techniques on how best to develop and deploy Web services and addresses topics such as RPC and Document Style Web services, static and dynamic invocation, stateless Web services and more. The second part of the workshop is dedicated to the new J2EE API for Web services available as part of J2EE 1.4.

Going through the hands-on labs at your own pace, you will learn how to publish a Java class as a J2EE stateless or stateful Web service, publish a session EJB as a J2EE Web service, and publish a J2EE Web service using JAX-RPC.

Space is LIMITED to the first 100 attendees. Register now for this FREE workshop. Computers will be provided by the Oracle Developer Days team with all the necessary software, so there's no need to bring your own computer.

AGENDA

7:30-8:00 am - Registration

8:00-9:00 am - Session #1 - Best Practices for Web Services Development & Deployment

9:00-10:00 am - Lab #1 - Publish a Java Class as a J2EE Stateless or Stateful Web Service

10:00-10:50 am - John Magee, VP, Oracle - Keynote (BREAK)

11:00 am-12:00 pm - Expo Floor Time

12:00-1:00 pm - Session #2 (WORKING LUNCH) - J2EE APIs for Web Services

1:00-2:00 pm - Lab #2- Publish a Session EJB as a J2EE Web Service

2:00-2:30 pm - Expo Floor Time (BREAK)

2:30-3:00 pm - Lab #3- Publish a J2EE Web Service Using JAX-RPC

PRESENTERSArun Srinivasan, Director of Product Management, Java Tools,
OracleRob Clark, Director of Product Management, J2EE, Oracle
Mike Lehmann, Product Manager, Web Services, Oracle9iAS and Oracle9i
JDeveloper, Oracle*FREE Oracle Tutorial when you register for a VIP Pass**Guarantee your seat when you register for a Full Conference Pass*

Register Online at
www.sys-con.com/edge

SPECIAL INSERT: WEB SERVICES EDGE 2003

REGISTRATION FORM

CONFERENCE: Sept. 30 – Oct. 2, 2003 EXPO: Sept. 30 – Oct. 1, 2003

Santa Clara Convention Center • Santa Clara, CA

THREE WAYS TO REGISTER FOR CONFERENCE

- 1) **On the Web:** Credit Cards or "Bill Me." Please make checks payable to SYS-CON Events.
 2) **By Fax:** Credit Cards or "Bill Me" 201-782-9651
 3) **By Mail:** 135 Chestnut Ridge Road, Montvale, New Jersey 07645, Attention: Registration

Please note: Registrations are not confirmed until payment is received.

Please complete sections 1, 2, 3 and 4

1 YOUR INFORMATION (Please Print) ☐ Mr. ☐ Ms.

First Name _____ Last Name _____
 Title _____
 Company _____
 Street _____
 Mail Stop _____
 City _____
 State _____ Zip _____ Country _____
 Phone _____
 Fax _____ E-Mail _____

2 PAYMENT METHOD: (Payment in full due with registration)

☐ Check or Money Order Enclosed (Registration confirmed upon receipt of payment)
 Check # _____ Amount of Check \$ _____
 Charge my ☐ Visa ☐ MasterCard ☐ American Express ☐ Discover
 Name on card _____
 Card # _____ Exp. Date _____
 Signature _____
 Billing Address (if different from mailing address) _____

3 PLEASE INDICATE YOUR CONFERENCE CHOICE

Total Registration fee \$ _____

	By 9/5/03	Before 9/26/03	Onsite
<input type="checkbox"/> GP Gold Passport Good for all three days of the .NET, Web Services, XML, Java, and Mac OS X Tracks, including preferred seating for the Oracle, IBM and Microsoft Russ' Toolshed Tutorials, Keynotes, Panel Discussions, and your choice of One Sun Microsystems Java University SM Class Select one: <input type="checkbox"/> Architecting Web Services Using J2EE (Oct. 1) <input type="checkbox"/> Java 2 Platform: Architect Certification Fast Path (Oct. 2)	\$1,295.00	\$1,395.00	\$1,495.00
<input type="checkbox"/> 3D Three Day Conference (Does not include Sun Java Education)	\$1,195.00	\$1,295.00	\$1,395.00
<input type="checkbox"/> 2D Two Day Conference (Does not include Sun Java Education) (select any two days: <input type="checkbox"/> Tue. <input type="checkbox"/> Wed. <input type="checkbox"/> Thurs.)	\$1,095.00	\$1,195.00	\$1,295.00
<input type="checkbox"/> 1D One Day Conference (Does not include Sun Java TM Education) (select any one day: <input type="checkbox"/> Tue. <input type="checkbox"/> Wed. <input type="checkbox"/> Thurs.)	\$595.00	\$595.00	\$695.00
<input type="checkbox"/> JU1 Sun JavaTM University Class Select one: <input type="checkbox"/> Architecting Web Services Using J2EE (Oct. 1) <input type="checkbox"/> Java 2 Platform: Architect Certification Fast Path (Oct. 2)	\$695.00	\$695.00	\$795.00
<input type="checkbox"/> JU2 Sun Java University Class Attend both Architecting Web Services Using J2EE (Oct. 1) and Java 2 Platform: Architect Certification Fast Path (Oct. 2)	\$1,195.00	\$1,295.00	\$1,395.00
<input type="checkbox"/> VIP PASS Good for access to the Exhibit Floor, Keynotes and Panel Discussions, Product Demonstrations, and your choice of (Select one): <input type="checkbox"/> Microsoft Russ' Tool Shed (Sept. 30) <input type="checkbox"/> How to Develop, Deploy, and Manage Web Services Using IBM Tools (Sept. 30) <input type="checkbox"/> Web Services Workshop presented by Oracle (Oct. 1)	FREE	FREE	\$50.00
<input type="checkbox"/> EO Expo Only	FREE	FREE	\$50.00

