

No. 1 *i*-Technology Magazine in the World

JDJ

JDJ.SYS-CON.COM

VOL.10 ISSUE:7



SEE PAGES 30-31

RICH-CLIENT-PORTLETS

The half-object + protocol design pattern

RETAILERS PLEASE DISPLAY
UNTIL AUGUST 31, 2005

\$5.99US \$6.99CAN

08>



PLUS...

- ▶ J2EE Development
the Eclipse Way
- ▶ Back to Two Tiers
and Plain JSP
- ▶ Mastering
Mustang
- ▶ Small Business
Solutions

SIAMESE FIGHTING FISH ATTACK AND KILL EACH OTHER IF PUT IN THE SAME ENVIRONMENT.

ENTERPRISE APPLICATIONS OFTEN BEHAVE SIMILARLY.

WINDOWS SERVER SYSTEM WITH .NET REDUCES THE HASSLE OF APPLICATION INTEGRATION.

CRM and supply chain applications really can live in harmony. How? With Windows Server System™ and .NET. The .NET Framework, an integral component of Windows Server System, is the development and execution environment that allows different components and applications to work together seamlessly. That means applications are easier to build, manage, deploy, and integrate. The .NET Framework uses industry standards such as XML and Web Services which

allow enterprise applications to be connected to infrastructure of any kind. In addition, the productivity-enhancing features of .NET, such as automatic mapping of data to and from XML, also help to simplify integration by reducing the amount of code required to get it all done.

Find out more about application integration with Windows Server System and .NET: Simply get the Connected Systems Resource Kit at microsoft.com/connectedsystems



WITH WINDOWS SERVER SYSTEM AND .NET, YOU CAN OVERCOME INTEGRATION ISSUES. HOWEVER, WINDOWS SERVER SYSTEM IS USELESS AGAINST THE SIAMESE FIGHTING FISH.


Microsoft
**Windows
Server System™**

Sun Did It in 1986; Microsoft Took Longer



Jeremy Geelan



Editorial Board
 Desktop Java Editor: **Joe Winchester**
 Core and Internals Editor: **Calvin Austin**
 Contributing Editor: **Ajit Sagar**
 Contributing Editor: **Yakov Fain**
 Contributing Editor: **Bill Roth**
 Contributing Editor: **Bill Dudney**
 Contributing Editor: **Michael Yuan**
 Founding Editor: **Sean Rhody**

Production
 Production Consultant: **Jim Morgan**
 Associate Art Director: **Tami Lima**
 Executive Editor: **Nancy Valentine**
 Associate Editor: **Seta Papazian**
 Research Editor: **Bahadır Karuv, PhD**

Writers in This Issue

Calvin Austin, Marc Domenig, Yakov Fain, Ashish Garg, Ashwini Garg, Jeremy Geelan, John Goodson, Rolf F. Kamp, Onno Kluyt, Matt Raible, Arthur Ryman, Brian Russell, Coach Wei, Joe Winchester

To submit a proposal for an article, go to <http://jdi.sys-con.com/main/proposal.htm>

Subscriptions

For subscriptions and requests for bulk orders, please send your letters to Subscription Department:

888 303-5282
 201 802-3012
subscribe@sys-con.com

Cover Price: \$5.99/issue. Domestic: \$69.99/yr. (12 Issues)
 Canada/Mexico: \$99.99/yr. Overseas: \$99.99/yr. (U.S. Banks or Money Orders) Back Issues: \$10/ea. International \$15/ea.

Editorial Offices

SYS-CON Media, 135 Chestnut Ridge Rd., Montvale, NJ 07645
 Telephone: 201 802-3000 Fax: 201 782-9638

Java Developer's Journal (ISSN#1087-6944) is published monthly (12 times a year) for \$69.99 by SYS-CON Publications, Inc., 135 Chestnut Ridge Road, Montvale, NJ 07645. Periodicals postage rates are paid at Montvale, NJ 07645 and additional mailing offices. Postmaster: Send address changes to: Java Developer's Journal, SYS-CON Publications, Inc., 135 Chestnut Ridge Road, Montvale, NJ 07645.

©Copyright

Copyright © 2005 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 Dorothy Gil, dorothy@sys-con.com. SYS-CON Media and SYS-CON Publications, Inc., reserve the right to revise, republish and authorize its readers to use the articles submitted for publication.

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

Newsstand Distribution Consultant
 Brian J. Gregory/Gregory Associates/W.R.D.S.
 732 607-9941, BJGAssociates@cs.com

Java and Java-based marks are trademarks or registered trademarks of Sun Microsystems, Inc., in the United States and other countries. SYS-CON Publications, Inc., is independent of Sun Microsystems, Inc. All brand and product names used on these pages are trade names, service marks or trademarks of their respective companies.



Who do you suppose registered their corporate Internet domain name first: Microsoft, Oracle, or Sun? The answer is Sun; it did so in 1986.

When in the early 1980s Dr. David Mills, John Postel, Zaw-Sing Su, and Dr. Paul Mockapetris were all involved in the development of the Domain Name System, known ever since by its initials DNS, their aim was to allow organizations to have meaningful names for paths to their systems, since by then computers had begun connecting to each other over wide area networks. However, it was unlikely that at the time any of those fine professionals ever had an inkling of what kind of unprecedented "land-grab" the system was destined to spawn.

The explicit purpose of domains was that they were to be administrative entities; they would divide the name management required of a central administration by assigning it to sub-administrations. That part of it worked. But domain names are something of a zero-sum game. If someone else owns Vatican.biz, then that's that – not even the Roman Catholic Church can register it unless they purchase it from the Alaska-based "Web marketing" company that does.

Inevitably, such anomalies and curiosities abound. Microsoft Corporation owns Microsoft.org, as you might expect; but the owner of Sun.org lives not in Santa Clara but in Tokyo – and it isn't Sun Microsystems. Another oddity: Microsoft has already renewed its corporate domain name through 2014, but IBM has at this writing only renewed IBM.com through 2006.

Notorious domain-name disputes have naturally been plentiful, usually involving "cyber-squatting." In 1998 two Texas men registered microsoftwindows.com and microsoftoffice.com, and Microsoft Corp filed suit, alleging that the two men were infringing on the company's trademarks and misleading the public. "They wanted between \$50,000 and \$100,000 at one time," Microsoft spokesman Adam Sohn said at the time. "The idea is that we weren't going to be subject to blackmail." It came as no surprise when it was revealed that

the same two men had registered a long list of domain names with the intent to sell them, including AlamoRentaCar.com and CitibankMasterCard.com.

Back on April 1 of this year, a Jacksonville, Florida-based writer called Roger Cadenhead purchased the rights to BenedictXVI.com, over two weeks before the announcement by Cardinal Joseph Ratzinger that he would be assuming the papacy under the name of Benedict XVI. Cadenhead actually purchased five other domain names too: ClementXV.com, InnocentXIV.com, LeoXIV.com, PaulVII.com, and PiusXIII.com. He risks the wrath of the world's 1.1 billion Catholics if he doesn't hand it over gracefully. (He claimed he bought the six domains as a "game.")

The 1999 federal law known as the Anti-Cybersquatting Consumer Protection Act (ACPA) authorizes a trademark owner to sue an alleged cybersquatter in federal court and obtain a court order transferring the domain name back to the trademark owner. In some cases, the cybersquatter must pay monetary damages. But it is way too late for those who already lost "their" name to a preexisting trademark.

Domain names in Javaland are somewhat skewed by the volcanic Indonesian island's understandable desire to have an Internet presence of its own quite separate from James Gosling's programming language. So while onjava.com takes you to articles about POJO application frameworks and comparisons of Spring and EJB 3.0, eastjava.com takes you to pictures and descriptions of Bromo Mountain, Ijen Crater, Batu City, and Madura Island. Likewise you can still go to, say, amazonwatch.org and find about the indigenous peoples of Brazil, Bolivia, Peru, Ecuador, Colombia, and Venezuela rather than just end up looking at yet another "Favorites" list about books that can be bought at Amazon.com.

So perhaps after all there is room for everyone in cyberspace. But do bear one staggering statistic in mind: in the .com top-level domain space alone, approximately 400,000 new names are registered in a typical 24-hour period, while 300,000 are deleted and 70,000 transferred. Cyberspace is no place for the faint of heart. ☛

Jeremy Geelan is group publisher of SYS-CON Media and is responsible for the development of new titles and technology portals for the firm. He regularly represents SYS-CON at conferences and trade shows, speaking to technology audiences both in North America and overseas.

jeremy@sys-con.com



DESKTOP



CORE



ENTERPRISE



HOME

Develop with intelligence and speed

Altova accelerates development and integration projects with software, services and solutions that enhance productivity and maximize results. Uncover why Altova is the smart choice of over 1.5 million application developers worldwide!

Investigate www.altova.com today.

JDJ contents

JDJ Cover Story



Rich-Client Portlets

by Marc Domenig

The half-object + protocol design pattern

54

FROM THE GROUP PUBLISHER

Sun Did It in 1986; Microsoft Took Longer

by Jeremy Geelan

3

JAVA ENTERPRISE VIEWPOINT

Getting Ready for a Java Technical Interview

by Yakov Fain

6

PATTERNS

Facading on the Fly

A framework proposal for improving network speed

by Ashish Garg and Ashwini Garg

8

XML

XML Rich-Client Technology Brings Zero-Install Rich Client Capabilities to J2EE

There is a way out from under .NET

by Coach Wei

12

WTP

Web Tools Platform: J2EE Development the Eclipse Way

The story of WTP 0.7, its scope, design principles, architecture, ecosystem, and plans

by Arthur Ryman

18

SOLUTIONS

Challenges in the J2EE Web Tier

How open source frameworks drive innovation

by Matt Raible

26

CORE AND INTERNALS VIEWPOINT

Mastering Mustang

by Calvin Austin

34

Q & A

'Being a Better Platform'

Interview with Mike Milinkovich, Executive Director, Eclipse Foundation

Interview by Jeremy Geelan

36

SPECIFICATIONS

JDBC 4.0: Features Worth the Wait

A significant advance on the standard

by John Goodson

46

YAKOV'S GAS STATION

Small Business Solutions – Part One

by Yakov Fain

50

DESKTOP JAVA VIEWPOINT

How Much Is that Buggie in the Program?

by Joe Winchester

52

LABS

Adding Charts to Web Applications

Give your Web app a facelift with WebCharts3D to make it a blockbuster

Reviewed by Yakov Fain

58

JSR WATCH

JCP Recognizes the Best Member, Spec Leads, and Specifications

by Onno Kluyt

62

Features



30

Back to Two Tiers and Plain JSP

by Brian Russell



40

Java Collections Framework and Managing Data

by Rolf F. Kamp



Yakov Fain
Contributing Editor

Getting Ready for a Java Technical Interview

If last September I was calling the Java job market healthy (see <http://java.sys-con.com/read/46228.htm>), today's market is hot. Once again recruiters are hungry and polite, but this doesn't mean you can easily get a new job. I'd like to share with you some rules and techniques that can increase your interview success rate.

The process of getting a job consists of three separate tiers; let's call it an *IPO* pattern:

- Getting the Interview
- Passing the interview
- Getting the Offer

I can't stress enough how important it is to work on achieving each of these goals separately, one step at a time! Your résumé is the most important entity of tier *I*. Adjust it for each position you are applying for (keep reading, I'm not asking you to lie). Make sure it's short and to the point (I've been developing software for more than 20 years and my résumé is only two pages long). If you are a Java programmer, nobody needs to know the details of that 10-year old PowerBuilder project. Don't even mention your work experience from the 1980s (this rule does not apply only to Bill Gates). Keep good notes of each thread of the *I* tier and always update your résumé based on the feedback.

The interview is scheduled and now your main goal is to make it a tier *P*, and not *F*. Do your homework and prepare a talk on some interesting and challenging Java problems that you might have experienced in one of your projects. If you didn't have any super complex projects, just pick up a topic from one of the multiple online Java forums. Remember, your interviewer has a difficult task: he or she needs to assess your technical skills within 30–60 minutes, so help! Try to get as many technical details about the job as possible from your recruiter. If they do Java sockets, research the work with non-blocking sockets; if they're into multi-threading, read about the new concurrent package offered in Java 5.0. For example, if you have prepared a talk on the internals of the Java garbage collector, you're

not allowed to leave the interview without talking about this. But what if the interviewer won't ask you about GC? It doesn't really matter. Find a way to switch the conversation to GC and do your best. The interviewers are happy because they don't need to think what to ask next, and you're happy because you've had a chance to talk about a well-known subject. Will this technique work all the time? No. But it'll work most of the time.

If you're a junior developer, spend some time answering the multiple-choice type of tests that are usually required for certification exams. You don't need to get certified, but all these books and online mock tests will help you pass similar tests offered by some job agencies. Find some sample interview questions online (I've also prepared a sample set for you at <http://java.sys-con.com/read/48839.htm>).

Here's another tip: during the interview don't critique the application architecture of your potential employer. You'll have plenty of chances to provide technical advice after (and if) you're hired, so just focus on getting an offer.

During the interview be energetic and show your interest in this job. Even if you are a Java guru, don't behave as if you're doing them a favor just by coming for an interview. Personality matters. People don't like prima donnas.

Tier *O*: you've got an offer! Now think hard if you want to accept the offer or turn it down. Have I ever mentioned that you should look for a new job not when your employer decides to let you go or your contract ends, but when you have a stable job, the sky is blue, and the grass is green? This gives you a tremendous advantage: you can consider the offer without being under pressure from unpaid bills. Don't accept an offer just because the new job pays an extra \$5,000 a year, which translates into less than \$300 a month after taxes. But do accept the offer that will give you a chance to work with interesting technologies or business applications even if it won't pay you an extra dime. Take charge of your career and actively build it the way you want. ☛

Yakov Fain is a J2EE architect and creator of seminars "Weekend with Experts" (www.weekendwithexperts.com). He is the author of the best-selling book *The Java Tutorial for the Real World* and an e-book *Java Programming for Kids, Parents and Grandparents*. Yakov also authored several chapters for *Java 2 Enterprise Edition 1.4 Bible*.
yakovfain@sys-con.com

President and CEO:

Fuat Kircaali fuat@sys-con.com

Vice President, Sales and Marketing:

Grisha Davida grisha@sys-con.com

Group Publisher:

Jeremy Geelan jeremy@sys-con.com

Advertising

Senior Vice President, Sales and Marketing:

Carmen Gonzalez carmen@sys-con.com

Vice President, Sales and Marketing:

Miles Silverman miles@sys-con.com

Advertising Sales Director:

Robyn Forma robyn@sys-con.com

National Sales and Marketing Manager:

Dennis Leavey dennis@sys-con.com

Advertising Sales Manager:

Megan Mussa megan@sys-con.com

Associate Sales Managers:

Dorothy Gil dorothy@sys-con.com

Kim Hughes kim@sys-con.com

Editorial

Executive Editor:

Nancy Valentine nancy@sys-con.com

Associate Editor:

Seta Papazian seta@sys-con.com

Production

Production Consultant:

Jim Morgan jim@sys-con.com

Lead Designer:

Tami Lima tami@sys-con.com

Art Director:

Alex Botero alex@sys-con.com

Associate Art Directors:

Abraham Addo abraham@sys-con.com

Louis F. Cuffari louis@sys-con.com

Assistant Art Director:

Andrea Boden andrea@sys-con.com

Video Production:

Frank Moricco frank@sys-con.com

Web Services

Information Systems Consultant:

Robert Diamond robert@sys-con.com

Web Designers:

Stephen Kilmurray stephen@sys-con.com

Percy Yip percy@sys-con.com

Vincent Santaiti vincent@sys-con.com

Accounting

Financial Analyst:

Joan LaRose joan@sys-con.com

Accounts Payable:

Betty White betty@sys-con.com

Accounts Receivable:

Gail Naples gailn@sys-con.com

SYS-CON Events

President, SYS-CON Events:

Grisha Davida grisha@sys-con.com

National Sales Manager:

Jim Hanchrow jimh@sys-con.com

Customer Relations

Circulation Service Coordinators:

Edna Earle Russell edna@sys-con.com

Linda Lipton linda@sys-con.com

JDJ Store Manager:

Brunilda Staropoli bruni@sys-con.com

ELIMINATE PERFORMANCE ISSUES AND RELATED ANXIETY.

**FOR RELIEF OF STRESS DUE TO SUBOPTIMAL PERFORMANCE
JBuilder IS PROVEN EFFECTIVE ACROSS THE APPLICATION LIFECYCLE.**

JBuilder® from Borland® relieves the stress of performance anxiety associated with Application Development Dysfunction. A powerful component of Software Delivery Optimization (SDO) from Borland, JBuilder rapidly improves* individual performance and productivity. But it doesn't stop there. Thanks to deep integration across the entire Application Lifecycle, JBuilder can help your entire team create better software, faster. Don't let performance issues keep you from success.

Borland®



ASK BORLAND IF SDO IS RIGHT FOR YOU. borland.com/jbuilder

* JBuilder is indicated for organizations that wish to attain real competitive advantage in the marketplace. Side effects include increased margins, greater customer satisfaction and improved efficiency.
© 2005 Borland Software Corporation. All rights reserved. All Borland brand names are trademarks or registered trademarks of Borland Software Corporation in the United States and other countries. 23389

Facading on the Fly

by Ashish Garg and Ashwini Garg

A framework proposal for improving network speed

Network speed has improved tremendously over the years and has revolutionized enterprise computing, but even with today's network infrastructure sending messages across a network is of several orders slower than sending messages locally. The latency caused by the network is a function of the size of the messages and the number of round trips. The delay due to message size is more or less constant as long as the data size stays within the size of the buffer transmission but cutting down the network crossings could have a more significant and direct impact. In light of these considerations, this article proposes the design and implementation of a framework, using core Java principles like Reflection, Dynamic Proxies, Byte Code Engineering, Java Beans, and Thread Local, to club multiple network calls and have them execute at the server in one go.

Objective

In an *Enterprise Computing* model multiple users access applications and data stored on servers scattered across many locations. The applications offer low-level data in the form of services. But often, the user really needs a more consolidated and unified view rather than the low-level view these services offer. The *Session Façade* pattern helps by logically grouping services keeping in mind the business needs of the user. In doing so the façade serves two other purposes:

- It precludes the clients from having to access the fine-grained remote interfaces thus reducing network traffic and latency.
- It also decouples lower-level business components from one another allowing them to be conceived and built independently, making designs more flexible and comprehensible.

Thus the façade helps in getting over some of the limitations of network

speed but, since it's tied to the user's requirements, as more and more users start using the applications, more and more facades have to be added. While this may sound simple, it often runs into the following impediments:

- Maintainability issues due to uncontrolled growth in the number of views
- Delays and disruptions of service for existing clients while incorporating new changes
- Departmental politics may lead to some facades being preferred over others

In such a scenario it would be useful to have a framework that would help clients define, develop, and deploy their facades and have them execute on the server.

Approach

The semantics of the Java language don't support clubbing network calls into batches so we are proposing a framework that imposes certain minimal but non-intrusive changes in the regular style of programming to help overcome the limitation.

The basic idea includes:

- Demarcating batch (façade) boundaries through a start and end call
- Objects used in the façade need to be batch-aware, i.e., object behavior is dependent on whether or not it's executing in a batch
- Delaying object state changes by postponing the execution of the code till the end of the batch
- A new framework bean to comprehend and service such requests is needed

Implementation

Overview

1. The client signals the start of the batch process by a call to begin the method of BatchContext. The call changes the state of the current thread by binding the BatchContext to a ThreadLocal variable.
2. Objects modulate their behavior based on the availability/unavailability of the BatchContext in the current thread context. Unavailability triggers the usual behavior as programmed. On the other hand, availability results in:

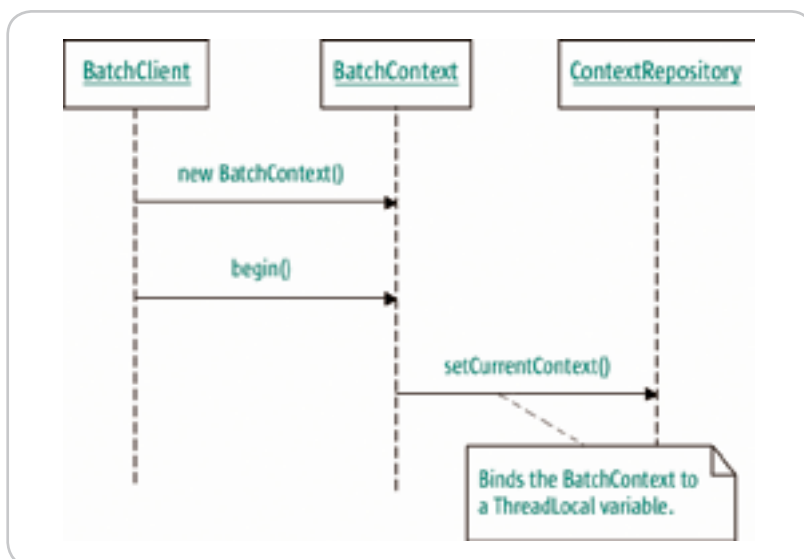


Figure 1 Batch initialization



Ashish Garg is a technical architect with Infosys Technologies Ltd. His interest area lies in Java and anything related.

ashish_garg@infosys.com



Ashwini Garg is a technical architect with Infosys Technologies Ltd. She has seven years of experience in client/server and J2EE technologies. Her main area of interest and expertise lies in architecting and designing J2EE applications.

ashwini.garg@gmail.com

- a. An addition to the *Participating-Objects* list – All the objects created in the batch boundary are added to the list of participating objects and indexed (indexing helps reduce the final cost of serialization). Similarly, calls to methods with a non-void return type results in instantiation and initialization of the returned object by a no-argument constructor. The returned object is added to this participating objects list. The call execution doesn't result in any state changes.
 - b. The serialization of invocation – The serialized instruction is stacked in the *InvocationHistory* object bound to the *BatchContext*. Different types of invocation objects exist for the lookup instruction (*LookupHomeInvocation*), remote object creation (*CreateRemoteInvocation*), remote method invocation (*RemoteMethodInvocation*), instance method invocations (*MethodInvocation*), and constructor invocations (*ConstructorInvocation*). The invocation object maintains a reference, via an index, to the object invoked, the arguments passed, and the object returned besides information about the method invoked, i.e., method name.
3. Use of *ServiceLocator* pattern for looking up the EJBs. The *ServiceLocator* acts as a batch-aware class; it registers the client's intent to do a lookup while the actual lookup happens later along with other network calls. The *ServiceLocator* is described in greater detail in the following sections.
 4. The client then signals the end of the batch process. This is done by a call to the *end* method of the *BatchContext* class. The call results in the entire list of objects and the instructions being submitted to the server. The results of the entire sequence are processed and returned together as one call.

Details

The following section describes in detail the internals of a batch call.

1. Batch begin – The events accompanying a batch begin call are shown in Figure 1.
2. Initialization of a Data Object in the batch call – The object is simply added to the list of Participating Objects. The conditional behavior of

the service class's construction isn't the default behavior as programmed by the developer; instead it's injected post-build time through bytecode manipulation and is discussed in the components section later.

3. Looking up an EJB using the modified *ServiceLocator* – In the batch mode the *ServiceLocator* would add a *LookupHomeInvocation* instruction and return a *DynamicProxy* implementation of the *EJBHome* and *Home Interface* as the home proxy. Returning a proxy implementation helps us trap and handle invocations to the home and remote interfaces.
 4. Calls on the Home interface – The home proxy (through its handler, *EJBHomeHandler*) in turn, on a
- create* call on home proxy, adds a *CreateRemoteInvocation* instruction and returns a *DynamicProxy* implementation of the *EJBObject* and the *Remote Interface* as the remote proxy.
 5. Calls on the Remote interface – Calls on the remote proxy (through its handler, *EJBRemoteHandler*) translate to a *RemoteMethodInvocation* instruction and the return of a dummy instance of the relevant return type.
 6. Call to the *end* method of the *BatchContext* class – This call results in the entire list of objects and the instructions being submitted to the server. The server runs through the instructions sequentially and executes them. In the process it alters the state of the objects participat-

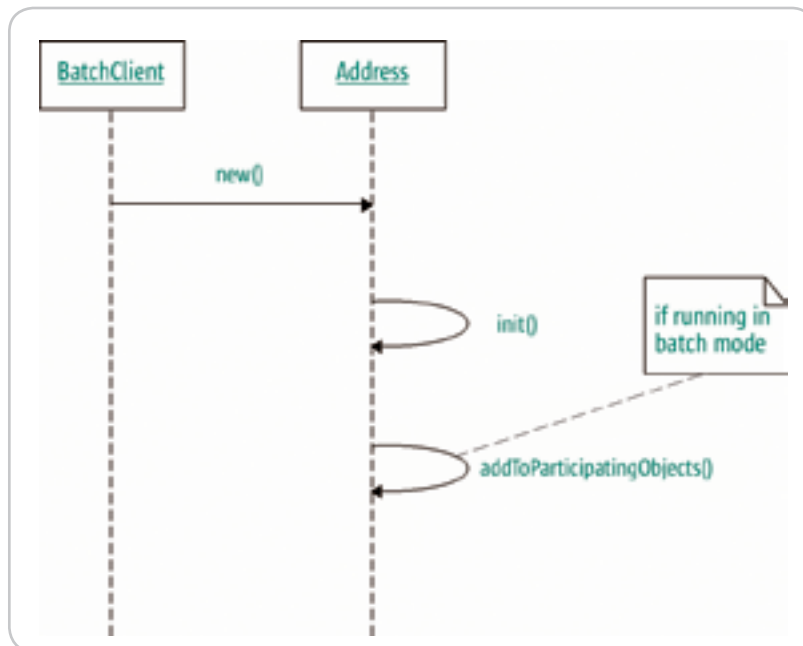


Figure 2 Instantiating an object

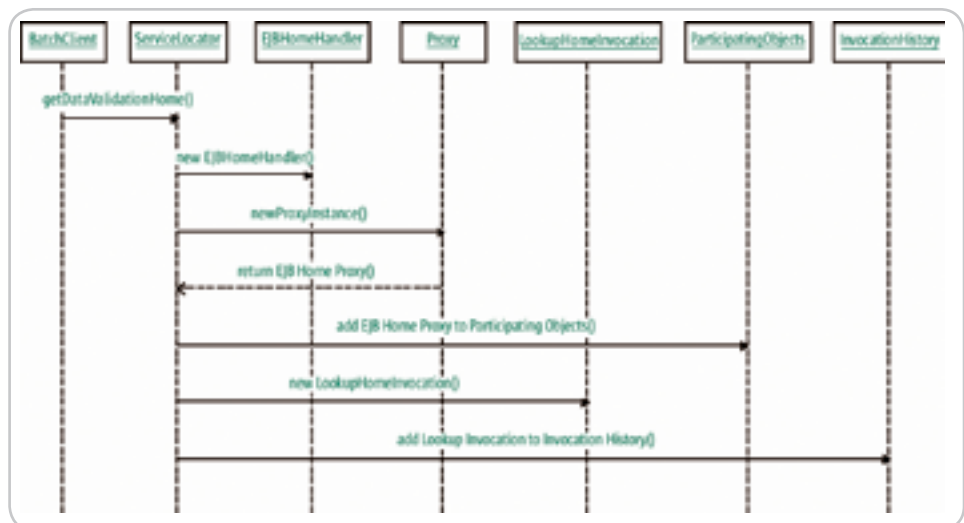


Figure 3 Looking up a Remote Service

ing in the batch. The altered state is returned to the client where the original references are used to modify the client-side object states using accessor methods to reflect the new states. The lack of pointers restricts the return types to mutable types. Apache Commons *BeanUtils* libraries are used to facilitate copying the states.

Server-side Components for Build and Runtime Support

The approach suggested is not immune to server-side changes, but the changes are simple, static, and one time. The following are the changes:

- A new framework bean to run the instructions, piled on the client-side, on the server-side
- Batch-aware data objects
- A batch-aware *ServiceLocator* class

The injection of batch-aware behavior is automated and provided as a simple Ant task (*InstrumentTask*). The Ant task runs through the server classes, examines them and does the following:

- Pairs the Remote against the Home classes and looks through the class definitions to identify the data objects and the exception objects to be included in the client libraries.
- Batch-aware conditional behavior is injected into data object definitions at the time the client library is prepared through a bytecode engineering step. The batch-aware data objects are generated by modifying the existing class

definitions using the Apache Byte Code Engineering Library (BCEL). The modified class renames the existing PUBLIC methods with a '_' in front of the original name. New methods are added with the original signatures. The new methods implement the standard runtime conditional behavior as discussed earlier. Similar changes are made to the constructor with special handling for super call.

- A batch-aware factory based on the J2EE *ServiceLocator* pattern is generated and included with the client-side libraries. This is necessary since there's no way to alter the behavior of the system-level classes like *Context* used during lookup. Besides, having a *ServiceLocator* for caching is a normal practice. The *ServiceLocator* will have get methods for all the EJBs available on the server.

Challenges and Restrictions

- Only user-defined classes can participate in the batch since they are the only ones made batch-aware.
- Primitive types aren't allowed for the above reasons and their inability to provide hooks to constructor calls.
- Objects supported by *BeanUtils copyProperties* methods are supported. In general, Java Bean types are supported.
- No looping constructs or conditional paths are possible in batch since they can't be intercepted and their behavior can't be made conditional at runtime. But new batch-aware constructs to support similar behavior can be created.
- Batch boundaries can't span across thread boundaries since threads carry the batch-aware status.

Performance

Elementary performance runs were done on a hypothetical problem from transportation and courier domains in both batch and non-batch mode. The results were averaged out over multiple calls to even out the one-time costs of both approaches. We wouldn't want to benchmark the results since the tests weren't done in a lab-type environment, but a definite improvement in time was observed even on very fast LANs. The source code for this article can be downloaded from <http://jdj.sys-con.com>.

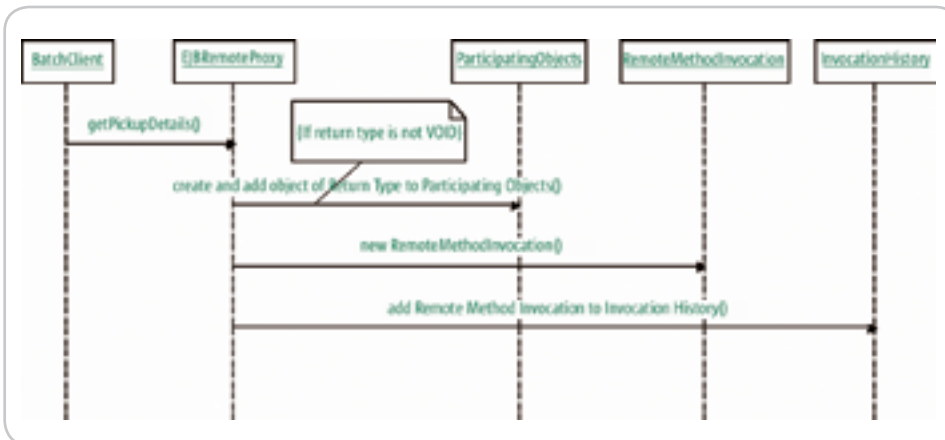


Figure 4 A Remote Invocation

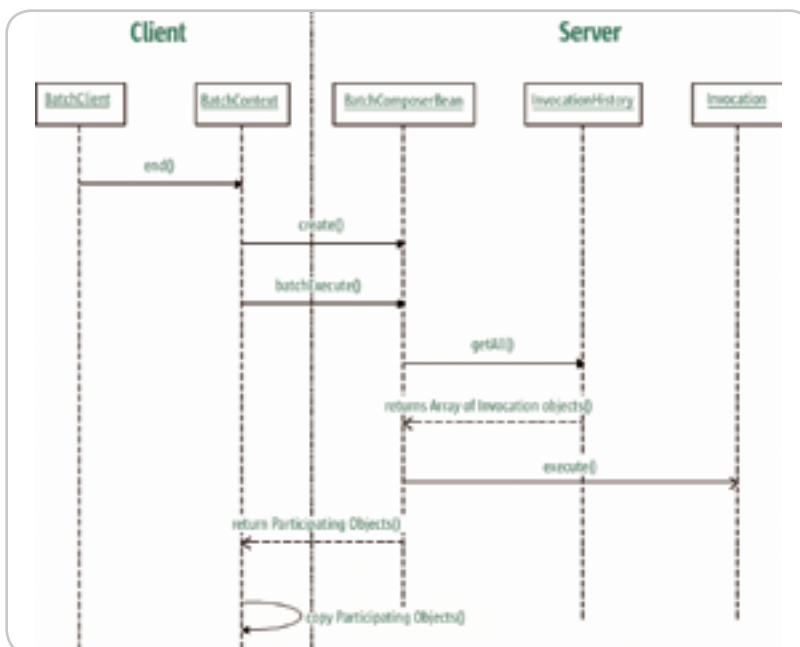


Figure 5 Going to the server

Perforce SCM.

Piece your global development team together.



Perforce Software Configuration Management brings you and your team together - wherever you happen to work.

[Fast]

[Scalable]

[Distributed]

You've got developers in Tokyo, Denver and London collaborating on your project. Your UI designer is in Austin and your QA team is in Cupertino. SCM nightmare? No. Just another day's work for Perforce.

Perforce Proxy, part of Perforce's multi-site solution, brings files to remote users on demand and caches them locally for other users at the same location. Local caching guarantees quick response times for remote users while maintaining real-time access to project activity and status information.

Unlike replication solutions with their inherent administrative overhead, a Perforce multi-site solution recovers files automatically, reduces the load on the Perforce Server, and requires no remote backups.

Fully integrated into Perforce's client/server architecture, Perforce Proxy requires only TCP/IP connectivity. A single Perforce Server can support any number of Perforce Proxy sites, allowing developers all over the world to collaborate.

PERFORCE
SOFTWARE

Download a free copy of Perforce, no questions asked, from www.perforce.com. Free technical support is available throughout your evaluation.

All trademarks used herein are either the trademarks or registered trademarks of their respective owners.

XML Rich-Client Technology Brings Zero-Install Rich Client Capabilities to J2EE

There is a way out from under .NET

by Coach Wei

Which platform to use Java or .NET? Developers ask this question all the time. Java has been widely adopted because of its overwhelming benefits on the server side, but Java has less to offer on the client side. .NET has made inroads into the enterprise by leveraging its stronger rich-client capabilities. An alternative solution for enterprise-scale Internet application development is the emerging XML-based rich-client technology.

.NET Erosion from the Client Side

There are good reasons why Java is the platform of choice for server-side computing. J2EE is an open standards-based platform that enables open integration. Java enjoys broad industry support, including vendors like IBM, Sun, and Oracle, as well as upstarts like Nexaweb and Sonic Software. J2EE is cross-platform, giving customers the freedom to deploy in different environments. It has proven enterprise strength. By comparison, .NET has obvious limitations. It is limited to Windows deployment only; it's a single-vendor solution, and lacks industry support from other vendors to meet enterprise requirements.

On the other hand, .NET has stronger client-side capabilities than Java. Java AWT is based on an architecture that offers limited out-of-box capabilities. Java Swing offers better out-of-box functionality, but it's complex and difficult to use. It is possible to develop Java applications with a rich look-and-feel using AWT or Swing, but complexity and developer skills requirements are high. By contrast, the barriers to developing strong .NET client applications are lower. Your typical corporate developer can easily write sophisticated VB.NET desktop applications with a professional look-and-feel.

Another option is to develop thin-client applications using HTML. HTML applications are "zero-install" thin-client applications, while both Java and .NET client apps have a heavy client-side footprint and require a significant download. Besides solving network bandwidth issues, "zero-install" translates directly into lower maintenance and support costs. The skill set requirements and complexity of HTML is much lower than either .NET and Java. Unfortunately, HTML isn't suitable for handling the level of complexity, scale, and time-sensitivity required by enterprise programs. For applications with non-linear workflow, complex integration, large data sets, or time criticality, Java or .NET rich clients have been the only viable options.

Most business applications are user-oriented. Client-side issues such as look-and-feel, richness, and performance directly impact business user productivity. As a result, client-side choice can influence server-side architecture decisions. Because Java doesn't have a compelling solution on the client side, many applications are written with .NET. Eventually this could lead to a greater adoption of .NET for easier integration and management, eventually eroding J2EE's market share.

XML Rich Client

Now there's an alternative. XML rich client-technology serves rich-client applications on-demand by using XML from J2EE or Service Oriented Architecture (SOA) environments. Using XML rich-client technology, J2EE can deliver enterprise Java applications with higher performance than .NET – not only visually and functionally richer, but easier to deploy and maintain, with a thinner footprint,

faster performance, greater scalability, and lower complexity.

Sample XML-Based Rich Client Technology Code

With XML rich-client technology, Web developers can build and deploy Web applications with the same richness and performance as the best Java Swing or .NET client apps without losing the "zero-install" and "universal delivery" advantages of HTML. The difference is that an XML-based rich-client application would send out XML to the client side, which is processed by the software's client, instead of HTML being processed by a browser, as in a normal Web application.

Figure 1 shows a sample form. The code behind it is as follows:

```
<?xml version="1.0" encoding="UTF-8"?>
<dialog height="212" title="Sample Form"
width="275">
  <layoutmanager
```

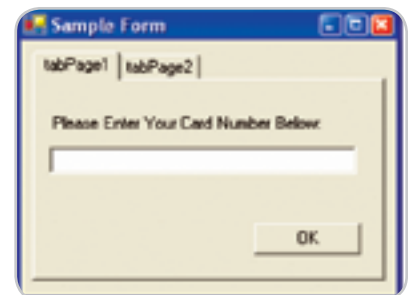


Figure 1 Sample rich-client form

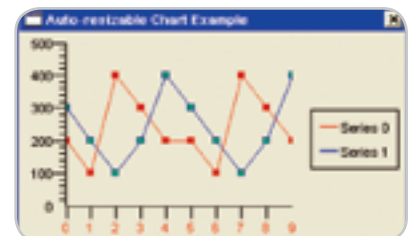


Figure 2 Sample chart



Coach Wei is CTO of Nexaweb (www.nexaweb.com), which has developed an XML-based rich-client technology platform for building and deploying enterprise Internet applications. He previously played a key role at EMC Corporation in developing a new generation of storage network management software. Coach is a graduate of MIT, holds several patents, and is an industry advocate for the proliferation of open standards.

coach@nexaweb.com

The Developer Paradox:

No time to test your code?

But spending **long hours** reworking it & resolving errors?



Check out **Parasoft Jtest® 7.0**

Automates Java testing and code analysis.

Lets you get your time back and deliver quality code with less effort.

■ **Automated:**

Automatically analyzes your code, applying over 500+ industry standard Java coding best practices that identify code constructs affecting performance, reliability and security.

Automatically generates and executes JUnit test cases, exposing unexpected exceptions, boundary condition errors and memory leaks and evaluating your code's behavior.

Groundbreaking test case "sniffer" automatically generates functional unit test cases by monitoring a running application and creating a full suite of test cases that serve as a "functional snapshot" against which new code changes can be tested.

■ **Extendable:**

Industry standard JUnit test case output make test cases portable, extendable and reusable.

Graphical test case and object editors allow test cases to be easily extended to increase coverage or create custom test cases for verification of specific functionality.