4

A. Your Job Title

- ☐ CTO, CIO, VP, Chief Architect
☐ Software Development Director/Manager/Evangelist
☐ IT Director/Manager
☐ Project Manager/Project Leader/Group Leader
☐ Software Architect/Systems Analyst
☐ Application Programmer/Evangelist
☐ Database Administrator/Programmer
☐ Software Developer/Systems Integrator/Consultant
☐ Web Programmer
☐ CEO/COO/President/Chairman/Owner/Partner
☐ VP/Director/Manager Marketing, Sales
☐ VP/Director/Manager of Product Development
☐ General Division Manager/Department Manager
☐ Other (please specify) _____

B. Business/Industry

- ☐ Computer Software ☐ Government/Military/Aerospace
☐ Computer Hardware and Electronics ☐ Health Care/Medical
☐ Computer Networking & Telecommunications ☐ Insurance/Legal
☐ Internet/Web/E-commerce ☐ Education
☐ Consulting & Systems Integrator ☐ Utilities
☐ Financial Services ☐ Architecture/Construction/Real Estate
☐ Manufacturing ☐ Agriculture
☐ Wholesale/Retail/Distribution ☐ Nonprofit/Religious
☐ Transportation ☐ Other (please specify) _____
☐ Travel/Hospitality

C. Total number of employees at your location and entire organization (check all that apply):

	Location	Company
10,000 or more	01 <input type="checkbox"/>	01 <input type="checkbox"/>
5,000 – 9,999	02 <input type="checkbox"/>	02 <input type="checkbox"/>
1,000 – 4,999	03 <input type="checkbox"/>	03 <input type="checkbox"/>
500 – 999	04 <input type="checkbox"/>	04 <input type="checkbox"/>
100 – 499	05 <input type="checkbox"/>	05 <input type="checkbox"/>
100 or less	06 <input type="checkbox"/>	06 <input type="checkbox"/>

D. Please indicate the value of communications and computer products and services that you recommend, buy, specify, or approve over the course of one year:

- ☐ \$10 million or more ☐ \$10,000 – \$99,999
☐ \$1 million – \$9.9 million ☐ Less than \$10,000
☐ \$500,000 – \$999,999 ☐ Don't know
☐ \$100,000 – \$499,999

E. What is your company's gross annual revenue?

- ☐ \$10 billion or more ☐ \$1 million – \$9.9 million
☐ \$1 billion – \$9.9 billion ☐ Less than \$1 million
☐ \$100 million – \$999 million ☐ Don't know
☐ \$10 million – \$99.9 million

F. Do you recommend, specify, evaluate, approve or purchase wireless products or services for your organization?

01 ☐ Yes 02 ☐ No

G. Which of the following products, services, and/or technologies do you currently approve, specify or recommend the purchase of?

- ☐ Application Servers
☐ Web Servers
☐ Server Side Hardware
☐ Client Side Hardware
☐ Wireless Device Hardware
☐ Databases
☐ Java IDEs
☐ Class Libraries
☐ Software Testing Tools
☐ Web Testing Tools
☐ Modeling Tools
☐ Team Development Tools
☐ Installation Tools
☐ Frameworks
☐ Database Access Tools / JDBC Devices
☐ Application Integration Tools
☐ Enterprise Development Tool Suites
☐ Messaging Tools
☐ Reporting Tools
☐ Debugging Tools
☐ Virtual Machines
☐ Wireless Development Tools
☐ XML Tools
☐ Web Services Development Toolkits
☐ Professional Training Services
☐ Other [Please Specify] _____

SYS-CON Events, Inc., and SYS-CON Media make no warranties regarding content, speakers, or attendance. The opinions of speakers, exhibitors, and sponsors do not reflect the opinion of SYS-CON Events and SYS-CON Media and no endorsement of speakers, exhibitors, companies, products, or sponsors is implied.