■ **Integrated:**

Integrates seamlessly into development IDE's to support "test as you go" development, and ties into source control and build processes for full team development support.

To learn more about Parasoft Jtest or try it out, go to www.parasoft.com/Jtest



Automated Software Error Prevention

```

layout="nulllayout"/>
    <tabbox borderstyle="null"
height="168" width="248" x="8"
y="8">
        <tab text="tabPage1">
            <panel>
                <layoutmanager
layout="nulllayout"/>
                <label height="23"
text="Please Enter Your Card
Number Below:" width="224" x="6"
y="24"/>
                <textbox height="20"
text="" width="216" x="8"
y="48"/>
                <button height="25"
text="OK" width="60" x="170"
y="100"/>
            </panel>
        </tab>
        <tab text="tabPage2"/>
    </tabbox>
</dialog>

<series points="300,200,100,200,400,300,20
0,100,200,400"
stroke="blue" type="line"/>
</content>
<valueaxis/>
<labelaxis fgcolor="red"/>
<legend/>
</chart>
</dialog>

```

The Benefits of XML Rich Client Technology

- **The Future is XML** – The rapid adoption of XML for server-side computing makes XML a natural candidate for client-side computing. Even Microsoft has made a commitment to this trend. The next-generation Windows, Longhorn, will support the use of XML for Windows desktop applications.
- **XML Rich-Client Technology Significantly Lowers Application Complexity and Skill Set Requirements** – One benefit of XML rich-client technology is that it enables enterprise-class Internet applications while lowering development/maintenance complexity. XML rich-client technology uses the power of XML, which is more efficient and extensible than procedural programming languages like Java, C#, C++, or JavaScript. Code created in XML is simpler, takes up fewer lines, and is easier to read and process. Someone with HTML skills can easily understand the code, while C# code and Java Swing code require intimate

knowledge of Object-Oriented programming.

Take the form below as an example. To create this form takes about 70 lines of C# code using Visual Studio .NET, but only 17 lines of XML code using an XML-based rich-client technology. On top of that, XML code is much easier to read, and can be processed and understood by someone with HTML skills. Both C# code and Java Swing code require intimate knowledge of Object-Oriented programming. The skill set requirement is considerably higher.

Here is the Visual Studio.NET C# code needed to create the form previously shown in Figure 1:

```

this.tabControl1 = new System.Windows.Forms.
TabControl();
this.tabPage1 = new System.Windows.Forms.
TabPage();
this.tabPage2 = new System.Windows.Forms.
TabPage();
this.button1 = new System.Windows.Forms.
Button();
this.textBox1 = new System.Windows.Forms.
TextBox();
this.label1 = new System.Windows.Forms.
Label();
this.tabControl1.SuspendLayout();
this.tabPage1.SuspendLayout();
this.SuspendLayout();

this.tabControl1.Controls.Add(this.tab-
Page1);
this.tabControl1.Controls.Add(this.tab-
Page2);
this.tabControl1.Location = new System.
Drawing.Point(8, 8);
this.tabControl1.Name = "tabControl1";
this.tabControl1.SelectedIndex = 0;
this.tabControl1.Size = new System.Drawing.
Size(248, 168);
this.tabControl1.TabIndex = 0;

```

```

this.tabPage1.Controls.Add(this.label1);
this.tabPage1.Controls.Add(this.textBox1);
this.tabPage1.Controls.Add(this.button1);
this.tabPage1.Location = new System.Drawing.
Point(4, 22);
this.tabPage1.Name = "tabPage1";
this.tabPage1.Size = new System.Drawing.
Size(240, 142);
this.tabPage1.TabIndex = 0;
this.tabPage1.Text = "tabPage1";

this.tabPage2.Location = new System.Drawing.

```

Figure 2 shows a sample chart. In normal Web apps, such charts are generated as static GIF images that are sent to the browser for display. With XML-based rich-client technology, XML can be sent directly to the client:

```

<?xml version="1.0" encoding="UTF-8"?>
<dialog title="Auto-resizable Chart Example"
bordercolor="blue"
width="534" height="325">
    <layoutmanager layout="borderlayout"/>
    <chart layoutpos="center">
        <content>
            <series points="200,100,400,300,200,10
0,400,300,200"
stroke="red" type="line"/>

```

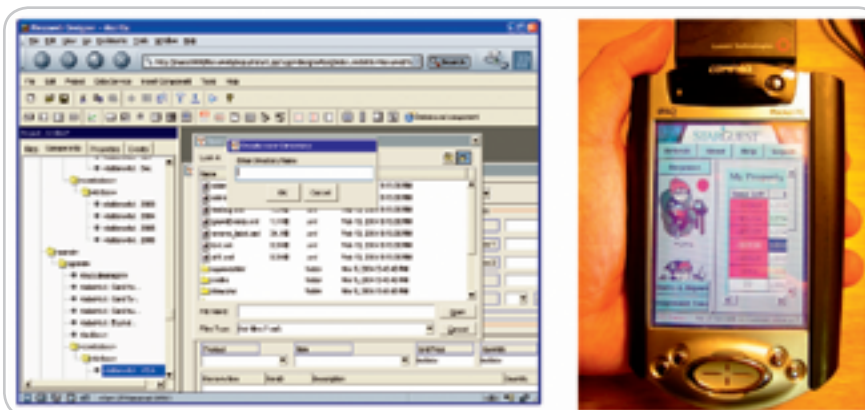


Figure 3 Sample J2EE/XML rich-client Web applications that are indistinguishable from the best client/server applications.

On the left, a Web application runs inside a Mozilla browser with the same richness as Windows desktop applications.

On the right, a customer service application runs on a PDA device.



Pure **Java**
Pure **Excel**
Power for users
Control for IT
Pure **Paradise**

Formula One e.Spreadsheet Engine

*API-driven, embedded, 100% pure-Java,
scalable spreadsheet toolset*

Spreadsheets play an essential role in the business world. And now, they can also perform a vital function in your Java applications. How? With the Formula One e.Spreadsheet Engine, an API-driven, 100% pure Java toolset for embedding Excel spreadsheet functionality in your applications.

Excel-enable your Java applications

Use a state-of-the-art graphical development environment to build fully-formatted Excel report templates that include formulas, functions, colors, merged cells, even charts. It's fast, easy and effective.

Embed live, Excel-compatible data grids

Include interactive Excel forms and reports in your Java applications. Let users manipulate them using their spreadsheet skills and then commit the changes to databases or applications – or save them as an XLS file on their desktops.

Automate complex calculations and rules

Embed Excel spreadsheets that automate complex calculations and business rules on J2EE servers. Stop translating spreadsheet logic into Java code today, and start leveraging the development skills of your organization's spreadsheet experts.

Read and write server-based Excel spreadsheets

Give your users server-based spreadsheets populated with up-to-the-minute information directly from data-bases, XML files and enterprise applications. Get control of runaway data warehouses and spreadmarts now.

Make spreadsheets a part of your strategies

Visit us today at www.reportingengines.com to request a **free trial** of the e.Spreadsheet Engine. We'll show you how to make Excel spreadsheets a vital and productive part of your enterprise computing strategies.

ACTUATE
ReportingEngines

www.reportingengines.com
sales@reportingengines.com
888-884-8665 + 1-913-851-2200

**FREE TRIALS,
DEMOS AND
SAMPLE CODE**


```

Point(4, 22);
this.tabPage2.Name = "tabPage2";
this.tabPage2.Size = new System.Drawing.
Size(288, 230);
this.tabPage2.TabIndex = 1;
this.tabPage2.Text = "tabPage2";

this.button1.Location = new System.Drawing.
Point(152, 104);
this.button1.Name = "button1";
this.button1.TabIndex = 0;
this.button1.Text = "OK";

this.textBox1.Location = new System.
Drawing.Point(8, 48);
this.textBox1.Name = "textBox1";
this.textBox1.Size = new System.Drawing.
Size(216, 20);
this.textBox1.TabIndex = 1;
this.textBox1.Text = " ";

this.labell.Location = new System.Drawing.
Point(8, 24);
this.labell.Name = "labell";
this.labell.Size = new System.Drawing.
Size(224, 23);
this.labell.TabIndex = 2;
this.labell.Text = "Please Enter Your Card
Number Below:";

this.labell.Click += new System.
EventHandler(this.labell_Click);

this.AutoScaleBaseSize = new System.
Drawing.Size(5, 13);
this.ClientSize = new System.Drawing.
Size(264, 187);

```

```

this.Controls.Add(this.tabControl1);
this.Name = "Form1";
this.Text = "Sample Form";
this.tabControl1.ResumeLayout(false);
this.tabPage1.ResumeLayout(false);
this.ResumeLayout(false);

```

• **XML Rich Client Technology Enables Richer, Thinner, Faster Applications** – Enterprise Internet Applications built using XML rich-client technology can be normal J2EE Web applications, but send XML to the client side instead of HTML. However, because of XML rich-client technology, they are richer, faster, and consume up to 90% less bandwidth. No client installation is required. These applications can run instantly on different browsers and PDA devices with the same functionality as Windows desktop applications, with user interface elements like multiple windows, menu bars, toolbars, and hierarchical trees.

For example, an enterprise management “dashboard” application built with XML rich-client technology can use sliding tabs to display tabular data as well as rich graphics and interactive charts, all updated in real-time via server push without the clunky “click-refresh” associated with normal HTML applications.

Applications of this nature are usually built using .NET or Win32 and require heavy download/installation. Using XML rich-client technology, they are lightweight, zero-install Web applications with lower bandwidth consumption and better performance.

• **XML Rich Client Technology Seamlessly Extends J2EE** – With XML rich-client technology, developers can still use JSP, tag libraries, servlet, struts, and other approaches for server-side presentation, EJB for server-side business logic, and any persistency layer for data storage. The application is deployed as a normal WAR/EAR file and managed as a normal Web application.

Conclusion

Combining XML rich-client technology with J2EE provides the following benefits compared to .NET:

- Enterprise-scale rich-client capability and complex workflow, scalable for large data sets and high transaction rates such as hundreds of messages per second.
- Zero-install capability with deployment/management advantages similar to HTML. While .NET applications can only be deployed to Windows XP desktops, XML rich client technology applications can be deployed hassle-free to over 95% of all desktops with any 4.0+ browser.
- XML rich-client technology working in concert with J2EE gives Web applications the “out-of-box” capability to seamlessly enable server push, reliable messaging, pub/sub, broadcasting and guaranteed order delivery. Built-in compression, incremental update, and distributed state management minimize network traffic and increase performance.

Using XML rich-client technology, companies can develop enterprise Internet applications that match desktop quality with higher performance, and still enjoy a J2EE server infrastructure: centrally managed and deployed, automatically updated, bandwidth and network efficient, with enterprise-level J2EE security, scaling, and broad industry support. ☛

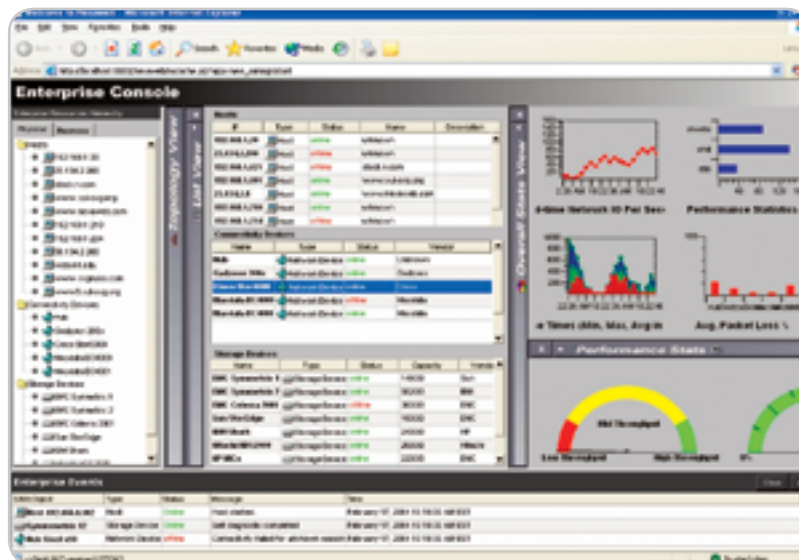


Figure 4 An enterprise management J2EE Web application that uses XML-based rich-client technology to achieve the same richness and functionality as heavy client/server applications. XML data is presented as trees, tables, and rich graphics in real-time without sending static “GIF” images to the client side.

Innovations by InterSystems



Real-Time Data Analytics With A Real-Fast Database.

Imagine being able to query a lightning-fast operational database in real time.

Now you can, with our multidimensional database for transaction processing and real-time analytics.

Only Caché combines robust objects and robust SQL, thus eliminating object-relational mapping. It requires little administration, delivers speed and scalability on minimal hardware, and comes with a rapid application development environment.

These innovations mean faster time-to-market, lower cost of operations, and higher application performance. We back these claims with this money-back guarantee: *Buy Caché for new application development, and for up to one year you can return your license for a full refund if you are unhappy for any reason.**

Innovative database. Guaranteed performance.

InterSystems
CACHÉ™



Rapid Integration Platform Makes Applications Perform Together.

Imagine being able to get your applications to perform together as an ensemble. Easily.

Now you can, with our universal integration platform.

Ensemble is the first fusion of an integration server, data server, application server, and portal development software – in a single, seamless product. This is the complete ensemble of technologies needed for rapid integration, fast development, and easy management.

These innovations mean all of your integration projects will be completed on time and on budget, with a simplified learning curve for your IT staff. We back these claims with this money-back guarantee: *For up to one year after you purchase Ensemble, if you are unhappy for any reason, we'll refund 100% of your license fee.**

Innovative integration. Guaranteed performance.

InterSystems
ENSEMBLE™

For a free copy of CACHÉ, or to request a free ENSEMBLE proof-of-concept project, visit www.InterSystems.com/Free8P

*Read about our money-back guarantees at the web page shown above.

© 2005 InterSystems Corporation. All rights reserved. InterSystems Caché and InterSystems Ensemble are trademarks of InterSystems Corporation. 5-05 ComboInno8JaDeJo

Web Tools Platform: J2EE Development the Eclipse Way

by Arthur Ryman

The story of WTP 0.7, its scope, design principles, architecture, ecosystem, and plans

The Eclipse Open Source Integrated Development Environment (IDE) (see <http://eclipse.org>) is rapidly gaining popularity among Java developers primarily because of its excellent Java Development Tools (JDT) and its highly extensible plug-in architecture. Extensibility is, in fact, one of the defining characteristics of Eclipse. As the Eclipse home page says, "Eclipse is a kind of universal tool platform – an open extensible IDE for anything and nothing in particular." Although Eclipse is itself a Java application, all tools, including JDT, are on an equal footing in that they extend the Eclipse platform via well-defined extension points.

Of course, an infinitely extensible, but empty, platform might be interesting to tool vendors, but very boring for developers. Therefore, the initial version of Eclipse came with the JDT and the Plug-in Development Environment (PDE), both examples of how to extend the platform and very useful tools in their own right. JDT supported J2SE development while PDE supported Java-based Eclipse plug-in development. The combination of JDT and PDE fueled the creation of thousands of commercial and Open Source plug-ins for Eclipse, many of which supported J2EE development. For example, IBM released Eclipse-based commercial J2EE products, including WebSphere Studio Application Developer, and Rational Application Developer, while eteration, JBoss, Genuitec, Exadel, and Innoo pract among others, released Open Source offerings. However, the profusion of J2EE plug-ins made it difficult for vendors to build on each other and for users to assemble an integrated suite of tools. For example, each J2EE toolset had its own way to support application servers.

As the popularity of Eclipse grew, it became apparent that the next logical step in its evolution was to add platform support for J2EE. This support would provide a common infrastructure for all J2EE plug-ins, with the goal of improving tool integration, reducing plug-in development expense, and simplifying the J2EE development experience for Eclipse users.

In June 2004, based on a proposal from IBM, the Eclipse Management Organization (EMO) agreed to create a new top-level project, the Web Tools Platform (WTP). However, it was believed that for WTP to be truly successful it needed a broad base of vendor support. A search began to engage additional vendors to partner with IBM. WTP was discussed in a BOF session at

the first EclipseCon conference held in February 2004, and ObjectWeb agreed to lead the project creation effort. ObjectWeb assembled a set of vendors to join the project and agreed to co-lead the Project Management Committee (PMC). WTP was formally launched in June 2004 based on initial contributions from eteration, Lomboz, and IBM Rational Application Developer.

WTP got further industry endorsement earlier this year when BEA joined the project and announced plans to base a future version of WebLogic Workshop on it. BEA co-leads the PMC along with ObjectWeb. At this year's EclipseCon, Sybase announced the Data Tools Project (DTP), which will add to the data tools in WTP and create a platform layer dedicated to



Arthur Ryman Arthur Ryman is the leader of the Eclipse WTP Web Standard Tools subproject. He is a software development manager and architecture at the IBM Toronto Laboratory where he has spent the last 10 years working on J2EE development tools. He is an adjunct professor of computer science at York University, a senior member of the IEEE, and a member of the IBM Academy of Technology. He holds a PhD in mathematics from Oxford University. Arthur is a co-author of the book *Java Web Services Unleashed*, and is currently working on a new book with co-authors Naci Dai and Lawrence Mandel about WTP titled *Java Web Application Development with Eclipse*.

ryman@ca.ibm.com

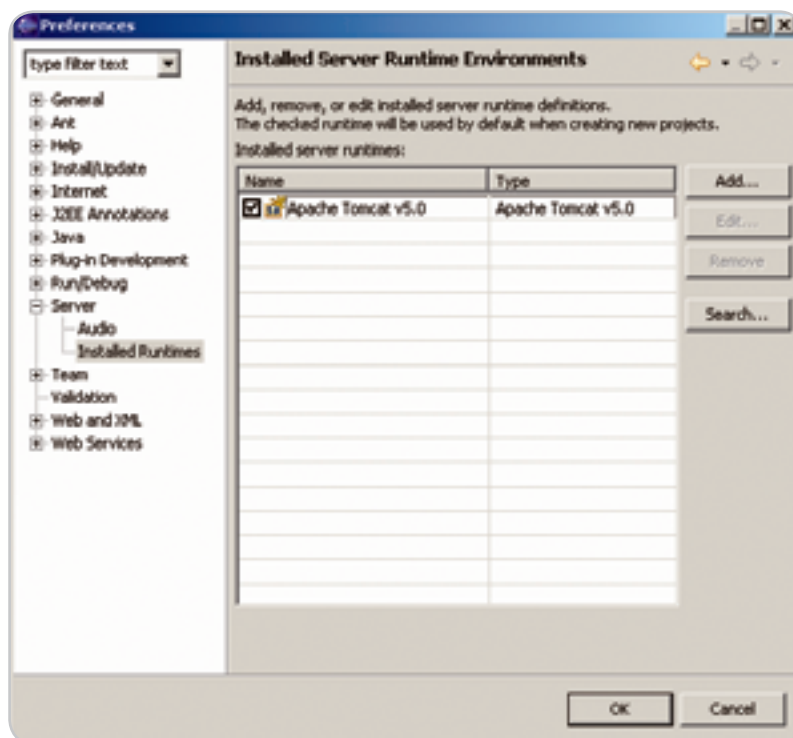


Figure 1 Server preferences page

database access. Oracle and Borland also announced Eclipse projects closely related to WTP. With major vendors such as IBM, BEA, Borland, Oracle, and Sybase all co-operating on a shared Open Source tool infrastructure, the center of gravity for J2EE tools has clearly shifted to Eclipse.

WTP 0.7 development is now well underway and has released a series of milestone drivers that can be downloaded from <http://eclipse.org/webtools>. The final release of WTP 0.7 is on track for a July 2005 delivery. The rest of this article gives you an overview of WTP, its scope, design principles, architecture, ecosystem, and plans.