If you require special assistance covered under the Americans with Disabilities Act, please call 201-802-3058 by September 16, 2003.



CANCELLATIONS, SUBSTITUTIONS, REFUNDS
 Fax written request to SYS-CON Registration 201-782-9651. Requests for refunds received prior to August 29, 2003, will be honored, less a 10% handling charge; requests received after August 29, 2003, and before September 12,

2003, will be honored less a 20% handling charge. No requests for refunds will be honored after September 12, 2003. Requests for substitutions must be made in writing prior to September 26, 2003. No one under 18 is permitted to attend. No warranties are made regarding the content of sessions or materials.

Speakers, sessions, and schedule are subject to change without prior notice.

No solicitation by anyone other than official exhibitors, sponsors or marketing partners is permitted. Such behavior is cause for expulsion without refund.

SUBSCRIBE TODAY TO MULTIPLE MAGAZINES AND SAVE UP TO \$400 AND RECEIVE UP TO 3 FREE CDs!

RECEIVE
YOUR DIGITAL
EDITION
ACCESS CODE
INSTANTLY
WITH YOUR PAID
SUBSCRIPTIONS



TO ORDER

- Choose the Multi-Pack you want to order by checking next to it below. • Check the number of years you want to order. • Indicate your location by checking either U.S., Canada/Mexico or International. • Then choose which magazines you want to include with your Multi-Pack order.

Pick a 3-Pack, a 6-Pack or a 9-Pack

<input type="checkbox"/> 3-Pack	<input type="checkbox"/> 1YR <input type="checkbox"/> 2YR	<input type="checkbox"/> U.S. <input type="checkbox"/> Can/Mex <input type="checkbox"/> Intl.
<input type="checkbox"/> 6-Pack	<input type="checkbox"/> 1YR <input type="checkbox"/> 2YR	<input type="checkbox"/> U.S. <input type="checkbox"/> Can/Mex <input type="checkbox"/> Intl.
<input type="checkbox"/> 9-Pack	<input type="checkbox"/> 1YR <input type="checkbox"/> 2YR	<input type="checkbox"/> U.S. <input type="checkbox"/> Can/Mex <input type="checkbox"/> Intl.

Linux World Magazine			
U.S. - Two Years (24) Cover: \$143	You Pay: \$79.99 /	Save: \$63 + FREE \$198 CD	
U.S. - One Year (12) Cover: \$72	You Pay: \$39.99 /	Save: \$32	
Can/Mex - Two Years (24) \$168	You Pay: \$119.99 /	Save: \$48 + FREE \$198 CD	
Can/Mex - One Year (12) \$84	You Pay: \$79.99 /	Save: \$4	
Intl - Two Years (24) \$216	You Pay: \$176 /	Save: \$40 + FREE \$198 CD	
Intl - One Year (12) \$108	You Pay: \$99.99 /	Save: \$8	
Java Developer's Journal			
U.S. - Two Years (24) Cover: \$144	You Pay: \$89 /	Save: \$55 + FREE \$198 CD	
U.S. - One Year (12) Cover: \$72	You Pay: \$49.99 /	Save: \$22	
Can/Mex - Two Years (24) \$168	You Pay: \$119.99 /	Save: \$48 + FREE \$198 CD	
Can/Mex - One Year (12) \$84	You Pay: \$79.99 /	Save: \$4	
Intl - Two Years (24) \$216	You Pay: \$176 /	Save: \$40 + FREE \$198 CD	
Intl - One Year (12) \$108	You Pay: \$99.99 /	Save: \$8	
Web Services Journal			
U.S. - Two Years (24) Cover: \$168	You Pay: \$99.99 /	Save: \$68 + FREE \$198 CD	
U.S. - One Year (12) Cover: \$84	You Pay: \$69.99 /	Save: \$14	
Can/Mex - Two Years (24) \$192	You Pay: \$129 /	Save: \$63 + FREE \$198 CD	
Can/Mex - One Year (12) \$96	You Pay: \$89.99 /	Save: \$6	
Intl - Two Years (24) \$216	You Pay: \$170 /	Save: \$46 + FREE \$198 CD	
Intl - One Year (12) \$108	You Pay: \$99.99 /	Save: \$8	
.NET Developer's Journal			
U.S. - Two Years (24) Cover: \$168	You Pay: \$99.99 /	Save: \$68 + FREE \$198 CD	
U.S. - One Year (12) Cover: \$84	You Pay: \$69.99 /	Save: \$14	
Can/Mex - Two Years (24) \$192	You Pay: \$129 /	Save: \$63 + FREE \$198 CD	
Can/Mex - One Year (12) \$96	You Pay: \$89.99 /	Save: \$6	
Intl - Two Years (24) \$216	You Pay: \$170 /	Save: \$46 + FREE \$198 CD	
Intl - One Year (12) \$108	You Pay: \$99.99 /	Save: \$8	
XML Journal			
U.S. - Two Years (24) Cover: \$168	You Pay: \$99.99 /	Save: \$68 + FREE \$198 CD	
U.S. - One Year (12) Cover: \$84	You Pay: \$69.99 /	Save: \$14	
Can/Mex - Two Years (24) \$192	You Pay: \$129 /	Save: \$63 + FREE \$198 CD	
Can/Mex - One Year (12) \$96	You Pay: \$89.99 /	Save: \$6	
Intl - Two Years (24) \$216	You Pay: \$170 /	Save: \$46 + FREE \$198 CD	
Intl - One Year (12) \$108	You Pay: \$99.99 /	Save: \$8	