A Quick Tour of WTP

One way to understand WTP is that it extends Eclipse along two dimensions, namely execution environments and artifact types. The execution environment dimension defines where code runs. Out-of-the-box, Eclipse lets you develop Java main programs that run in a command shell, applets that run in a Web browser, JUnit tests that run in a JUnit runner, and ANT tasks that run in ANT. WTP extends Eclipse by adding servers in general, and both J2EE and database servers in particular, as new execution environments. In general, you need to install an execution environment, configure it in Eclipse, and associate it with development artifacts that you want to run in it.

The development artifact dimension defines what developers create. Obviously, Eclipse majors in Java source code as a primary development artifact. However other artifacts, such as PDE plug-in manifests and Ant build scripts, are also supported. Each artifact type has associated with it builders, creation wizards, syntax-aware editors, validators, semantic search extensions, and refactoring support. Eclipse users expect editors to provide first-class programmer assistance such as code completion, syntax coloring, error markers, and quick fixes. WTP extends Eclipse with support for the large set of new artifact types encountered in J2EE development. These include HTML, CSS, JavaScript, XHTML, JSP, XML, XSD, WSDL, SQL, and all the J2EE deployment descriptors.

One of the key design goals of WTP is to extend Eclipse seamlessly to support these additional execution environments and artifact types. All

of the functions that Eclipse users have come to expect from Java source code should “just work” for the new artifacts. For example, if I select a Java main program, I can Run or Debug it. The same should apply to a JSP. When I select it, the Run command should do something sensible. Specifically for a JSP I expect the Run command to somehow deploy my code into a J2EE server and launch a Web browser with the URL for my JSP. Similarly, the Debug command should run my J2EE server in debug mode and the standard Eclipse Debugger should let me step through my JSP source code. My JSP editor should provide code completion for both JSP tags and inlined Java scriptlets. Furthermore, I expect the code completion for Java scriptlets to work exactly like the code completion for Java source files. I don’t want to learn new editing commands simply because I’m editing a new artifact type.

WTP 0.7 achieves many of these goals but there is much work to do to support J2EE fully. Consider the problem of refactoring a J2EE application. An operation as simple as renaming a Java class can have many consequences. If the renaming isn’t fully rippled through the application, a runtime error can occur. For example, in addition to references from other Java classes, a Java class can be referenced by JSPs and deployment descriptors. All of these artifacts must be updated to reflect the new name. Suppose the Java class is deployed as a Web Service and that WSDL is generated from it. The WSDL may also need to be regen-

erated. First-class refactoring of J2EE applications will be an ongoing focus for WTP.

Now let’s create a JSP version of “Hello, world.” If you’d like to follow along, you’ll need to do some setup. Download and install the latest stable driver of WTP from the Web site mentioned above. WTP provides support for many popular commercial and Open Source J2EE servers but doesn’t include the runtimes. So you also need to install a server on your machine. For purposes of illustration, I’ll use Apache Tomcat 5.0.28, which you can obtain from <http://jakarta.apache.org/tomcat/>. Finally, you’ll need a full JDK since JSPs require a Java compiler. I’m using Sun J2SDK 1.4.2_06.

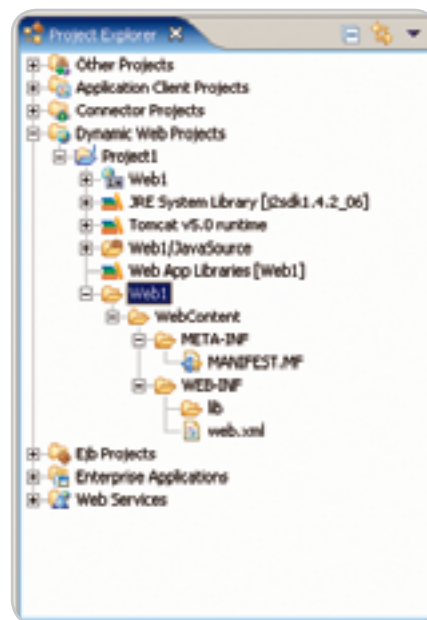


Figure 2 Project Explorer

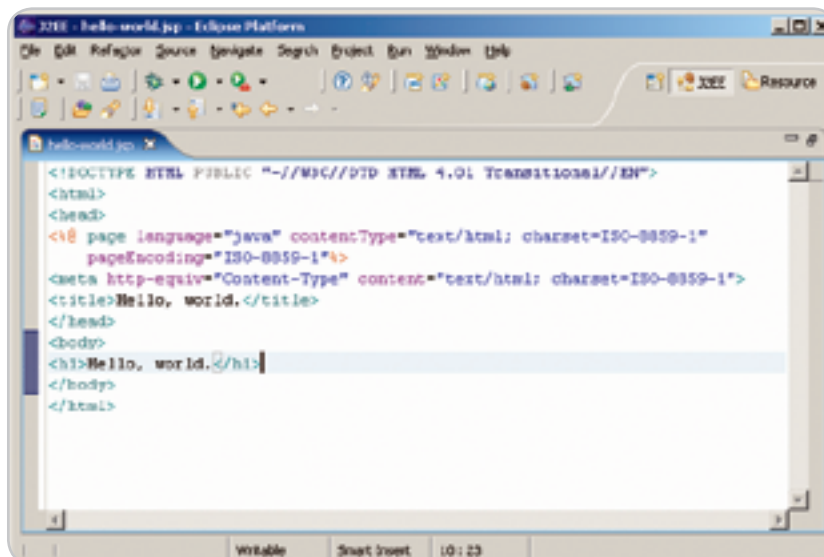


Figure 3 JSP editor with hello-world.jsp

WTP provides a Preference page for Servers. Open the Preference dialog and go to the Server page. Add your Tomcat 5.0 server and configure it to use your JDK (if you use a JRE then JSP compilation will fail). Figure 1 shows the Server Preference page.

Next, create a new Flexible Java Project named Project1 and a new J2EE Web module named Web1 in it. A Flexible Java Project is a J2EE project that can hold several J2EE modules. Figure 2 shows the J2EE Project Ex-

plorer after Project1 and Web1 have been created.

Now we're ready to create our JSP. Select the WebContent folder of the Web1 module and use the New File wizard to create a JSP named hello-world.jsp. The wizard fills in the skeleton of a JSP document and opens the file with the JSP editor. The JSP editor has full content assist for HTML and JSP tags, as well as Java scriptlets. Edit the file to say "Hello, world" and save it. Figure 3 shows the JSP editor.

Finally, we're ready to run the JSP. Select hello-world.jsp in the Project Navigator and the Run on Server command from the context menu. You'll be prompted to define the server to be used for the project since this is the first launch. Select the Tomcat server you previously defined and make it the project's default. The Web1 module will be added to the server configuration and the server will start. A Web browser will then be launched with the URL for hello-world.jsp. Figure 4 shows the Web browser with the Web page generated by hello-world.jsp.

Debugging JSPs is also simple. To demonstrate debugging, let's add a Java scriptlet to the JSP. The scriptlet will retrieve a query parameter named "user" and display a welcome message. Listing 1 shows the modified JSP that includes the Java scriptlet.

Set a breakpoint on the line that begins `String user =` by double-clicking in the left margin. Select the file hello-world.jsp in the Project Explorer and invoke the Debug on Server command from the context menu. The server will restart in debug mode and the Debug Perspective will open with execution halted at the breakpoint. Figure 5 shows the Debug perspective when the JSP is passed the query parameter `user=JDJ` readers on the URL.

You can now step through Java scriptlets and explore Java variables as usual. In Figure 5, the variable `user` has been selected in the Variables view that shows its current value `JDJ` readers. Click the Resume icon to finish processing the JSP. Figure 6 shows the Web browser displaying the resulting Web page.

WTP 0.7 Features

The preceding Quick Tour shows a small cross-section of the features available in WTP 0.7. The full set of features in WTP 0.7 includes server, Web, XML, Web Service, J2EE, and data tools. I will now briefly describe these. Consult the WTP Web site for more details.

The server tools let you define and control servers. Servers can be associated with projects, and can be started, stopped, started in debug mode, and controlled in other ways. Projects can be deployed to servers, or servers can be configured to access your Eclipse

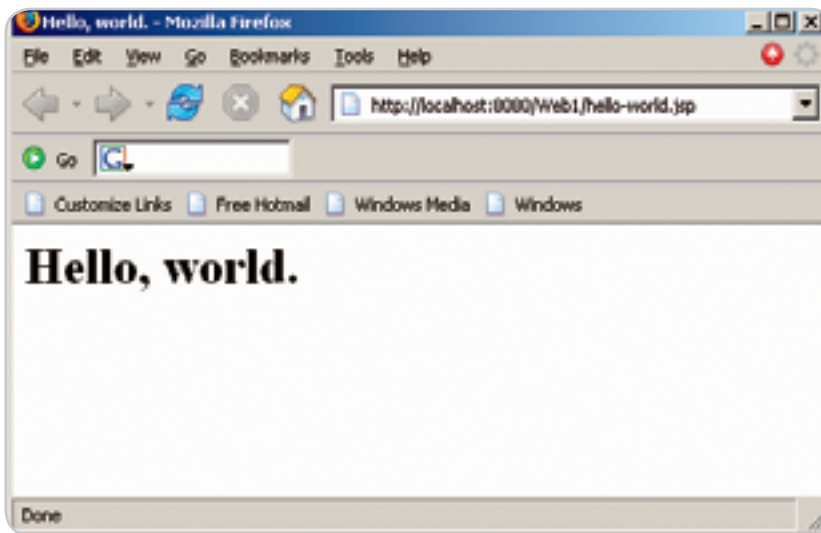


Figure 4 Web browser with hello-world.jsp

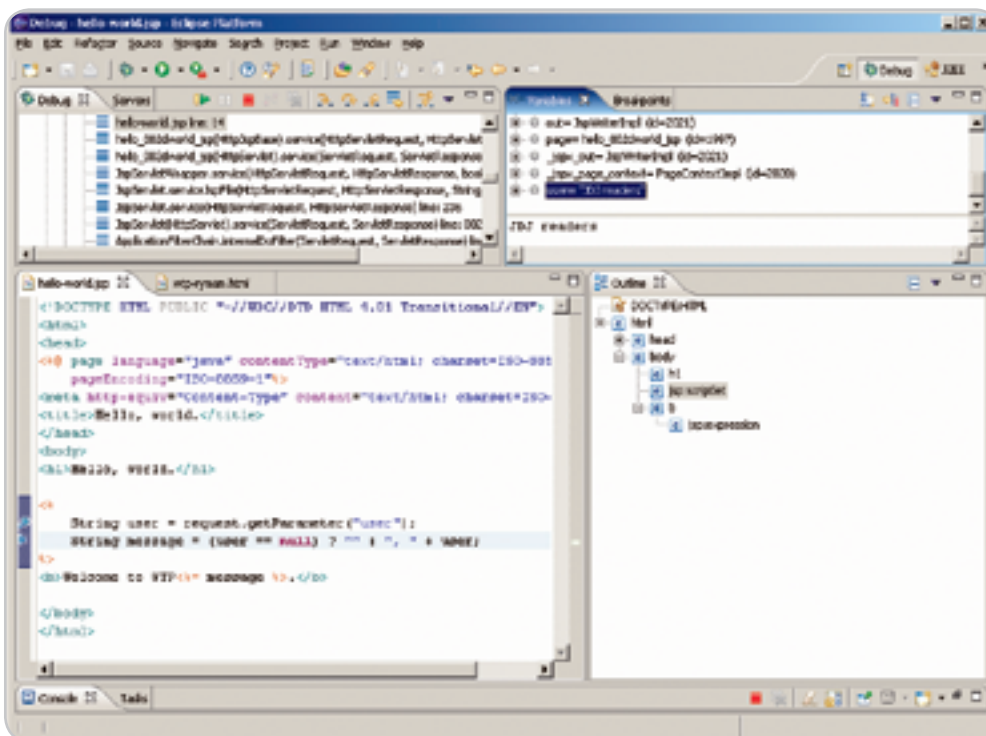


Figure 5 Debug perspective

Get the complete picture



Discover the New

ILOG JViews Graphics Components

**First comprehensive Java graphics toolkit for Eclipse
Full Java Server Face (JSF) Support**

Build better GUIs in less time. Exceed your application requirements.

Reduce your development time & risk. Improve user experience & value.

Advanced BPM modeling and monitoring displays, data charts, gantt scheduling & resource displays, network & equipment displays, network diagrams, EMS/NMS telecom displays, custom dashboards and more. Whatever your display needs – for desktop or Web client – JViews has the solution.

Learn more. Test-drive an Eval. Call: 1-800-for-ILOG or go to:

- ILOG JViews Diagrammer <http://diagrammer.ilog.com>
- ILOG JViews Charts <http://charts.ilog.com>
- ILOG JViews Gantt <http://ganttt.ilog.com>
- ILOG JViews Maps <http://maps.ilog.com>
- ILOG JViews TGO <http://jtgo.ilog.com>



Changing the rules of business™

workspace content directly. WTP includes server support for Apache Tomcat, Apache Geronimo, and JBoss, as well as many other popular commercial and Open Source J2EE application servers. The server tools include an extension point so that new server types can be easily supported. You can add a new server type by providing either a simple XML configuration file or a Java plug-in.

The Web tools let you create static Web pages based on HTML, XHTML, CSS, and JavaScript. The Web tools include source editors that are based on the WTP Structured Source Editor (SSE) Framework. WTP Web editors provide content assist, syntax highlighting, validation, and other standard Eclipse editor functions. The Web tools also include an embedded Web browser and a TCP/IP monitor that's very handy for debugging HTTP traffic.

The XML tools include source editors for XML, DTD, and XSD that are based on the SSE Framework. Besides source editing, graphical editing is also provided for XSD. The XML tools also include code generators for creating XML instance documents from DTD or XSD.

The Web Service tools include a WSDL editor, a Web Service Explorer, a Web Services Wizard, and WS-I Test Tools. The WSDL editor includes an SSE-based source editor and a graphical editor. XSD editing is seamlessly integrated with WSDL editing. The Web Service Explorer lets you search and publish to UDDI registries, and dynamically test WSDL-based Web Services. The Web Service Wizard ties together the full development

lifecycle. It lets you deploy Java classes as Web Services, generate server and client code from WSDL, and generate test clients, as well as being integrated with the Web Service Explorer for publishing and discovery. The Web Service Interoperability (WS-I) Test Tools let you validate WSDL and SOAP for compliance with the WS-I profiles.

The J2EE tools let you create J2EE projects and artifacts like JSPs, servlets, and EJBs, as well as the J2EE deployment descriptors, and deploy these to app servers. The J2EE tools have an SSE-based JSP source editor and a J2EE Project Navigator that displays J2EE components as objects. This provides a higher-level view of your project resources, for example, by displaying all the files related to an EJB as a single EJB object.

The data tools include support for connecting to JDBC-based databases such as Cloudscape, Derby, and other commercial and Open Source databases, and exploring their tables. The data tools also include an SQL source editor that lets you easily execute SQL statements and view the results.

The WTP Noosphere

In his essay "Homesteading the Noosphere," Eric Raymond likened Open Source developers to homesteaders who stake out their turf in the sphere of ideas. I will therefore describe the turf that WTP has staked out in the Eclipse noosphere.

WTP components are organized along the lines of open standards. WTP classifies the world of standards along two dimensions – technology and formality.

The technology dimension ranges from neutral standards on one extreme to J2EE standards on the other. Technology neutral standards form the foundation of the Web. In fact, the Web has succeeded because it's defined in terms of formats (such as HTML and XML) and protocols (such as HTTP) that specify how systems interact, but don't specify how systems are implemented. This neutrality has allowed vendors to choose the most appropriate implementation technology and compete on the basis of the quality of their implementations. On the other hand, J2EE standards specify application portability rules for J2EE implementations. Both kinds of standard are essential for the viability of the Web.

The formality dimensions define how the standards are created. At one extreme we have the de jure standards bodies such as ISO and IEEE and at the other we have technologies that aren't associated with any formal standards definition organization, but have become de facto standards because of their popularity. Organizations such as W3C, OASIS, and JCP, which define the standards relevant to WTP, have formal processes and are near the de jure end of the spectrum. Popular Open Source technologies such as Struts and Hibernate are at the de facto end.

Figure 7 shows the world of standards classified along the technology and formality dimensions. The turf of WTP is, in principle, all the important standards that are relevant to Web application development. However, the charter of WTP focuses on the standards that are formally defined by recognized standards-definition organizations, i.e., those that cluster towards the de jure end of the spectrum. WTP consists of two sub-projects, Web Standard Tools (WST) and J2EE Standard Tools (JST) that cover the formally defined Web and J2EE standards.

The reasoning behind this scope is that WTP aims to be a platform that many vendors can build on. The formal standards form the building blocks that most vendors want. Support for this base layer of standards is, in a sense, the "table stakes" of any tool. Vendors can cooperate on this

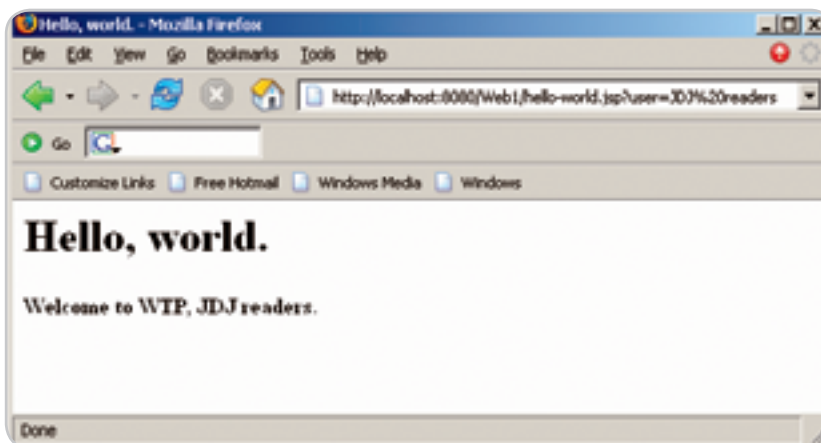


Figure 6 Web browser with modified hello-world.jsp

WebRenderer™



Standards compliant embeddable web browser component

WebRenderer is a cutting edge embeddable Java™ web content rendering component that provides Java applications and applets with a fast, standards compliant HTML and multimedia rendering engine. WebRenderer provides a feature-packed API including complete browser control, a full array of events, JavaScript interface, DOM access, document history and more.

Why WebRenderer?

- Standards Support (iHTML 4.01, CSS 1 & 2, SSL, JavaScript, XSL, XSLT etc.)
- Exceptionally Fast Rendering
- Predictable Rendering
- Scalability (deploy in Applications or Applets)
- Security (based on industry standard components)
- Stability and Robustness

Embed WebRenderer to provide your Java® application with standards compliant web content rendering support.

To download a 30 day trial of WebRenderer visit
www.webrenderer.com

JadeLiquid™
Software

Copyright JadeLiquid Software 2004. JadeLiquid and WebRenderer are trademarks of JadeLiquid Software in the United States and other countries. Sun, Sun Microsystems, the Sun Logo and Java are trademarks or registered trademarks of Sun Microsystems, Inc. in the United States and other countries. All other trademarks and product names are property of their respective owners.

base layer and produce high-quality common components while sharing the development expense. Conversely, by not including the de facto standards, WTP leaves room for vendors to innovate and differentiate. For Open Source to succeed contributors must have a way to generate a profit otherwise they won't be able to continue contributing. We hope that this design will yield an excellent set of core Open Source J2EE tools for users, and a solid platform that supports a thriving aftermarket of extenders.

The standards arena is very active and as existing standards are revised and new standards defined WTP will support them based on their market relevance. There may also be a migration of de facto standards to the de jure quadrants. WTP's charter may expand in the future to include new sub-projects. However, immediately, WTP consists of the WST and JST sub-projects.

The WTP Ecosystem

WTP has the dual goals of providing both tools for the developer community and a platform for tool vendors to extend. Satisfying the needs of vendors requires that WTP define a set of platform APIs. The significance of a platform API is that it will be preserved in future releases. This means that a plug-in that runs in WTP 0.7 will also run – without recompilation – in future versions of WTP. The stability of platform APIs is key to vendor adoption. Clearly if WTP changed its APIs from release to release, vendors would expend significant effort reacting to the changes, and this would slow the

rate at which users and vendors move to new versions of the platform.

WTP relies heavily on the user community for testing, bug reports, and enhancement requests, and the development of the user community is one of our main focuses this year. The WTP Web site has tutorials, articles, presentations, and event information. WTP will be well represented on the conference circuit this year. Look for upcoming WTP presentations at events such as EclipseWorld, JavaOne, and the Colorado Software Summit. There are also a couple of WTP books in the works. A thriving user community is a magnet for vendors. As the WTP user community grows so will the number of tools built on it.

Finally, WTP has a role to play in education. Since WTP is free Open Source and supports industry standards, it's an ideal learning tool for the coming generation of J2EE developers. I hope to see universities, community colleges, and even high schools use it for teaching.

The WTP contributor community is drawn from both vendors and users. There are many ways to contribute. You can start by downloading WTP, kicking the tires, and telling your friends about it. If you find a problem or have an idea, open a Bugzilla report. Monitor the newsgroup, and share your solutions to problems with others. If you can write, submit a tutorial or contribute to the online Help system. If you have fixed a problem, submit a patch. If you have time to work on WTP, check Bugzilla for open problems or look at the WTP Help Wanted page. And after

you have established a track record of valuable contributions, you can be voted in as a committer.

What's Next?

WTP 0.7 is scheduled for release in July 2005. We are planning to follow that with WTP 1.0 later in the year. The focus of WTP 1.0 will be on the further development of platform APIs to enable the first wave of products based on WTP. Following that, WTP 1.5 will be released with Eclipse 3.2 in 2006. Candidate items for WTP 1.5 include support for revisions of major specifications such as J2EE 5.0, SOAP 1.2, and WSDL 2.0, as well as new JSRs and Web Service specifications.

We also expect the shape of WTP to change as new projects emerge and mature at Eclipse. New vendors are joining Eclipse and projects are being created at a rapid clip. For example, the data tools in WTP will move into a new Data Tools Project. Technology projects such as those proposed for EJB 3.0 and JSF will likely move into WTP as they mature.

A Final Word

Like all Open Source projects, the success of WTP depends on the contributions of an enthusiastic community. The project is still in its formative stage and there's much work to do. The project needs users, testers, writers, developers, speakers, trainers, mentors, evangelists, extenders, distributors, and leaders. If you are interested in J2EE development, then please consider this article as your formal invitation to join the WTP community. ☺

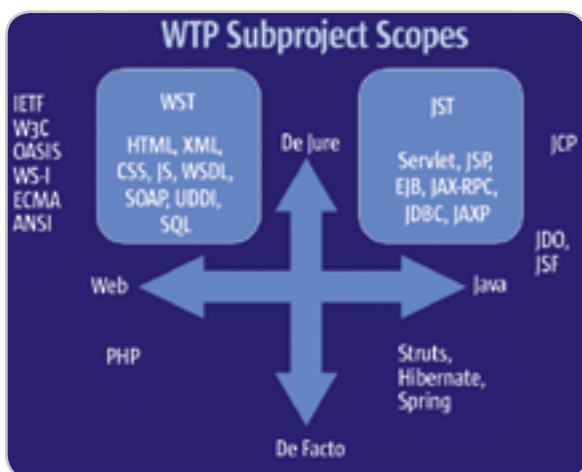


Figure 7 The scope of WTP

Listing 1: Hello-world.jsp with Java scriptlet

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN">
<html>
<head>
<%@ page language="java" contentType="text/html; charset=ISO-8859-1"
    pageEncoding="ISO-8859-1"%>
<meta http-equiv="Content-Type" content="text/html; charset=ISO-8859-1">
<title>Hello, world.</title>
</head>
<body>
<h1>Hello, world.</h1>

<%
String user = request.getParameter("user");
String message = (user == null) ? "" : " " + user;
%>
<b>Welcome to WTP<%= message %>.</b>

</body>
</html>
```


Style Report 7

Light Weight, Integration Ready Enterprise Reporting & Analysis



open standards open architecture Linux Windows Java J2EE Tomcat SOAP LDAP DHTML

Traditional BI Challenges:

- ▶▶ Monolithic, resource intensive
- ▶▶ Proprietary software stack
- ▶▶ Requires ETL/Data warehouse

Style Report Solutions:

- ▶▶ Light weight, integration ready
- ▶▶ Open software stack, J2EE drop-in
- ▶▶ Real time transactional DB and OLAP



For more information and to download a free evaluation copy www.inetsoft.com/jdj

Challenges in the J2EE Web Tier

by Matt Raible

How open source frameworks drive innovation

Over the course of its life, the J2EE Web Tier has faced many challenges in easing Web application development. While it's a scalable, enterprise-ready platform, it isn't exactly developer-friendly. Particular challenges to Web developers include the need for a standard Web framework, compatible expression languages, and availability of components. Several Web frameworks have been developed to resolve these issues, each with its own strengths and weaknesses. This article discusses the unique challenges of the J2EE Web Tier and how various technologies have attempted to resolve them. By learning from and competing with each other, these Web technologies play an important role in pushing the limits of excellence to produce ever-higher standards of Web application development.

Problem: Too Many Frameworks for J2EE

A plethora of frameworks is available for building Java-based Web applications. Most of the Web frameworks in Java result from the difficulty in using servlets and JSPs, which are part of the J2EE standard. While servlets and JSPs serve as the underlying APIs for most frameworks, they don't have any built-in features to ease development. Using plain servlets and JSPs is cumbersome, and you end up writing a lot of *plumbing code*, when you should be writing *application code*. By using a Web framework, you can concentrate on coding your application, rather than the architecture that makes it work.

Over 50 Java Web frameworks exist to make the developer's life easier. It begs the question, "Why not have a standard Web framework as part of J2EE?"

Solution: A Standard Framework Called **JavaServer Faces**

At the JavaOne Conference in 2002, Sun Microsystems announced the de-

velopment of a standard Web framework to include as part of the J2EE bundle. Its name was *JavaServer Faces* (JSF) and it was designed to put a pretty *face* on developing Web applications in a J2EE environment.

JSF was developed to be *tool-friendly*, so IDE vendors could create WYSIWYG environments where developers could drag-and-drop their applications into a usable system. Microsoft has always had good tools for its technologies, and Sun wanted to fuel innovation in the Java IDE space to produce tools similar to Visual Studio .NET.

JSF's architecture is similar to ASP.NET, which is more page-centric than controller-centric. Controller-centric frameworks make requests go through a front-controller servlets that dispatches requests to smaller "worker" servlets. In Spring MVC, these workers are *controllers*, while in Struts and Web-Work they're *actions*. A worker servlet will typically put things in the request for the view to render. On the other hand, JSF is page-centric. This means that users will typically navigate to a template page in the application rather than go through a worker servlet. JSF pages are backed by classes that contain data and actions that the UI can invoke. These *actions* are also known as *listeners* and contain logic that ties the class to a backend system.

Early Challenges of JSF

The development of JSF went through some challenging times. Sun announced it in June 2002, but didn't release its initial 1.0 version until March 2004. During that time, many other Java Web frameworks cropped up and JSF got some bad press. The spec changed significantly between iterations, and it often lacked backward compatibility. Authors and publishers produced books that were out-of-date before they

were released. Developers who tried to work with JSF often found that the functionality didn't work according to the documentation. Furthermore, since Sun architected JSF with tools vendors in mind, some developers (who were used to coding their JSPs by hand) became frustrated with JSF's verbosity.

While JSF was being developed, so were JSP 2.0 and its Expression Language (EL). JSP's EL made retrieving values and resolving variables easier. Rather than using `<jsp:useBean>` and `<jsp:getProperty>`, a developer could simply use `#{bean.property}` to retrieve a property from a JavaBean. Developers could also use the new EL with the Java Standard Tag Library to implement iterations, calculations, internationalization (i18n), and number/date formatting.

Ideally, JSF would have used the JSP 2.0 EL syntax to resolve variables and expressions. However, JSF's Expression Language didn't have some of the concepts that existed in JSP's EL. First, it didn't have a page scope like the EL did. Second, JSF Expressions were often *formulas* (the left side of the equation) rather than *solutions* (the right side). For example, in JSP EL, the following expression means "call `getAction()` on the `userForm` bean in any scope."

```
#{userForm.action}
```

While the same expression in JSF's EL can mean the same thing, it can also be an invocation of a method such as "call the action method of the `userForm` bean."

```
#{userForm.action}
```

To complicate things further, developers were demanding a workable version of JSF. The JSF Expert Group didn't want to wait for JSP 2.0 and J2EE 1.4

Matt Raible is a J2EE consultant and developer living in Denver, Colorado, and the Spring and Web Frameworks Practice Leader for Virtuas. He has been developing Java Web applications since 1999 and is president of Raible Designs, Inc. He is also a member of the J2EE 5.0 Expert Group and hopes to help them make J2EE easier for developers. Matt is the author of *Spring Live* (SourceBeat Publishing).

mattr@sourcebeat.com



Powering Eclipse to the next level...

NitroX™

PROFESSIONAL TOOLS FOR ECLIPSE™

JSP | Struts | JSF

Professional development for JSP, Struts & JSF

Simultaneous visual/source Struts & JSF configuration

Code completion for all levels

Automatic validation & error checking for all levels

Debugging support for all levels

AppXRay, AppXaminer, AppXnavigator

download at: www.m7.com/power



Top Reviews Winner: InfoWorld (8.3), CRN ★★★★★, SDMagazine ★★★★★

to be finished, so they created their own expression language, hereafter referred to as *JSF EL*.

Problem: JSF EL vs JSP EL

For the most part, the JSF EL hasn't affected other Web frameworks because it's JSF-specific; however, the lack of compatibility between the two expression languages (JSTL and JSF) has been a point of developer contention and frustration. JSTL has been one of the most widely accepted and praised additions to the Servlet API, but it didn't work with JSF as many expected.

The JSP EL has also produced some problems for framework developers who support JSP as a view choice. Mainly, some framework developers can no longer use the `{...}` syntax as a placeholder to indicate variables. Since these placeholders are reserved for JSP 2.0, if the framework resolves variables outside of the standard *scopes* (page, request, session, application), variable resolution will simply fail. Frameworks like WebWork and Tapestry use **OGNL** as their expression language, and they resolve variables according to a **ValueStack** and component hierarchy, respectively.

Note: OGNL stands for Object-Graph Navigation Language. It is an expression language for getting and setting the properties of Java objects. You use the same expression for both getting and setting the value of a property. OGNL is more powerful than the JSP/JSTL Expression Language. Not only can it get and set values, it can invoke methods. Furthermore, OGNL expressions can contain almost any Java code.

Solution: A Unified Expression Language

To solve the disconnect between JSF EL and JSP EL, the JSP 2.1 and JSF 1.2 specifications have created a Unified Expression Language. If you're using a JSF 1.2 implementation, you can use JSP expressions in your JSF applications. It also adds a *variable resolver* to JSP. This means that frameworks like WebWork can control what `{...}` means and tell it to talk to its ValueStack instead of the standard scopes. This is important for many framework developers because they don't want to invent their own syntax for resolving expressions. The new JSP and JSF versions are part of J2EE 5.0 and will be required by any J2EE 5.0-compliant containers.

Problem: JSP Unfriendly to Component-Based Frameworks

JSP is the primary view choice for JSF apps, but it's clunky at best. Most frameworks that use JSP simply render values as they encounter them when loading a page. Relationships between portions of a page will only occur at the HTML level, rather than on the server-side.

JSF has a different model – it builds a component tree when a page loads. JSP adds page components to the tree in the order they appear on the page. This makes it difficult to create relationships between components, such as labels and input fields. To work around this, you can wrap your forms with an `<h:panelGrid>` tag, but then you have to remove any HTML from your form (or wrap it with an `<f:verbatim>` tag). You end up with a JSP page that doesn't have a single line of HTML in it. In other words, you're back to the pre-JSP days with JSE, where Java code, rather than HTML authors produce the entire HTML.

Solution: HTML Templates

The good news is that JSP isn't the required view technology for JSF. It was simply chosen because of the proliferation of JSP-based apps and because so many developers were familiar with it. JSF 1.2 lets JSF use HTML Templates like Tapestry does. These templates will either reuse existing HTML attributes (such as *id*), or add an additional one (such as *juwid*). Hans Bergsten provides an example of this in his "*Improving JSF by Dumping JSP*" article. An Open Source project called Facelets (<http://facelets.dev.java.net>) also implements such a solution.

JSF doesn't just render HTML; that's just one of the possible *render kits*. In theory, it's possible to create render kits for all kinds of view options: XUL, J2ME, or even Laszlo.

The Future of J2EE's Web Tier

The future of J2EE's Web Tier depends largely on JSF – at least from a *standards* perspective. However, many popular Open Source alternative Web frameworks are available, such as Spring MVC, Struts, Tapestry, and WebWork. These frameworks have already helped shape the future of J2EE's Web Tier. JSF incorporates many of the ideas and features of Struts. The *managed properties* of a JSF Managed Bean are very much like Spring's Dependency Injection. A JSF Managed Bean looks very

similar to a WebWork Action. JSF mimics many of the concepts in Tapestry to enhance its ease of use, such as a rich set of components and HTML templating. Furthermore, all of these frameworks are part of J2EE in a sense, since they build on top of J2EE's Servlet API. Without J2EE, these frameworks probably wouldn't exist in their current form. Likewise, J2EE wouldn't be where it is today without these frameworks and the innovation and competition they've provided.

Having choice among frameworks is a good thing because it forces framework developers to innovate and compete for users. Not only do the different frameworks show different ways of doing things, but they're also borrowing from and competing with one another to become better. Competition breeds innovation. J2EE stands to benefit from this innovation because the good ideas can be added to JSF and the bad ones can be removed. The Java Community Process and Expert Groups for J2EE and JSF are very open to the community, so the community will contribute and help improve it, much like they have with the many Open Source alternatives.

All of the frameworks mentioned do the same thing, just in different ways. JSF and Tapestry are making developing Java Web applications easier due to smarter defaults and better plumbing. They don't require the developer to know much about the Servlet API, and they handle most input types transparently – without using custom converters. Furthermore, their concept of *components* lets developers create reusable components that they can use with only a few lines of code.

The future of J2EE's Web Tier isn't one of self-determination. Many other factors will play into the success of a *standardized* Web framework. J2EE and its developers can ensure its continued success on the Web by being willing to adapt. As a community, we need to continue to recognize good technologies and ideas from experienced developers and Open Source projects. We need to be willing to support remote scripting with Ajax, and we need to adapt to produce rich-client experiences, like those that Laszlo offers with the Open Source Flash-rendering system. By accepting and enhancing these technologies, we can continue to use J2EE and its APIs to make our lives easier. By developing more efficiently with the best technologies available in Java, we can continue to produce satisfied customers. ☺

Javavavoom!

Get a high-performance, transactional storage engine that's 100% Java.

Berkeley DB Java Edition 2.0

Download at www.sleepycat.com/bdbje

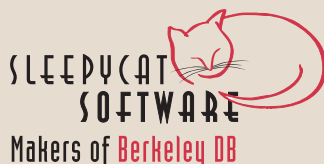
Now there's a high-performance storage engine that loves Java just as much as you do: Berkeley DB Java Edition (JE). Brought to you by the makers of the ubiquitous Berkeley DB, Berkeley DB JE has been written entirely in Java from the ground up and is tailor-made for today's demanding enterprise and service provider applications.



Berkeley DB JE has a unique architecture that's built for speed. The software executes in the JVM of your application, with no runtime data translation or mapping required. And because it supports J2EE standards such as JCA, JMX and JTA, you can be sure you have the widest range of options.

Experience the outstanding performance of Berkeley DB JE for yourself.

Download Berkeley DB Java Edition today at www.sleepycat.com/bdbje, and view the presentation "*Design and Implementation of a Transaction Data Manager*."





Back to Two Tiers and Plain JSP

The issue of picking a language

by Brian Russell

There comes a time, for many Web sites, when the transition from static HTML to dynamic HTML has to be made. Whether it's a static company Web site that needs to become a dynamic online store, or a simple collection of family pictures that's become too large to manage with HTML alone, a decision has to be made to move to an environment that makes it easier to build and maintain the site. Deciding to use server-side programming to create your site on-the-fly can become the only option, but what language you decide to use can be a difficult and important decision.

Server-Side Options

Server-side programming options are plentiful. Each offers its own pros and cons, and each promises to be the best choice. Examples of languages used for dynamic Web site development include Perl, PHP, ASP (and .NET), ColdFusion, and Java. Factors, such as cost, can quickly drive you to one language. The cost could be in software licensing, server usage, or server management. Some of the options, like Perl and PHP, are distributed freely, which can have quite an influence over someone looking to step into the dynamic world. These two are also commonly found on most Web servers, further increasing their appeal. Java, which is also freely available, is also found on many servers. If you're just starting to get into dynamic Web site development, it's unlikely you'd have your own server and would rent or lease it from a hosting service provider instead. A quick Google search will show you that the number of hosting companies is amazingly high. What you'll also find, after checking out some of those companies, is that Web servers running Perl/PHP usually cost less than Java ones.

Java on the Server

The use of Java as a server-side programming language has increased during the past few years. The way Java is used has also changed dramatically. Applications are commonly written to use the most powerful and robust architectures available. From EJB-driven database access to XML based extraction, everything new and exciting in Java requires more configuration and coding.

User interface architectures are also everywhere, allowing easy integration of the complex backend processing in Web page development. To an outsider, or someone new to Web development, Web applications written in Java can appear to be bloated, unmanageable, and a headache to be avoided. Most of what you read about Java is based on new technologies. Someone entering this world can be easily overwhelmed with all of the APIs, configuration options, and complexity that the Java community seems to embrace. This leads the new programmer away from Java and into the other options available like PHP.

Java Server Pages

Java Server Pages (JSP), like other scripting languages, contains a combination of HTML and code to be interpreted and executed by the server. In the case of JSP, the code is written in the Java programming language. JSP can also contain special tags that appear to be HTML but are really references to additional Java code to be executed by the server. These are known as tag libraries. Before JSP existed, Java servlets were used to write out HTML to the Web browser. This made it difficult to make simple changes to a page because the servlets contain many print statements to output the HTML code. Making changes to the servlets also required that they be compiled before getting deployed to the application server. JSP simplified this process by letting the developer work with a simple text document. The application server takes this document and generates and compiles a servlet. This is an important difference between Java and the other Web languages. Java code is compiled, while PHP, for example, is not.

Does this mean Java is faster than the other languages? The simple answer is yes, but the reality is that there are many factors that play to what makes one system faster than another. The way the code is written can have a major impact on the performance of the application. Other factors, like server performance, database performance, and other external bottlenecks are also influential.

Editors and IDEs

When working with JSP you can use any kind of editor you like. Most Web developers wouldn't think twice about using a simple text editor to edit HTML or other languages like PHP or Perl, but when people think of Java they often think of an integrated development environment (IDE) like Eclipse, Sun's JavaStudio Creator, IBM's Rational Application Developer for WebSphere or BEA's WebLogic Workshop. If you're going to be developing a full-blown

Brian Russell is a software engineer for Priority Technologies, Inc. in Omaha, Nebraska.

brian.russell@prioritytech.com

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



Think JReport.

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

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

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

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

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

When it comes to embedded Java reporting, JReport is the perfect fit. Download a FREE copy of JReport today at www.jinfony.com/jp6.



J2EE Web application, then an IDE would be the way to go. If you are just doing JSP development, then you can choose something simpler. It's often easier to use an IDE designed for the language you're working with because the IDE can add important development tools, such as code assistance or syntax checking. Although these features are desirable, they're not necessary. In the end, JSP developed in an IDE works the same as the code written in VI. Relying less on the tool can also help you be a better programmer because you'll learn from your mistakes instead of just having them corrected for you.

Web Architectures

The architecture of a system can determine how robust it will be. The architecture also determines how complicated it will be. For many people, Java application development brings to mind a three-tier architecture that separates the display logic, business logic, and data retrieval. The Model-View-Controller (MVC) architecture is a common example. A typical MVC design would include JSP for providing the user interface (view) to the client, and servlets (controllers) working between the JSP and the persistence layer (EJB, Hibernate, etc.) for data access (model) and application flow.

An MVC architecture is an ideal paradigm for larger enterprise applications. It lets different types of developers work on a system's components independently. It also provides much better scalability and maintainability, but it does introduce more overhead that can slow performance when compared to a simple two-tier architecture.

The two-tier architecture is, very simply, the code and the database. The code is a combination of display logic and business logic. Some people may argue that the Web browser is a third tier. The important point here is that the presentation and business layers be combined into one.

If you're just going to be working on a site for yourself for personal use, or even business use, the multi-tier architecture is more work than needed to handle the site. A two-tier architecture is commonly used in other languages and it can be easily implemented in JSP development. When evaluating the Web programming language options, Java is often thought of for large-scale sites while other languages like PHP are thought to be for smaller sites. JSP development can be done as quickly and easily as the others, often with better results.

Keeping It Simple

There's a time and place for every type of architecture. If that weren't true, we wouldn't have all of these architecture options available to us today.

The two-tier architecture is an excellent choice when putting together a family Web site, a personal hobby site, or even a small business site. In situations where one person is solely responsible for a site, it can be easier to build a site with just JSP. Even if there are a couple of people who will work on the site, it can still be done if they all have similar capabilities. With this architecture, the JSP contains all of the display logic, business logic, and database retrieval code. Anyone working on the site would need to understand all aspects of Web development.

Suggesting this type of development can be a source of argument. Many software engineers strive to develop applications using just the multi-tier paradigm. They argue that it offers easier development because components are separated, which also promotes component ownership and specialization in the developers. Componentized design also promotes reuse. It can also be argued that multi-tier architecture provides better scalability and maintainability. These are valid points, but there are opposing arguments to be made.

A two-tier system is easier to design and allows for faster develop-

ment. When developing JSP, it's also possible to use reusable components. JSP can easily include other JSP at compile time or runtime. This lets you create standard JSP to be reused. When the code is imbedded in the JSP, it can still be easily maintained; it's just a different type of process. When the code is split into components, there's a well-defined level of abstraction. Each component does a specific job, such as database access or user-interface output, and the components fit together to build the application.

When a JSP contains code that handles all aspects of an application, understanding each component's function in the page becomes more cerebral than code-based. Code for database queries is mixed with HTML. Code for application flow is mixed with data display logic. Although it sounds like a nightmare to experienced Java developers working in the MVC world, it's an example of how many sites are developed using JSP and languages like PHP, Perl, or ASP. Sites written this way can be easily put together because there's less structure to implement. The addition of a multi-tier structure sacrifices time spent on additional coding for easier long-term maintenance by multiple developers.

Development

Web development using any of the available languages leads to similar results. In the end, you have a combination of code and HTML. With Perl, you'll see more code writing out HTML, whereas with the other scripting languages you'll see HTML with embedded code. JSP, PHP, and ASP will look very similar, the main difference being the code language used. All offer similar features such as easy interfaces to parameter retrieval or cookie management. Flow control elements, such as FOR loops, also have similar implementations. These similarities make the transition to JSP development easier for experienced programmers who've been using a different language.

JSP permits full use of the Java programming language. Whatever can be done in a Java class, such as a servlet, can be done in a JSP. This gives the programmer great flexibility. It also enables experienced programmers to integrate object-oriented elements easily into their JSP development. Helper classes can be easily implemented to make common and repetitive tasks easier, such as database access. Although these helper classes can be used, you don't have to. Developing dynamic sites with JSP can be done with simple JSP structures that handle all the design elements. A JSP can handle all of the database access, for example. It can retrieve a database connection from a connection pool or create its own, make database queries, display output, and finally close the connection. And if you can do it in another language, you can do it in Java.

Conclusion

When looking at Java and other languages and their evolution over the past few years, it's easy to see how complex it's gotten. Layer upon layer of complexity has been added, and Java is the leader in this field of change. This complexity, and the time and effort needed to grasp it properly, can lead new developers away from Java and into alternatives like PHP, ASP, ColdFusion, and the other Web scripting languages. It can also lead developers down a path to unnecessary architecture bloating. Small or mid-sized projects can be developed in JSP without complicated architectures like Model-View-Controller. MVC and similar architectures have a time and place, but it's important to understand that Java development doesn't have to be complicated and can be done as easily as in PHP. Unfortunately, Java development carries the stigma of complicated development tools and complicated object-oriented code. These misconceptions and misunderstanding of Java development drives people away. The flexibility of Java and the environment in which it's developed should be bringing them to it instead. ☛

Think Java development is complex & overengineered?

~~Think again.~~

~~I want to be known for consistently delivering applications and providing business value.~~

~~That is why I use ThinkCAP~~



ThinkCAP™

ThinkCAP 6.0 simplifies and accelerates the development and maintenance of J2EE-based web applications by 50%.

ThinkCAP's Visual Workbench and integrated MVC framework bring high productivity to corporate application developers (those with PowerBuilder, RPG, or Oracle-like skills) while letting J2EE programmers build reusable business logic & components using their tools of choice. The result is productivity for the whole development team.

Because of features such as Smart Data Binding, Visual Page Flow, and seamless Hibernate/Castor support, your team focuses on value added functionality — not low-level "plumbing." ThinkCAP integrates over 20 open source projects providing a practical mix of high productivity, robust functionality, and standards support.

Download a free trial version or view a demo at www.clearnova.com.

CLEARNOVA
Deliver Web Applications at the Speed of Business

Highly Productive Visual Workbench

Actions-based MVC Framework

Page Flow Designer & Generator

Easy to Learn Event Model

Advanced Data Aware Components

Forms, DataViews, Queries, Navigations

Workflows, Graphs, Treeviews, Grids, Tabs

Multirow Updatable DataViews

Extensible Browser & Server-side Validation

Point & Click Smart Data Binding™

Service Flow Designer for Web Services/SOA

Content Management & Workflow Engine

Platform Independence: Any Java Server/OS/IDE/Database

Integrated Role-Based Security

Seamless Integration with over 20 Open Source Libraries

Team Development with CVS Support



Calvin Austin

Core and Internals Editor

Mastering Mustang

This month I wanted to review some of the latest happenings in the JDK 6.0 (Mustang) release.

Although we have to wait to see the initial JSR draft, you can download the binaries the engineers are actually working on. Some of the newest features to be added include JSR 223 scripting support and JAX-WS 2.0.

JAX-WS 2.0, if you weren't already aware, is what would have been JAX-RPC 2.0. Apparently the name JAX-RPC was just too confusing for developers to work out what it did according to one source. While I don't share that belief and I'm sure not everyone at Sun does, the combination of JAXB and JAX-RPC together is worthy of a rename especially since there could be potential incompatibilities with earlier JAX-RPC releases on the client side.

Getting the JDK 6.0 Download

Getting the prerelease of JDK 6.0 is actually nicer than getting the final release. As the binaries are hosted on <http://java.net>, there are no annoying additional click-through licenses. However, producing weekly drops is still a relatively new exercise; although there are references to AMD64, those downloads should work fine on the Intel x86-64 equivalent. As for the reference to the "Solaris SPARC AMD64 self-extracting file" on build 40, I trust that is a typo and not an impending new hybrid chip. The downloads reuse the Java Pack API, which reduces the download size and adds a small step to the installation.

First impressions with the Linux release are good. Startup performance seems to be in the same ballpark and the demos I tried worked. Although it's getting harder to find Web sites that use Java applets, there are some still to be found on <http://java.com>.

What Is in the Release So Far?

The introduction of JAX-WS brings a few more scripts to the platform. xjc is

the binding compiler wrapper to map XML to Java types. There is also a new command `wsimport` that when running gives clues that it used to be the better-known `wscompile`. `wscompile` is used to generate the Java helper files from a WSDL or service definition.

I mentioned the addition of JSR 223 earlier. At the moment the changes are at the API level only; there are no examples or shell interface. The scripting API is exposed as `javax.script` and a Rhino script implementation (Rhino is an open source JavaScript written in Java) as `com.sun.script.javascript` and `sun.org.mozilla`.

The quickest way to get started with what the technology has to offer is to head over to www.mozilla.org/rhino/doc.html. The example `RunScript` launcher script can be easily modified to import the `sun.org.mozilla` classes instead of `org.mozilla`. In addition, you need to catch a sun JavaScript exception. The `RunScript` launcher allows you to supply JavaScript as a command-line option to the JVM. Once the API has been made official, the normal method would be to use the new `javax.script` engine hooks.

An example of using `RunScript` with JDK 6.0:

```
java RunScript 'java.lang.System.
getProperty("java.version")'
1.6.0-ea
```

Another neat feature is an incremental improvement to the Java IO API. I mentioned in an earlier article that the NIO JSR changes have slipped the release. However, you can now see how much disk space is available without having to resort to a JNI call.

The following code snippet reports 1K blocks of space on directory / export.

```
File f = new File("/export");
System.out.println(f.getFreeSpace()/1024);
System.out.println(f.getTotalSpace()/1024);
```

Client Java Changes

Although many of the server changes are bounded by API approval, the majority of the client changes are implementation-dependent only. So as long as the API remains the same, the engineering team can continue with development. This means that a good portion of the client-side functionality has been done.

One very simple feature I was able to test out was the splash screen API and splash screen launcher. You can try it out too. The following example was run from the `demo/jfc/Stylepad` directory:

```
java -splash:src/resources/rabbit.gif -jar
Stylepad.jar
```

You'll then see the rabbit image appear as a splash screen. Many of the other client features are purely API driven, including the desktop integration. This is probably sufficient for ISV/tools developers; however, it may take a while to see serious adoption by the majority of developers.

Vote Away

One request from the JDK 6.0 team I wholeheartedly agree with is logging or voting on bugs you discover. I found a couple when writing this editorial, both of which were already identified by the QA teams. I was able to vote on them though.

The Sun JDK product has a long integration cycle, meaning that getting approval to integrate bugs even up to a year before the release gets increasingly difficult. Unless a bug is a true showstopper, don't expect a fix if the release date is within three months (a cycle of testing takes over a month).

In conclusion, the JDK 6.0 release can certainly be taken out for a test download today. Hopefully the line on startup time and quality can be maintained until we see the final release. ☺

A section editor of JDJ since June 2004, Calvin Austin is an engineer at SpikeSource.com. He previously led the J2SE 5.0 release at Sun Microsystems and also led Sun's Java on Linux port. calvin.austin@sys-con.com

Smart Development Environment

15th Annual Jolt Product Excellence Award in the Design Tools category

Winner

SDE won the 15th Software Development Magazine Jolt Product Excellence Award 2005 in Design Tools category over IBM Rational Software Architect and Borland Together Designer.



VP Suite is available for a free 30-day trial at www.visual-paradigm.com

Visual Paradigm products



2004 Quadruple award winning product



Visual Paradigm helps
ACCELERATE the entire
MODEL-CODE-DEPLOY
process in a **DISCIPLINED** and
COLLABORATIVE way to **EXCEED**
customers' expectations.

sales@visual-paradigm.com
www.visual-paradigm.com



Visual Paradigm

'Being a Better Platform'

Interview by Jeremy Geelan

Interview with Mike Milinkovich, Executive Director, Eclipse Foundation

Mike, thanks for agreeing to talk to **JDJ** and bringing us all up to date with Eclipse since our last interview with you. The best way is probably just to fire off questions and allow you to answer without getting in your way!

JDJ: Overall, how's the independence from IBM going? Since most of Eclipse's committers were IBM employees when you first went independent, how is building a community around Eclipse that is not so IBM focused going?

MM: It's going really well. Frankly, it's going much better and much faster than I had originally anticipated when I started the job. In many ways we have accomplished in the past year what I had expected would take two years or more.

First, the importance of adding companies such as BEA, Borland, and Computer Associates to our board cannot be overstated. Each of these companies competes fiercely with IBM in the marketplace. Each is making million dollar plus investments in Eclipse (\$250,000 per year in dues, plus a minimum of eight developers). Each did their own analysis as to whether the Eclipse Foundation was truly independent. And each joined.

Second, the number of projects led by non-IBMers has increased dramatically over the past year. In terms of top-level projects, we have gone from three projects in which two were led by IBM to a total of eight projects with two still being led by IBM.

Third, the number of committers working on Eclipse projects who are IBM employees has steadily dropped over the past year from roughly 75 percent to just over 50 percent. The number will soon drop below 50 percent. This decrease has been mostly the result of increasing the total number of committers. We certainly do not turn away good people from IBM!

In fact, I would also like to recognize the investment that IBM has made in Eclipse. They started this adventure and their continuing investment remains impressive. I also think that the Java community as a whole should recognize the wisdom and strategic thinking shown by IBM in working to establish the Eclipse Foundation as a separate entity. IBM has truly demonstrated how to create a community.

Eclipse is truly really completely fully and utterly independent. Anyone who says otherwise has an agenda.

JDJ: How has the assimilation of so many new strategic developers after EclipseCon gone? Is the "plumbing" in place to handle this influx? Are you making adjustments to better handle large groups of people coming on board?

MM: I don't want to sugar coat things. We are having our growing pains as we start up all of these new projects. We need to help get these projects off to a good start and our processes

and people are stretched in doing so. However, it's not just the new strategic developers who are causing growth. We have been receiving project proposals from many different directions.

The reaction from the Eclipse community has been outstanding. People with experience within the Eclipse community are stepping up to help us refine our processes and work with the new project leaders and committers to help them get started.

Back in May we had our latest round of Eclipse Council meetings and it was by far the best set of meetings we've had. We were actually debating hard topics like what does it mean for a project to achieve "Eclipse quality," how can we work toward being an even more stable and predictable open source community to further encourage commercial adoption, and what are the tangible things that existing projects can do to help new projects get started. I think it's a very good sign when our community leaders are constructively discussing the tough issues.

JDJ: How much of the themes and priorities specified on your Web site – www.eclipse.org/org/councils/themes.html – were you able to accomplish in the 3.1 release?

MM: All of them. None of them.

The themes are not intended to be deliverables. They are intended to provide high-level focus to many different projects at Eclipse to



Jeremy Geelan is

group publisher of SYS-CON Media and is responsible for the development of new titles and technology portals for the firm. He regularly represents SYS-CON at conferences and trade shows, speaking to technology audiences both in North America and overseas.

jeremy@sys-con.com

“Certainly at Eclipse we will never stop working on ‘Being a Better Platform.’ We take the construction, stability, and evolution of the Eclipse APIs very seriously”

“If you are a developer interested in desktop Java development and you have not yet looked at the Eclipse Rich Client Platform, you really need to”

concentrate on similar topic areas and help our entire community move forward. I predict that we will be working on some of those themes for years to come. Certainly at Eclipse we will never step working on “Being a Better Platform.” We take the construction, stability, and evolution of the Eclipse APIs very seriously. We want developers to feel comfortable relying on our interfaces.

There are a couple of things about the Eclipse community that are very unique in the open source world that bear repeating.

First, the community is committed to quality and stability. The committers on our projects work very hard to maintain API stability from release to release. Anyone who has ever done this knows that achieving such a goal is a non-trivial effort. But the community honestly believes that API stability is one of the key elements that has made Eclipse so successful.

The second is that we are very focused on predictability. We want our technologies to be adopted commercially. If you want companies to bet their own product plans on an open source project, you need to deliver on schedule. If you look at the Eclipse Platform project, the 3.1 release will be the fourth release in a row where they have hit their dates. If anyone wonders if you can deliver open source projects with several million lines of code on schedule using developers from multiple companies at multiple sites, Eclipse is the project that erases any doubts that it can be done. It is a real testament to the amazing people involved in the project. Pound for pound, I think that Eclipse has some of the best software engineers on the planet.

JDJ: Will the WTP be included with Eclipse by default when it's done?

MM: I don't know. We are still discussing cross-project packaging options within the community.

JDJ: Tell JDJ's readers about the RCP initiative. How has the uptake of the RCP been going? Are you seeing a lot of interest?

MM: If you are a developer interested in desktop Java development and you have not yet looked at the Eclipse Rich Client Platform, you really need to. Eclipse RCP offers a great platform for developing managed applications with a rich user experience. It's doing more to generate interest in Java on the desktop than any other initiative.

We have seen enormous uptake of RCP. I wish we could talk about all of the projects we've seen. One of the most intriguing is the NASA Space Mission Control used for

mission planning for the Mars Rover. The most ambitious is the IBM Workplace Client initiative. Macromedia's recent announcement states that they will be building their next-generation Rich Internet Application (RIA) environment on Eclipse RCP.

Eclipse RCP offers a wide variety of features for the application developer. First, it offers a semantically complete and durable component model based on the OSGi bundles standard. The Workbench provides an application shell that developers can plug perspectives, editors, and views into. There are facilities and frameworks for building editors, help facilities, welcome pages, and defining resources.

The best part about the RCP is that it is very real and it is here now. Developers who are interested in using Java on the desktop don't have to wait for Mustang. Developers who are concerned about platform lock-in don't have to wait until whenever Longhorn ships. RCP enables multi-platform rich client application development *now*.

We've got problems with your name on them.

At Google, we process the world's information and make it accessible to the world's population. As you might imagine, this task poses considerable challenges. Maybe you can help.

We're looking for experienced software engineers with superb design and implementation skills and expertise in the following areas:

- high-performance distributed systems
- operating systems
- data mining
- information retrieval
- machine learning
- and/or related areas

If you have a proven track record based on cutting-edge research and/or large-scale systems development in these areas, we have brain-bursting projects with your name on them in Mountain View, Santa Monica, New York, Bangalore, Hyderabad, Zurich and Tokyo.

Ready for the challenge of a lifetime?
Visit us at <http://www.google.com/jdj> for information. EOE



RCP has been around for about a year. The big news with 3.1 is really the improvement in tools. The Eclipse Visual Editor now supports the ability to create Eclipse editors using SWT. The Plug-in Development Environment (PDE) has been improved to make it easier to create and deploy Eclipse rich-client applications.

And in case you're wondering, RCP is not just about SWT. You can build RCP applications that utilize Swing. For the Eclipse community, the Swing versus SWT debate is uninteresting. This is not a religious debate. It's about picking the best tool for the job at hand. For those enterprise de-

velopers and ISVs who want to build applications with a native look and feel, SWT is a great framework. For those who are less concerned about platform fidelity, we support Swing.

JDJ: *With the goal of building out tools for the entire software life cycle, where will the commercial vendors be left to compete? Is there a tension between having the whole life cycle tooled and the vendors that are strategic developers currently competing on that level?*

MM: Eclipse is not about supplanting commercial opportunity. It's about creating it.

It's a common misconception that Eclipse is about building tools. That is not the primary focus of our community.

The main objective of Eclipse is to create a universal development platform made up of frameworks and well-constructed APIs. Then we provide exemplary and extensible tools to demonstrate the use of the frameworks. But the point of this work is to build a platform on which vendors can implement their products. The tools are extensible so they can be further customized to meet the specific needs of commercial platforms.

Now many developers are perfectly happy to construct their toolset on top of Eclipse and open source plug-ins. In the case of our Java development tools, the adoption rate is impressive. But the majority of enterprise development shops are going to be looking for commercially supported and tested tool chains.

JDJ: *Turning to Swing/SWT interoperability, can you describe where it is and where it's going?*

MM: I really haven't seen a lot of new Swing/SWT interoperability features in Eclipse 3.1. Most of the feedback from our community has been that what we already have is pretty great.

JDJ: *Speaking of SWT, do you still think it's worthwhile to try to get cross-platform consistency from native code? Given that all this work has already been done for Swing, do you feel that SWT is still a good bet? Is it wise to repeat work that has been done for so many years in the Java code base?*

MM: Of course it is. Java is a great programming language. But the idea that Java should be constrained to only fit one style of application and one way of doing things is just lame. SWT has never been about supplanting Swing. It has been about providing a realistic and high-performance alternative for people whose application requirements demand it.



Mike Milinkovich
Executive Director
Eclipse Foundation



Eclipse is not about supplanting commercial opportunity. It's about creating it"

We believe that there are probably many more Swing applications being built with Eclipse than any other toolset.

The comment about "repeating work" is inaccurate. Swing continues to get better at emulating platforms, which is fundamentally different than the SWT strategy. We are not repeating work. We are implementing an alternative strategy for implementing GUIs with Java.

I believe that the competition from SWT has done more than any other factor to improve Swing. Competition is good. Choice is good. SWT is here to stay.

JDJ: Now that the RCP is well established with the release of

3.1, what direction will Eclipse be headed in the 4.0 time frame?

MM: Instead of talking about a specific release, let me talk about what we want to focus on in the next 12–18 months. A lot of this will be driven by the themes and priorities in our development roadmap. Some of the highlights will include:

1. Continue the evolution and adoption of RCP. We believe the technology around RCP is mature and real now. The focus needs to be on developing frameworks for RCP and having applications adopt RCP as their development platform.
2. Expect to see a lot of work and emphasis on providing tools

and frameworks for the embedded developer. We have just launched a new Device Software Development Platform project and I think you'll see some great stuff coming from it over the next 12–18 months.

3. We will continue to extend the set of projects across the software development life cycle
4. Finally, all of the projects are focused on ensuring their tools and frameworks can scale to meet the usability and performance challenges of enterprise development, including support for lots of code and lots of developers.

JDJ: Thanks, Mike, for talking with JDJ. ☺

What is the easiest way to create mission-critical mobile messaging applications in Java and J2EE?

By using NCL's Provato™ !

Used by Operators, Aggregators, Content Providers and Wireless Application Developers to create and deploy mobile applications, **Provato™** gets the message across.

Provato connects with the message centers (SMSCs and MMSCs) of operators and aggregators over industry standard protocols. Developed in 100% java, it handles routing, concurrent applications, multiple networks, message persistence, failover, connection management and integrates with network management. With support for SMS, long messaging, Unicode, binary, WAP Push and Multimedia Messaging, NCL's Provato is endorsed by operators and aggregators worldwide. Provato runs as a standalone server or deploys on a J2EE application server. Application interfaces include JMS, SOAP/XML and HTTP. A user-friendly browser-based interface simplifies (remote) management and configuration - your java application need only call send and receive. So now you can focus on the business logic of your messaging application.

For more details visit www.nclt.com/jdj
NCL Technologies Limited
Tel: +353 1 6761144 Email: jdj@nclt.com



Java Collections Framework & Managing Data

by Rolf F. Kamp

Every application has to consider how its data is stored, manipulated, and accessed. Fortunately for Java developers Sun provides commonly employed data structures as part of the Java platform in the Java Collections Framework (JCF). A framework is a set of well-defined interfaces that, if used properly, can benefit productivity tremendously, increase reusability, reduce software costs, and improve quality.

The JCF acts as a supporting structure for data manipulation. Well-thought-out choices in this arena make the difference between superior-performing software modules that are easy to build and maintain and those that have performance problems and are difficult to build and modify.

Interfaces and abstract classes are at the heart of the JCF. These describe the data structures and the operations supported by the data structure. A good understanding of the JCF can make for easy-to-maintain, easy-to-write modules based on a well-engineered framework.

In terms of data structures, a collection is a group of elements that can be treated as one unit. Consider why the following independent data elements can be stored in one entity called a collection:

- Employees belonging to one department
- Students in a grade
- Hourly temperature readings over a 24-hour period

Grouping these elements into a collection permits the easy management of these otherwise independent data elements.

Decisions related to how data is accessed dictate what data structure will be used to house the data. Consider how you organize everyday items. Your address book is in alphabetical order because you look up people by name, while a bag of M&Ms stores the M&Ms in no order at all because they're arbitrarily accessed in no certain order (or are you one of those people that eats M&Ms by color?).

Java's Historical or Legacy Collections

The JCF was introduced as part of the JDK1.2. Before the JCF, Java supported three collections, each of which had its own syntax for accessing elements. An array is fixed in size, can store primitives or objects, and references elements with square brackets ([]). Arrays house homogeneous data elements. That is, all elements in an array must be the same type. Arrays contain a data element named length that represents the number of elements comprising the array.

A variable can be used to specify the size of an array, but once declared, an array can't change size. This can result in specifying too large an array or, at runtime one array can be copied into a larger array. The System class provides an arraycopy() method that copies elements from one array into another.

If we need to house more than 24 ints in our temperatureReadings int array, the arraycopy() method from the System class can be used as shown below:

```
int temperatureReadings[] = new int[24];
// iarray can only house ints or primitives compatible with int
temperatureReadings [0] = 72;
temperatureReadings [0] = 98.6;
// will cause compile error-possible loss of precision
temperatureReadings [0] = new MyType();
// will cause compile error-incompatible types
int largerArray[] = new int[48];
System.arraycopy(temperatureReadings, 0, largerArray, tempera-
tureReadings.length);
```

Hashtable and Vector collections are part of the java.util package. A Hashtable implements a hash table data structure (also known as an associative array) by storing objects accessible by an arbitrary key. Objects in a Hashtable are referenced with the get() and put() methods. The put() method maps a key to a value. If the key exists in the Hashtable, the previous value is returned and the new value is mapped to the key.

A Hashtable can be used to store the number of members in an organization. Say we need to keep track of how many "Charter Members," "Gold Members," and "Silver Members" there are. Membership type is the key and the number of members in the membership type is the value.



Rolf F. Kamp is a Sun Certified Java Developer and has worked with Java in varied environments. In addition to his development work, he is an adjunct faculty member at Brookdale College, NJ.

rfk@att.com

Both arguments to the `get()` method are objects. If the value returned from `get()` isn't null, a value for the key already exists and is returned from `get()`:

```
Hashtable members = new Hashtable();
Integer oldValue;
oldValue = (Integer)members.put("charter", new Integer(123));
if (oldValue == null) {
    System.out.println("key charter did not exist in Hashtable");
} else {
    System.out.println("key charter contained value " + oldValue);
}
oldValue = (Integer)members.put("gold", new Integer(456));
if (oldValue == null) {
    System.out.println("gold did not exist in Hashtable");
} else {
    System.out.println("key gold contained value " + oldValue);
}
System.out.println(members.get("gold")); // prints 456
```

All keys existing in the Hashtable can be retrieved into an Enumeration by executing the `keys()` method:

```
members.put("charter", new Integer(90));
members.put("gold", new Integer(87));
members.put("silver", new Integer(154));
Enumeration enumeration = members.keys();
while (enumeration.hasMoreElements()) {
    String key = (String)enumeration.nextElement();
    Integer value = (Integer)members.get(key);
    System.out.println(key + " " + value);
}
```

A Vector stores objects and can change in size as objects are added or removed. There are many ways to add and remove objects from a Vector including Methods `add()`, `set()`, `setElementAt()`, `remove()`, and `removeAll()`. Vectors grow as needed. They are often used as "adjustable-sized" arrays, when the number of elements to be stored is unknown:

```
Vector vector = new Vector();
vector.add(new Integer(90));
vector.add(new Integer(83));
vector.add(new Integer(193));
Enumeration enumeration = vector.elements();
while (enumeration.hasMoreElements()) {
    Integer value = (Integer)enumeration.nextElement();
    System.out.println(value);
}
```

The ArrayList, discussed below, is like the Vector without the overhead of synchronization. Because the ArrayList isn't synchronized, it delivers four or more times the performance of a Vector. Of course using an array delivers the best performance.

The Java Collections Framework

Each historical collection had its unique syntax for managing elements. The JCF introduced the Collection interface – which the List and Set interfaces derive from – and the Map interface – which HashMap, Hashtable, and TreeMap derive from. In JDK 1.2, the Hashtable and Vector classes

were updated to implement the Map and List interfaces, respectively.

The JCF offers the developer a consistent set of methods for managing collections. As with most well designed frameworks, most of the key functionality is captured in common modules or interfaces. The JCF captures most of the functionality in the Collection, Map, List, Set, Iterator, and ListIterator interfaces. To understand the JCF, one must first understand these interfaces.

The Collection Interface

JCF's commonality of methods makes the framework easy to learn. The Collection interface that the Set and List interfaces derive from contains methods for managing collections. Regardless of the collection class employed, the following methods can be used to manage the collection. Once these methods are understood, most of the List and Set functionality is understood.

- **boolean add(Object)** – Adds the referenced Object to the Collection. Classes implementing the List interface will add the Object to the List. Classes implementing the Set interface will add the Object to the Set if it doesn't already exist. In all cases additional rules can be built in (such as not adding null Objects). A return value of true indicates the Object was added to the Collection (i.e., the Collection was changed)
- **void clear()** – Eliminates all the elements stored in the Collection.
- **boolean contains(Object)** – Indicates whether or not the Collection contains the referenced Object. Said differently,

The advertisement for Common Controls features a blue background with a white grid pattern. At the top, the logo "COMMON CONTROLS" is displayed in a stylized font, with the website "www.common-controls.com" to its right. Below the logo, a section titled "The Java™ Presentation framework for J2EE™ Web applications" is shown. Underneath this, a list of technologies it is based on includes Java™, Servlets™, Java Serverpages™, and Struts. To the right of this list is a small screenshot of a web application interface. A red starburst graphic with the text "For Free!" is positioned to the left of a promotional message: "Get your free trial version - www.common-controls.com. See the common controls in action - go for the Online Demo!". Below this, a statement reads: "Contains the most common control elements which are required for the development of J2EE™ applications with rich HTML frontends like:". This is followed by two rows of orange buttons with white text: "Lists", "Trees", "Tabfolders", "Menu", "Forms" in the first row, and "BreadCrumbs", "Calendar", "Colorpicker" in the second row. At the bottom, the website "www.common-controls.com" is repeated in a large, bold font.

“Using JCF effectively can make the difference between superior-performing software modules that are easy to build and maintain and those that have performance problems and are difficult to build and modify”

as long as the reference passed in isn't null, the return value of the Object's equals() method will be returned. Returns true if the Collection contains the argument.

- *boolean isEmpty()* – Indicates whether or not the Collection contains any elements. A return value of true indicates the Collection doesn't contain any Objects.
- *Iterator iterator()* – Returns an Iterator object that can be used to traverse the Collection.
- *boolean remove(Object)* – Eliminates an instance of the referenced Object. If multiple instances of the referred Object are elements in the Collection, just one instance is eliminated. The element eliminated is identified using logic similar to the contains() method. A return value of true indicates the referenced Object was eliminated from the Collection.
- *int size()* – Returns the number of elements in the Collection.
- *Object[] toArray()* – Creates an array housing all the elements in the Collection.

The Iterator and ListIterator Interfaces

Key to understanding the JCF is the methods used to access elements. The Iterator interface contains three methods that support the traversal and removal of elements in the Collection. It's similar in function to the legacy Enumeration interface.

- *boolean hasNext()* – A value of true is returned if the next() method would return an Object from the Collection. This method is typically used to govern a looping construct.
- *Object next()* – Returns the next element in the Collection.
- *void remove()* – Eliminates the element returned by the next() method. If an Object returned from next is identified as one to be removed from the Collection, this method is called.

The ListIterator interface used to traverse Lists implements the Iterator interface and adds methods supporting the bi-directional traversal of a Collection. Besides the Iterator methods, the ListIterator interface contains these methods:

- *boolean hasPrevious()* – Just as Iterator's hasNext() method returns true if the next() method will return an Object, hasPrevious() returns true if the previous() method will return an Object.
- *Object previous()* – Just as Iterator's next() method returns the next element in the Collection, this method returns the previous element in the Collection. Calls to previous() and next() can be made in an intermingled fashion.
- *void remove()* – Eliminates the element returned from next() or previous().

The Set Interface

In mathematics, a set is a collection of elements, all of which are unique. Java's Set interface implements the Collection interface and doesn't add any methods. No entry in a Set can occur more than once. An Object's equals() method is used when determining if an element is already in a Set. An element is considered in a Set if o1.equals(o2) returns true.

The java.util.HashSet and java.util.TreeSet classes implement the Set interface. The data structure housing the elements in a HashSet is a hash table. A hash table provides extremely efficient access to the elements in a HashSet (and a HashMap discussed below). A hash table generates a key for each stored element. Hash table data structures contain key/value pairs. The key is termed a hash code, which is used to access the desired element directly. Searching a hash table is very efficient, since, after the key is computed for the desired element, the element is directly accessed. All elements in a HashSet have their value set to null in their hash table internal to the HashSet. So accessing an element in a HashSet involves simply accessing the key or hash code.

A data structure imposing relations among elements producing a hierarchical structure with one element serving as a root is known as a tree. Elements in a tree can be unordered, indicating that there's no distinction between any elements; or ordered, indicating the placement of the elements in the tree has meaning or order. Java's TreeSet stores elements in a balanced tree meaning leaves in the tree are approximately the same distance from the root. This arrangement makes accessing and removing elements very efficient. Elements in a TreeSet are maintained in ascending sorted order. A TreeSet's Iterator visits elements in ascending order.

Assuming we have a Fraction class with properly defined equals() and compareTo() methods, the following code will place six Fraction objects in a TreeSet. Duplicate Fraction objects won't be added. The while loop will iterate through the TreeSet presenting the Fraction objects in ascending order:

```
TreeSet treeSet = new TreeSet();
treeSet.add(f1);
treeSet.add(f2);
treeSet.add(f3);
treeSet.add(f4);
treeSet.add(f5);
treeSet.add(f6);
Iterator iterator = treeSet.iterator();
while (iterator.hasNext()) {
    // lists unique Fractions in ascending order
}
```


Placing these Fraction objects in a HashSet looks syntactically similar but won't present the Fraction objects in order. This code will determine if a Fraction object exists in the HashSet:

```
HashSet hashSet = new HashSet();
hashSet.add(f1);
hashSet.add(f2);
hashSet.add(f3);
hashSet.add(f4);
hashSet.add(f5);
hashSet.add(f6);
if (hashSet.contains(f2)) {
    // Fraction equal to f2 is in hashSet
}
```

The List Interface

Lists are flexible data structures that shrink and grow as elements are inserted and deleted. Elements in list data structures can be accessed, deleted, or inserted at any position. Java's List interface permits duplicates and maintains insertion order. Generally speaking, there are two methods of representing a list data structure as an abstract data type, an array and a linked list. The java.util.ArrayList and java.util.LinkedList classes implement the List interface as an array and linked list, respectively. The LinkedList class should be used when elements will be frequently inserted and deleted. The ArrayList class should be used when elements are arbitrarily accessed frequently. In JDK 1.2 the

java.util.Vector class has been retrofitted to implement the List interface.

An ArrayList can be used to store all Fraction objects, including duplicate Fraction objects. This is the preferred collection to use if arbitrary access is common and insertions and deletions are infrequent:

```
ArrayList arrayList = new ArrayList();
arrayList.add(f1);
arrayList.add(f2);
arrayList.add(f3);
arrayList.add(f4);
arrayList.add(f5);
arrayList.add(f6);
Iterator aiterator = arrayList.iterator();
while (aiterator.hasNext()) {
    // lists all 6 objects in no order—even duplicate objects
}
```

A LinkedList will also store duplicate objects. This is the preferred collection to use if insertions and deletions are common and searching is minimal. The sort() method from the Collections interface can be used to sort any Collection as shown below:

```
LinkedList linkedList = new LinkedList();
linkedList.add(f1);
```



DynamicPDF™ Merger v3.0

Our flexible and highly efficient class library for manipulating and adding new content to existing PDFs is available natively for Java

- ♦ Intuitive object model
- ♦ PDF Manipulation (Merging & Splitting)
- ♦ Document Stamping
- ♦ Page placing, rotating, scaling and clipping
- ♦ Form-filling, merging and flattening
- ♦ Personalizing Content
- ♦ Use existing PDF pages as templates
- ♦ Seamless integration with the Generator API

DynamicPDF™ Generator v3.0

Our popular and highly efficient class library for real time PDF creation is available natively for Java

- ♦ Intuitive object model
- ♦ Unicode and CJK font support
- ♦ Font embedding and subsetting
- ♦ PDF encryption ♦ 18 bar code symbolologies
- ♦ Custom page element API ♦ HTML text formatting
- ♦ Flexible document templating

Try our free Community Edition!



DynamicPDF™ components will revolutionize the way your enterprise applications and websites handle printable output. DynamicPDF™ is available natively for Java.