WebLogic Developer's Journal			
U.S. - Two Years (24) Cover: \$360	You Pay: \$169.99 /	Save: \$190 + FREE \$198 CD	
U.S. - One Year (12) Cover: \$180	You Pay: \$149 /	Save: \$31	
Can/Mex - Two Years (24) \$360	You Pay: \$179.99 /	Save: \$180 + FREE \$198 CD	
Can/Mex - One Year (12) \$180	You Pay: \$169 /	Save: \$11	
Intl - Two Years (24) \$360	You Pay: \$189.99 /	Save: \$170 + FREE \$198 CD	
Intl - One Year (12) \$180	You Pay: \$179 /	Save: \$1	
ColdFusion Developer's Journal			
U.S. - Two Years (24) Cover: \$216	You Pay: \$129 /	Save: \$87 + FREE \$198 CD	
U.S. - One Year (12) Cover: \$108	You Pay: \$89.99 /	Save: \$18	
Can/Mex - Two Years (24) \$240	You Pay: \$159.99 /	Save: \$80 + FREE \$198 CD	
Can/Mex - One Year (12) \$120	You Pay: \$99.99 /	Save: \$20	
Intl - Two Years (24) \$264	You Pay: \$189 /	Save: \$75 + FREE \$198 CD	
Intl - One Year (12) \$132	You Pay: \$129.99 /	Save: \$2	
Wireless Business & Technology			
U.S. - Two Years (24) Cover: \$144	You Pay: \$89 /	Save: \$55 + FREE \$198 CD	
U.S. - One Year (12) Cover: \$72	You Pay: \$49.99 /	Save: \$22	
Can/Mex - Two Years (24) \$192	You Pay: \$139 /	Save: \$53 + FREE \$198 CD	
Can/Mex - One Year (12) \$96	You Pay: \$79.99 /	Save: \$16	
Intl - Two Years (24) \$216	You Pay: \$170 /	Save: \$46 + FREE \$198 CD	
Intl - One Year (12) \$108	You Pay: \$99.99 /	Save: \$8	
WebSphere Developer's Journal			
U.S. - Two Years (24) Cover: \$360	You Pay: \$169.99 /	Save: \$190 + FREE \$198 CD	
U.S. - One Year (12) Cover: \$180	You Pay: \$149 /	Save: \$31	
Can/Mex - Two Years (24) \$360	You Pay: \$179.99 /	Save: \$180 + FREE \$198 CD	
Can/Mex - One Year (12) \$180	You Pay: \$169 /	Save: \$11	
Intl - Two Years (24) \$360	You Pay: \$189.99 /	Save: \$170 + FREE \$198 CD	
Intl - One Year (12) \$180	You Pay: \$179 /	Save: \$1	
PowerBuilder Developer's Journal			
U.S. - Two Years (24) Cover: \$360	You Pay: \$169.99 /	Save: \$190 + FREE \$198 CD	
U.S. - One Year (12) Cover: \$180	You Pay: \$149 /	Save: \$31	
Can/Mex - Two Years (24) \$360	You Pay: \$179.99 /	Save: \$180 + FREE \$198 CD	
Can/Mex - One Year (12) \$180	You Pay: \$169 /	Save: \$11	
Intl - Two Years (24) \$360	You Pay: \$189.99 /	Save: \$170 + FREE \$198 CD	
Intl - One Year (12) \$180	You Pay: \$179 /	Save: \$1	

3-Pack
Pick any 3 of our
magazines and save
up to \$275⁰⁰
Pay only \$175 for a
1 year subscription
plus a **FREE CD**

- 2 Year - \$299.00
- Canada/Mexico - \$245.00
- International - \$315.00

6-Pack
Pick any 6 of our
magazines and save
up to \$350⁰⁰
Pay only \$395 for a
1 year subscription
plus 2 **FREE CDs**

- 2 Year - \$669.00
- Canada/Mexico - \$555.00
- International - \$710.00

9-Pack
Pick 9 of our
magazines and save
up to \$400⁰⁰
Pay only \$495 for a
1 year subscription
plus 3 **FREE CDs**

- 2 Year - \$839.00
- Canada/Mexico - \$695.00
- International - \$890.00

OFFER SUBJECT TO CHANGE WITHOUT NOTICE

Subscribe Online Today www.sys-con.com/2001/sub.cfm

**SYS-CON
MEDIA**

Making role-based, composite applications a reality

The Trend Toward Adaptive Software

BY JOE LICHTENBERG



ABOUT THE AUTHOR

As vice president of marketing for Bowstreet, Joe Lichtenberg is responsible for setting the company's product direction and messaging. Joe holds a bachelor's degree in mechanical engineering from the University of Vermont and a master's degree in manufacturing engineering from Boston University.

E-MAIL

jlichtenberg@bowstreet.com

The days of customers pouring millions of dollars into buying and customizing "stovepipe" proprietary enterprise applications are giving way to a new scenario: companies are building composite applications that provide their employees, partners, and customers with the ability to interact with their various disparate back-end systems. Composite applications span industries and can take many forms. For example, a manufacturing company might provide an intranet for its sales force to access customer data from its CRM system, check order status and order history from its ERP system, and chat or participate in threaded messages via collaboration software, all from one place.

Such a system enables sales reps to be more productive, improves customer satisfaction, and increases forecast visibility inside the company. The same manufacturing company might also create an e-commerce extranet that allows customers to create, edit, cancel, and track orders by interacting directly with the company's ERP system, minimizing order entry errors and improving the purchasing process for customers. Companies in all industries are creating dashboards that aggregate key performance indicators from various applications into one place, for better decision making and for compliance with the Sarbanes-Oxley Act,

the SEC's stringent rules aimed at gaining investors' trust by raising accountability standards for public companies.

A confluence of technology has made the development of such applications possible.

Enterprise application vendors are decomposing their previously monolithic applications into a set of low-level, standards-based components, as exemplified by SAP's NetWeaver and xApps, and PeopleSoft's AppConnect. Similar to the way Lego blocks can be combined to form many different shapes, these components can be combined into customized applications that leverage data and func-

tionality from disparate back-end applications.

These applications are being deployed most commonly as portals and portlets. Portlets are simply mini-applications running inside a portal framework. Portlets are extremely well suited to the task, as they enable granular, discrete pieces of application functionality to be built, deployed, and reused. For example, the order status portlet from the sales intranet can be reused in the e-commerce extranet. Also, portlets can communicate with each other so that the portlet that presents customer data from the CRM system can transmit a customer ID to the order status portlet, such that the appropriate customer data from the ERP system is automatically served up. Companies use this portlet-to-portlet communication as a new and lightweight form of application integration. Finally, standards such as JSR168 and WSRP will allow any portlet to run in any portal.

Although the applications and infrastructure have evolved to support the creation of such applications, they have not yet become commonplace.

Why Not?

Much of the value of these applications comes not only from providing centralized access to functionality from disparate enterprise applications, but in doing so in a customized, role-based manner, creating a highly tailored experience for each user or group (or for any application context, for that matter). For example, in the intranet described above, each sales rep, upon accessing the intranet, would be recognized and automatically served with the data from each system that is pertinent to his or her accounts only. Each district manager would see a roll-up of the accounts for all of his or


“Similar to the way Lego blocks can be combined to form many different shapes, these components can be combined into customized applications that leverage data and functionality from disparate back-end applications”

her reps. And the vice president of sales might see additional restricted data not seen by reps and managers. In the extranet example, each customer would see an individualized version of the extranet that might display only the items they are able to order, and provide custom pricing, purchasing processes, and branding. To build such applications using traditional software, developers will code multiple variations via conditional logic or multiple code branches, which is time consuming, error prone, and difficult to maintain as the application grows or changes. Traditional software simply does not lend itself to building dynamic, role-based composite applications.

Enter Adaptive Software

There is a new type of software, called adaptive software, that is making the development of role-based, composite applications a reality. Adaptive software is the key component that makes it practical – for the first time – to build composite applications that adapt to various user roles or application contexts, allowing any aspect of the application – including the data, the processes, and the presentation – to be customized on demand for individual employees, partners, and customers.

Just as a spreadsheet allows analysts to create complex financial models and then vary parameter values to get a new set of results, adaptive software similarly captures the developers' design intent and automates the creation of standards-compliant software applications by feeding the application with new sets of parameters or profiles. New versions of the application can be dynamically generated at design time by explicitly providing new profiles, or at runtime, by programmatically determining new profile values based on any application context, and dynamically regenerating the various aspects of the application. Adaptive software frees developers from the tedious task of building and maintaining the multiple variations of an application required with traditional programming techniques.

With adaptive software, companies with heterogeneous environments of enterprise applications, infrastructure software, and portal software can take advantage of the trend toward componentization and standardization to increase efficiencies, lower costs, and improve customer loyalty by more fully leveraging the systems they already have in place. Software companies and enterprises alike are talking about e-business on demand and the adaptive enterprise. Adaptive software is making it a reality. 

WebSphere
DEVELOPER'S JOURNAL

Advertiser Index...

COMPANY	URL	PHONE	PG
DIRIG SOFTWARE	WWW.EBIZPERFORM.COM	603-889-2777	2
H & W COMPUTER	WWW.HWCS.COM	208-377-0336	19
KENETIKS	WWW.KENETIKS.COM	888-KENETIKS	11
MACROMEDIA	WWW.MACROMEDIA.COM/GO/WSDJ/	415-252-2000	51
MQSOFTWARE	WWW.MQSOFTWARE.COM	952-345-8720	6,7
PROLIFICS	WWW.PROLIFICS.COM/WEBSERVICES	800-675-5419	5
QUEST SOFTWARE	HTTP://JAVA.QUEST.COM/JPROBE/WDJ	800-663-4723	15
SYS-CON MEDIA	WWW.SYS-CON.COM/2001/SUB.CFM	888-303-5282	45
SYS-CON MEDIA DATABASE	WWW.EDITHROMAN.COM, WWW.EPOSTDIRECT.COM	800-223-2194, 800-409-4443	49
TRILOG GROUP	WWW.FLOWBUILDER.COM	800-818-2199	52
VERSATA	WWW.VERSATA.COM/BUSINESSLOGICDESIGNER	800-984-7638	21
WEBSERVICES EDGE WEST 2003	WWW.SYS-CON.COM/WEBSERVICES2003WEST	201-802-3058	35, 39-44
WILY TECHNOLOGY	WWW.WILYTECH.COM	888-GET-WILY	3

WebSphere
DEVELOPER'S JOURNAL

Coming Next Month...

ILLUMINATING WEBSHERE

Understanding and Optimizing Java Management Extensions (JMX)

BY GIRISH KULKARNI AND BARRY NANCE

PORTAL

Using WebSphere Portal to Manage an Order-Tracking Process

BY CHRISTINA LAU AND COLIN YU

TUTORIAL

Developing Applications with WebSphere Studio's Visual Editor for Java and JavaBeans

BY COLETTE BURRUS

PRODUCT REVIEW

Rational Rapid Developer: Product opens J2EE development up to a variety of application developers

BY JAY JOHNSON

ASCENTIAL SOFTWARE DELIVERS REAL-TIME DATA INTEGRATION ON WAS 5.0

(Westboro, MA) – Ascential Software Corporation, an enterprise data integration leader, has announced support for IBM WebSphere Application Server 5.0.



The combination of Ascential

Enterprise Integration Suite and Real-Time Integration Services with WebSphere delivers advanced technology solutions to power the real-time enterprise.

Integrating the Ascential Software Enterprise Integration Suite and WebSphere Application Server makes the real-time enterprise a reality, enabling users to take advantage of reliable, accurate, and current information available from anywhere across the enterprise, at any time, to drive strategic business initiatives. Harnessing the benefits of a service-oriented architecture, solutions from Ascential Software eliminate concerns about data sources, quality, volumes, and latency, enabling customers to leverage information across strategic applications such as customer

relationship management, enterprise resource planning, and supply chain management.

www.ascential.com

INSTANTIATIONS SHIPS NEW RELEASE OF CODEPRO STUDIO

(Portland, OR) – Instantiations, a leading provider of advanced Java development solutions, has begun shipping CodePro Studio 2.3, a comprehensive suite of products that enhance **Instantiations** WebSphere Studio and Eclipse development environments. The updated CodePro family includes CodePro Advisor (best practices tools), CodePro Agility (productivity tools), CodePro Build (build management tools), and CodePro Studio (the complete suite).

CodePro Studio 2.3 includes significant new features that are designed to maximize productivity, increase code quality and reduce development costs. CodePro Studio now includes team collaboration and workspace administration tools. These unique new capabilities are built on the Eclipse Koi technology infrastructure – an open source

project led by Instantiations.

Developers will benefit from powerful collaborative features like inter-workspace messaging, global task scheduling and centralized workspace configuration.

CodePro Studio products seamlessly integrate into IBM WebSphere Studio and Eclipse development environments, according to the company. CodePro adds a rich set of features to the base development environment, resulting in greater capability, and allows developers to work with the tools of their choice. CodePro products have a proven reputation for enabling organizations to reduce time to market, deliver reliable applications, and significantly reduce costs.

www.instantiations.com/codepro

CANDLE EXPANDS SUPPORT FOR LINUX, WEBSphere MQ

(El Segundo, CA) – Candle Corporation, a leading enterprise infrastructure management provider, has announced the expansion of its offerings for Linux environments by providing new Linux support for IBM



WebSphere MQ management as well as 64-bit platform support across its entire Linux management solution portfolio. Building on the debut of its Linux support in August 2000, Candle now offers a full portfolio of Linux enterprise management solutions.

Candle strengthened its Linux offerings with the debut of Omegamon XE (Extended Edition) for Linux – an end-to-end solution for testing, tuning, and managing the performance and availability of Linux on IBM eServer zSeries mainframes. Candle's expanding Linux solution suite underscores its commitment to supporting customers' enterprise Linux deployments.

www.candle.com

OPENDMAND AND COMPUFLEX UNITE ON RAD/PERFORMANCE

(Newark, NJ) – OpenDemand Systems, Inc. and Compuflex International has announced the formation of a strategic partnership to deliver the industry's first Rapid Application Development (RAD) and Rapid Performance Optimization (RPO) platform, OpenLoad-WebAccel PowerPack.

This powerful, easy-to-use platform enables firms to quickly build and test database-enabled JSP applications five to eight times faster than traditional development tools. The development platform is fully integrated with popular IDEs



such as WebSphere Studio Application Developer (WSAD), Eclipse, and Macromedia DreamWeaver. By using application patterns, the development platform enables the design of the most advanced thin-client GUI capabilities available, while the testing platform provides the capability to quickly identify opportunities to improve design performance. Pricing for the OpenLoad-WebAccel PowerPack starts at \$3,999.00.

www.opendemand.com



DataPower Release Enhances MQSeries Support

(Cambridge, MA) – DataPower Technology, Inc., the leading provider of intelligent XML-Aware Networking (XAN), announced Release version 2.3 of firmware for the DataPower XS40 XML Security Gateway and the DataPower XA30 XML Accelerator, including the first WebSphere MQSeries protocol support in a network device, the first SOAP-based management interface for an XML-aware network device, and enhanced security features for nonrepudiation and XML denial of service (XDoS) protection.

In addition, Release 2.3 includes enhanced XML forensics, message velocity checking, and invalid message capture. DataPower's firmware Release 2.3 advances the XS40's and XA35's positions as the

best of a new class of network devices that are fully aware of the XML, SOAP, and Web services messages and that efficiently apply intelligent acceleration, security, and management services across the network.

The product's integration with WebSphere MQSeries extends beyond HTTP to deliver reliable wirespeed Web services security and intelligent acceleration for XML and SOAP messages traveling on top of IBM's MQ protocol. Among other benefits, this enables customers the ability to deploy a unified XML security solution across stable back-end systems and external Web services.

www.datapower.com

SYS-CON MEDIA

304,187 of the World's Foremost IT Professionals

DIRECT MAIL, EMAIL OR MULTI-CHANNEL

Target CTOs, CIOs and CXO-level IT professionals and developers who subscribe to SYS-CON Media's industry leading publications

Java Developer's Journal...
The leading publication aimed specifically at corporate and independent java development professionals



LinuxWorld Magazine...The premier weekly resource of linux news for executives with key buying influences



Web Services Journal...The only Web Services magazine for CIOs, tech, marketing & product managers, VARs/ISVs, enterprise/app architects & developers



XML Journal...The world's #1 leading XML resource for CEOs, CTOs, technology solution architects, product managers, programmers and developers



PowerBuilder Developer's Journal...The only PowerBuilder resource for corporate and independent enterprise client/server and web developers



WebSphere Developer's Journal...The premier publication for those who design, build, customize, deploy, or administer IBM's WebSphere suite of Web Services software



ColdFusion Developer's Journal...The only publication dedicated to coldfusion web development



WebLogic Developer's Journal...The official magazine for BEA WebLogic application server software developers, IT management & users



.NET Developer's Journal...The must read iTechnology publication for Windows developers & CXO management professionals



Wireless Business & Technology... The wireless magazine for key corporate & engineering managers, and other executives who purchase communications products/services

Recommended for a variety of offers including Java, Internet, enterprise computing, e-business applications, training, software, hardware, data back up and storage, business applications, subscriptions, financial services, high ticket gifts and much more.

NOW AVAILABLE!
The SYS-CON
Media Database
304,187 postal
addresses



For email information:
contact Frank at 845-731-3832
frank.cipolla@epostdirect.com
epostdirect.com 800-409-4443 fax 845-620-9035

For postal information:
contact Kevin at 845-731-2684
kevin.collopy@edithroman.com
edithroman.com 800-223-2194 fax 845-620-9035



Why Java Needs Rapid Application Development

Ease of use will be the next battleground

BY ZACK URLOCKER

At this year's annual gathering of the tribe at JavaOne in San Francisco, Sun announced its intention to increase the number of Java developers from 3 million to 10 million. And how is Sun going to achieve this? By making Java easier. All I can say is, "hallelujah, Scott. Welcome to the party."

Hats off to Sun for 'fessing up to the dirty little secret that Java developers have been struggling with for the past few years: J2EE is too difficult.

J2EE, and WebSphere in particular, have been tremendous successes in defining a platform for enterprise development that is hardware and OS independent, highly scalable, and reliable. J2EE has taken many difficult things and made them possible. J2EE 1.5 defines some base-level language constructs and meta-information tags that will help. But for Java to be relevant in the face of Microsoft's push for .NET, it needs a next generation of tools that let you assemble applications visually from existing components and services. Different terms have emerged for this new type of tool, but whether you call it an integrated services environment (ISE) or just plain old-fashioned rapid application development (RAD) is not the big concern.

Job Security or Job Obscurity?

Most companies I meet with have a small number of J2EE gurus who really understand the guts of the platform. These are uber-architects who live and breathe this stuff. But for every architect, there are usually 10 Java application developers. Some are good at database code, others excel at user interfaces, and still others might be experts in defining EJBs. The problem with J2EE today is that you need to be an expert in just about everything and inevitably have to wade into a lot of low-level plumbing code that has nothing to do with the application itself.

Some purists might argue against Sun's efforts to simplify Java as "dumbing it down." After all, if you've figured out how to be a guru, then you may not like the idea of inviting a bunch of Visual Basic or 4GL programmers to the party.

In these days of "do more with less" IT budgets, many companies are reluctant to green-light projects that require more than six months of development, especially if it's going to take months before



there's anything to see. For consultants trying to pitch clients, it's even worse. You usually need to show a working prototype – built on your own dime – to win the business. RAD tools for J2EE enable you to create a live, functional prototype that can be validated with users, managers, and the client before the heavy lifting is done. Rather than creating static screen shots and endless Visio diagrams, you can create a functional prototype that can evolve into a full-production Java application much more quickly. That

way developers, analysts, and users are all on the same page and working together.


Of course, not all RAD tools are created equal. You should look for RAD tools that are standards-based, provide an abstraction model through visual representation of business processes, and still let you get down to the code when you need to.

Java vs .NET

While Sun's desire to make Java easier is a good thing, make no mistake that Sun (and IBM and BEA and Oracle and...) are also fighting a much bigger battle with Microsoft to win the hearts and minds of developers. While Java is ahead in terms of enterprise scalability, reliability, and security, Microsoft will make ease of use the next battleground.

Microsoft will play the siren song of faster results to every CIO and systems integrator in the world. "Don't worry," they'll say, "it'll scale better in the next release." And to Microsoft's credit, it probably will. Microsoft is good at gradually improving its software until it's good enough.

For Java to have a solid future and not become just a niche technology, it cannot be better but more difficult; it needs to be both better and easier. Otherwise, sooner or later IT managers will face a tough choice between quick-and-dirty results or doing it right but taking longer. If you've lived through the adoption of graphical user interfaces or client/server development, then you can guess how things will turn out.

If you think your company couldn't benefit from RAD, I'd like to hear why. Meanwhile, I've got a prototype I need to deliver to a client in the next three days and I'd like to do it in J2EE. 

ABOUT THE AUTHOR... Zack Urlocker has been in the software tools and infrastructure business for over 15 years helping create such award-winning products as Delphi, JBuilder, and the M7 Application Assembly Suite. Zack is currently vice president of marketing at M7 Corporation.

E-MAIL

zurlocker@m7.com

MACROMEDIA

WWW.MACROMEDIA.COM/GO/WSDJ/

TRILOG GROUP

WWW.FLOWBUILDER.COM