```

linkedList.add(f2);
linkedList.add(f3);
linkedList.add(f4);
linkedList.add(f5);
linkedList.add(f6);
Collections.sort(linkedList); // Fraction's compareTo() method
used to sort objects
Iterator iterator = linkedList.iterator();
while (iterator.hasNext()) {
    // lists 6 objects in sorted order
}

```

The Map Interface

Java arrays, ArrayLists, LinkedLists, and Vectors store elements based on an offset expressed as an integer key. Data structures known as maps (or map) associate elements of one type (called keys) to elements of another type (known as values). Java's `java.util.Map` interface establishes an Object as a key. This facilitates associating an Object (serving as a key) with a value. For example, we can associate the String object "SUNW" with its share price of, say, 4. Many languages refer to maps as associative arrays.

Classes implementing this interface commonly use these methods from the Map interface:

- `void clear()` – Removes all elements from the Map.
- `boolean containsKey(Object)` – Returns true if there is a key used by the Map equal to this Object. There will be one and only one key equal to an Object. Equality is determined by the Object's `equals()` method.
- `boolean containsValue(Object)` – Returns true if there is a value used by the Map equal to this Object. There can be any number of values equal to an Object. Equality is determined by the Object's `equals()` method.

- `Set keySet()` – Returns a Set of keys for this Map.
- `Object get(Object)` – Returns the value of the Object serving as a key. Equality is determined by the Object's `equals()` method.
- `Object put(Object k, Object v)` – Maps a key (k) to a value (v). If a mapping exists for the key, the previous value is returned and replaced with the specified value.

Since Maps aren't Collections, they don't contain iterators. When it's necessary to iterate through a Map, the `keySet()` method can be used to acquire the Set of keys in the Map. The Set of keys can then be iterated through, accessing each key/value mapping.

Two classes implement the Map interface, `java.util.TreeMap` and `java.util.HashMap`. A class implementing the comparable interface can be stored in a `TreeMap`, since the `TreeMap` is sorted according to the keys. The `HashMap` doesn't store keys in any order and so it's the most efficient way of directly accessing via key Objects:

```

HashMap hm = new HashMap();
hm.put("A", f1);
hm.put("B", f2);
hm.put("C", f1);
hm.put("A", f3);
hm.put("D", f4);
Set hmKeys = hm.keySet();
Iterator setiterator = hmKeys.iterator();
while (setiterator.hasNext()) {
    String key = setiterator.next().toString();
    Fraction value = (Fraction)hm.get(key);
    System.out.println(key + " " + value.toString());
}

```

A `TreeMap` will store keys according to the key's `compareTo()` method. If implemented in the typical manner, keys are added in ascending order:

```

TreeMap tm = new TreeMap();
tm.put("A", f1);
tm.put("B", f2);
tm.put("C", f1);
tm.put("A", f3);
tm.put("D", f4);
Set tmKeys = tm.keySet();
Iterator setiterator = tmKeys.iterator();
while (setiterator.hasNext()) {
    String key = setiterator.next().toString();
    Fraction value = (Fraction)tm.get(key);
    System.out.println(key + " " + value.toString());
}

```

Summary

The JCF offers the developer a powerful set of classes to manage data. One JCF class doesn't do the job for all applications. Carefully selecting an appropriate JCF class makes all the difference between a well-performing, easy-to-maintain module and one that performs poorly and is difficult-to-maintain. ☞

| JCF Class | Positioning | Ordering | Duplicates | Sample Use |
|------------|---------------|-----------------|-------------|--------------------------------------------------------------------------|
| TreeSet | None | Ascending | Not Allowed | Keep unique list of words from a document sorted |
| HashSet | None | None | Not Allowed | Keeps a unique list of words from a document to be referenced frequently |
| ArrayList | Numerical | None | Allowed | Keeps an "in-memory" lookup table of static data |
| LinkedList | Numerical | Insertion Order | Allowed | Keeps a list of words that change frequently |
| HashMap | Object as key | None | Unique Keys | Maps words to word frequency |
| TreeMap | Object as key | Ascending | Unique Keys | Maps sorted words to word frequency |

JCF Classes Cheat Sheet

Overriding equals() and hashCode() and the Comparable Interface

The java.lang.Object class – from which all java classes are derived – contains the equals() and hashCode() methods. JCF classes use these methods to do sorting and ordering. From Sun's documentation of java.lang.Object, we see the following rules for overriding equals() and hashCode():

- When given null equals() must return false (o.equals(null) = false)
- equals() is symmetric (when o1.equals(o2)=true then o2.equals(o1) = true)
- An Object is always equal to itself (o.equals(o) = true)
- equals() is transitive (when o1.equals(o2)=true and o2.equals(o3)=true then o1.equals(o3) = true)
- When two Objects return the same value from hashCode(), they must return true from equals()
- hashCode() must return the same value when run on the same Object when executing a Java program
- equals() must consistently return the same value

The equals() method for the Fraction class first checks that the Object passed in is not null and is a Fraction. Next, each Object's numerators and denominators are compared for equality. Additional code is required if a subclass of Fraction is created:

```
public boolean equals(Object anotherFraction) {
    if (anotherFraction == null) return false;
    if (!(anotherFraction instanceof Fraction)) return false;
    return ((numerator == ((Fraction)anotherFraction).getNumerator()) &&
        (denominator == ((Fraction)anotherFraction).getDenominator()));
}
```

The following hashCode() method returns an integer representation of the Object's numerator and denominator by performing floating-point division of the numerator and denominator and multiplying the quotient by a large value:

```
public int hashCode() {
    return (int)(((double)numerator / denominator) * Integer.MAX_VALUE);
}
```

The Comparable Interface

The java.lang.Comparable interface declares the compareTo() method. A collection uses an Object's compareTo() method to determine an Object's natural ordering. The compareTo() method returns an integer indicating whether the Object is equal to (value of 0 returned), greater than (value greater than 0 returned), or less than (value less than 0 returned) the Object passed in. The java.lang.ClassCastException is thrown if the Object passed in isn't the expected type.

```
class Fraction implements Comparable {
    private int numerator, denominator;
    public Fraction() { numerator = 1; denominator = 1; }
    public Fraction(int numerator, int denominator) {
        this.numerator = numerator; this.denominator = denominator;
    }
    public int getDenominator() { return denominator; }
    public int getNumerator() { return numerator; }
    public int hashCode() {
        return (int)(((double)numerator / denominator) * Integer.MAX_VALUE);
    }
    public int compareTo(Object anotherFraction) throws ClassCastException {
        if (!(anotherFraction instanceof Fraction)) {
            throw new ClassCastException("Fraction expected");
        }
        return numerator * ((Fraction)anotherFraction).getDenominator() -
            denominator * ((Fraction)anotherFraction).getNumerator();
    }
    public boolean equals(Object anotherFraction) {
        if (anotherFraction == null) return false;
        if (!(anotherFraction instanceof Fraction)) return false;
        return ((numerator == ((Fraction)anotherFraction).getNumerator()) &&
            (denominator == ((Fraction)anotherFraction).getDenominator()));
    }
}
```


JDBC 4.0: Features Worth the Wait

by John Goodson

A significant advance on the standard

Pooling is great — except it's not very tunable, it's hard to map end users back to connections in the pool, and if a connection ever becomes invalid inside the pool, expunging only that connection from the pool is nearly impossible; JDBC 4.0 addresses all these drawbacks

As a member of every previous JDBC Expert Group, it sometimes seems as if the specification process moves too slowly for the few features being added or it seems as if the new feature list is good, but “not good enough.” The JDBC 4.0 specification fits neither of these perceptions. The specification includes a lot of new features — too many, in fact, to describe in this article. As this article was written a few months before the publication of the public draft of the JDBC 4.0 specification, it's possible that some features, features even more significant than those described here, might make it into the preview release. Most developers will be pleasantly surprised at the enhancement list, which includes everything from performance-tuning options to support for extended-level database features. Here we'll describes in detail some of the new features that are available in JDBC 4.0, along with the reasons why those features are important.

The JDBC 4.0 specification is now in public review and will ship as part of J2SE 6.0 sometime in 2006. The key goals of the JDBC Expert Group were to align with the most important features of the SQL 2003 specification, provide constructs that help improve developer productivity

(sometimes called Ease of Development or EOD features), fine tune pooling constructs, and improve scalability. Unlike the JDBC 3.0 specification, there are some major new additions in the JDBC 4.0 spec, such as XML support. Overall, it's not just a collection of bug fixes to the JDBC 3.0 specification; JDBC 4.0 is clearly a significant advance of the standard.

XML Support

One of the most useful new features in JDBC 4.0 is support for the SQL 2003 XML data type. That is, the JDBC spec has been expanded to support XML data types in the database, Java XML bindings, and SQL/XML extensions to the SQL grammar. Although many databases support XML data types today, applications must use either JDBC driver extensions to transfer data to or from the database or use the Clob interface, which is limited in nature to a string representation of the XML.

A new SQL data type, SQLXML, is part of the specification. Applications can use `getTypeInfo()` to determine if their database supports a native XML data type. For example, `getTypeInfo()` against a SQL Server 2000 instance doesn't return a result row corresponding to the SQLXML data type, indicating that there's no native XML data type available for that particular database backend. By contrast, `getTypeInfo()` against a SQL Server 2005 instance returns a result row, indicating that an XML data type is available. It also returns information indicating that the native type name is “XML.” From this

information, applications can create tables that contain columns of the XML data type.

So applications can populate data in XML columns and retrieve data from those columns, JDBC has been expanded to include native Java bindings for XML. Initially, the Expert Group looked at adding all types of XML bindings, such as text, JAXP, DOM, JDOM, SAX, stream, and StAX. In the end, allowing so many different bindings had too many drawbacks. The Expert Group decided to allow bindings for Java strings, Java character streams, and StAX.

```
// JDBC XML Data Type
package java.sql;
import java.io.InputStream;
import java.io.OutputStream;
import javax.xml.stream.XMLStreamReader;
import javax.xml.stream.XMLStreamWriter;

public interface SQLXML {
    public XMLStreamReader createXMLStreamReader() throws SQLException;
    public XMLStreamWriter createXMLStreamWriter() throws SQLException;

    public String getString() throws SQLException;
    public int setString() throws SQLException;
}
```

If an application needs to process data inside DOM or SAX, for example, it's a straightforward exercise to convert that data from a StAX stream to those representations.

To create a Java construct that can be used to process XML data, an application can create a SQLXML object off of the Connection object using the `createSQLXML()` method. The object that's created doesn't contain any data initially. Data can be added to the object by calling `setString()` or associating an XML stream using `createXMLStreamWriter()` with the object. The following code illustrates how



As vice-president of product operations, **John Goodson** leads the product strategy, direction, and development efforts at DataDirect Technologies. For more than 10 years, he has worked closely with Sun and Microsoft on the development and evolution of database connectivity standards including J2EE, JDBC, .NET, ODBC, and ADO. His active memberships in various standards committees, including the JDBC Expert Group, have helped Goodson's team develop the most technically advanced data connectivity technologies. He holds a BS in computer science from Virginia Tech.

jgoodson@datadirect.com

“So applications can populate data in XML columns and retrieve data from those columns, JDBC has been expanded to include native Java bindings for XML”

an application can use these techniques to insert a row containing XML data into a table.

```
ResultSet rs = stmt.executeQuery ("select XMLC1 from T1");
while (rs.next()) {
    // Iterate through results and process
    SQLXML xmlvalue = rs.getSQLXML("XMLC1");
    XMLStreamReader reader =
        xmlvalue.createXMLStreamReader();

    for (int event = reader.next();
         event != XMLStreamConstants.END_DOCUMENT;
         event = reader.next()) {

        switch (event) {
            case XMLStreamConstants.START_ELEMENT:
                // ...
                break;
        }
        xmlvalue.free(); // free resource
    }
}
```

Similarly, applications can retrieve a string representation of the XML using the `getString()` method of the object or by associating an XML stream using `createXMLStreamReader()`. The following code illustrates how an application can SELECT a column of the SQLXML data type, create a SQLXML Java binding using `getSQLXML()` on the result set, and read the value into the new object using a STAX representation via `createXMLStreamReader()`.

```
// foo is a table: integer and XML columns
String sql =
    "insert into foo (icol,xmlcol) values (?, ?)";
PreparedStatement pstmt =
    connection.prepareStatement(sql);

int idVal = 1;
pstmt.setInt(1, idVal);

SQLXML xmlvalue = connection.createSQLXML();
XMLStreamWriter writer = xmlvalue.createXMLStreamWriter();
// - write events to the stream - code not supplied
writer.close();
pstmt.setSQLXML(2, xmlvalue);
int rowCount = pstmt.executeUpdate ();
```

SQL 2003 also includes extensions to the SELECT syntax that lets you construct XML results from tabular columns. The following code shows how to create a SELECT statement that produces a result set containing two columns: a `CustId` result column of type integer and a `CustInfo` column of type SQLXML.

```
select
    c.CustId,
    xmlelement(name customer,
        xmlelement(name name,c.Name),
        xmlelement(name address,c.Address)) as CustInfo
from Customers c
```

| CustId | CustInfo |
|--------|--------------------------------------------------------------------------------------------------------------------------------------------------------|
| 1 | <pre><customer> <name>Woodwork et/naah</name> <address>Baltimore et/address</address> </customer></pre> |
| 2 | <pre><customer> <name>Software Solutions et/naah</name> <address>Boston et/address</address> </customer></pre> |

The SELECT statement uses the new SQL/XML extension `XMLELEMENT` to process multiple base columns into a single XML result column. JDBC 4.0 has also been expanded to support using database metadata methods to determine which SQL/XML constructs are supported on the connection. Applications can then execute any supported SELECT statement with SQL/XML extensions to produce SQLXML result columns that can be processed using the new XML Java bindings.

Connection and Statement Pooling Enhancements

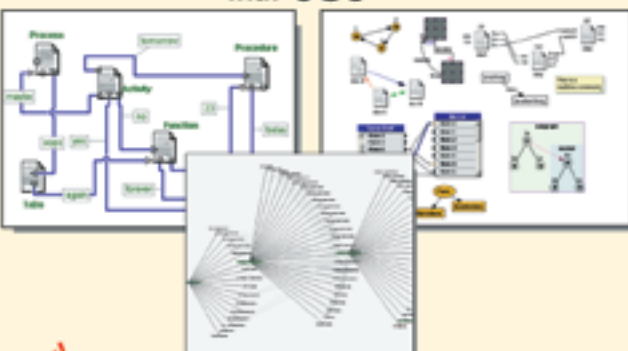
Most deployed JDBC applications use connection pooling, statement pooling, or a combination to obtain better application performance. Pooling is great — except it's not very tunable, it's hard to map end users back to connections in the pool, and if a connection ever becomes invalid inside the pool, expunging only that connection from the pool is nearly impossible. JDBC 4.0 addresses all these drawbacks.

Currently, prepared statement pooling is very atomic — either statement pooling is on or it's off. This kind of operation doesn't fit the programming model many applications use. In common deployments, it's likely that an application will have a certain set of SQL statements that are re-executed multiple times and a few SQL statements that might only be executed once or twice during the life of the application. Unfortunately, existing statement pooling implementations give no weight to a SQL statement executed 100 times versus one that's executed only twice. Again, either a statement goes into the pool, potentially causing another statement to be removed from the pool, or there's no pool. JDBC 4.0 provides a more granular level of statement pooling by letting applications hint to the pool manager about whether a SQL statement should be pooled.

The `PreparedStatement` interface has been expanded by the addition of two new methods: `isPoolable()` and `setPoolable()`. The `isPoolable()` method returns a Boolean flag that denotes whether the SQL statement identified on the prepared statement object should be pooled (by default, a statement is poolable when it's created). Applications can specifically request that a statement not be pooled by calling

Build Incredible Interactive Diagrams

with **JGo™**




New!
JGo for SWT/Eclipse
JGo Instruments for meters, dials, gauges

Create custom interactive diagrams, network editors, workflows, flowcharts, and design tools. For web servers or local applications. Designed to be easy to use and very extensible.

- **Fully functional evaluation kit**
- **No runtime fees**
- **Full source code**
- **Excellent support**

Free evaluation at:
www.nwoods.com/go
800-434-9820 or 603-886-9173



setPoolable(false). Using these constructs, application designers gain more control over the performance aspects of their applications. Queries that are reused are pooled and provide optimal performance, and queries that are used infrequently don't affect the pool.

Connection pooling is a mature feature available in all J2EE application servers and is used in many standalone Java applications. One might think that a technology that's been available for so long would have all the kinks worked out. JDBC 4.0 addresses some major concerns of connection pooling and discussions are already underway on connection pooling enhancements for the JDBC specification post-4.0.

Today, when the response time of your database queries is ridiculously slow because your application server is out of CPU cycles, your database appears to be "hung," or you try to monitor the status of your applications only to see that "some JDBC connection" is using all the CPU, the facilities available to help you find the culprit aren't very good. Once a JDBC connection is established, the tracking mechanism between that physical connection and

name, and department name for the JDBC connection. Monitoring tools can then retrieve this information to help pinpoint where the problem is.

Large-scale deployments often face another problem when a pool is populated with a large number of connections. How does the pool manager detect when a connection has become invalid? Today, there's no facility inside a JDBC driver to check and see if a connection is still valid. The Connection.isClosed() method is sometimes mistakenly thought to do this, but the intent of isClosed() is to check and see if a connection is open or closed, not whether the connection is still usable. If a connection pool manager decides that a connection is invalid or is suspect (through whatever proprietary means is available), the most common technique used is for the pool manager to terminate all the connections in the pool and re-initialize it. This is a very drastic approach to take and is extremely expensive in terms of performance. A new method on the Connection interface, isValid(), has been added so pool managers can specifically request the driver if a con-

JDBC 4.0 expands the java.sql.package's exception hierarchy by providing two distinct subclasses that indicate whether exceptions are transient (and might succeed if retried) or aren't transient (and won't succeed if retried). These subclasses are: SQLNonTransientException and SQLTransientException. SQLNonTransientExceptions are subclassed further into five distinct cases: SQLSyntaxErrorException, SQLInvalidAuthorizationSpecException, SQLIntegrityConstraintViolationException, SQLDataException, and SQLNonTransientConnectionException. SQLTransientExceptions are subclassed into three distinct cases: SQLTimeoutException, SQLTransactionRollbackException, and SQLTransientConnectionException.

The idea behind this change in the specification is that applications might only be concerned with whether this error is "expected" or not. If it's not expected, the operation can just be retried and may well succeed. In this case, there's no checking 30 different SQLStates to see if the statement should be re-executed. An application just checks to see if the SQLException was a SQLTransientException and can then be retried.

“Connection pooling is a mature feature available in all J2EE application servers and is used in many standalone Java applications”

an application's use of the logical connection is lost. The connection pool manager assigns physical connections in the pool to any application that meets authentication requirements; the pool manager doesn't keep any statistics on the application requesting a connection, and the connection itself is a black box to the application. In other words, if you are using a monitoring tool and see that a JDBC connection is "bogging down the system," it's impossible to track down which JDBC application is actually invoking the driver.

To solve this problem, JDBC 4.0 has added setClientInfo() and getClientInfo() to the Connection interface. After connecting, an application can call setClientInfo() to associate client-specific information to the JDBC connection object, such as application name, site

nection is still usable. If a connection is invalid, the pool manager can discard only the marred connection and not the contents of the entire pool.

SQLException Improvements

JDBC 4.0 is meant to make it easier for developers to write JDBC applications. There are too many changes to the specification to describe all of the "ease of development" features here; however, one of the features we'll talk about is handling SQLExceptions. Applications can call getSQLState() when a SQLException happens to get the details about the cause of the error. The problem developers face is that there are many different SQLStates that can be returned. Programmatically figuring out what higher-level reason caused the error is straightforward, but time-consuming, error-prone, and monotonous.

If an application needed to determine whether the error was a "programming error," such as an invalid data conversion, it could check the SQLException to see if it was an SQLDataException. Programming a single check is much easier than checking 12 different SQLStates.

Other New Features

We can only touch on a few of the many enhancements for JDBC 4.0 here. It includes support for a new ROWID data type, bindings for the National Character Set, improved management of Clob and Blob objects, an improved mechanism for installing and recognizing JDBC drivers on a system, new annotations and interfaces, and extensive JDBC specification clarifications. Please take a look at the JDBC 4.0 (JSR-221) details on jcp.org and provide feedback that's relevant to your JDBC use. ☛

Open Up!

Portals—Get what
you want easily



company, product and service names may be trademarks or service marks of others.

Ask the Experts

Prolifics is 1 of only 3 WebSphere Service Providers retained by IBM. Open up with portals! With 26 years of industry expertise, Prolifics' certified WebSphere portal experts deliver a personalized, secure, unified view to your enterprise assets.

Our portal solutions build agility into your business — enabling you to reach new markets, achieve customer satisfaction beyond expectation, and ease collaboration within your enterprise. Our teams of specialists work together with your business experts from gathering the business requirements, translating them to a technical specification, to delivering the final production application.

Portal Executive Assessment

Prolifics experts help you articulate the value of portals to your business sponsors and perform a quick business review identifying portal benefits as it applies to your organization.

Portal Workshops and Proof of Concepts

Prolifics demonstrates valuable portal features and translates the breadth of your portal objectives into a step-by-step roadmap to meet your current and future needs.

Portal Jump Start

Prolifics' consultants work with you from the start to install, configure and jump start your team on using WebSphere Portal Software.

Portal Development and Mentoring

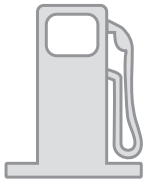
Prolifics' innovative approach to building portals drives down your overall cost and transforms the way you conduct business. We mentor and arm your team with the skills to manage, maintain and extend the portal environment.

Prolifics

Winner of 2003 IBM BUSINESS PARTNER
LEADERSHIP AWARD

116 John Street, New York, New York 10038
800.675.5419
solutions4websphere@prolifics.com
www.prolifics.com

Prolifics 



Small Business Solutions

Part One

by Yakov Fain

Several years ago I was thinking about buying a small gas station in my local town. I went to my friend Gregory Z., a successful businessman in this field, and asked him, "How do I start a gasoline business?" He gave me simple but wise advice: "You know nothing about gas, but know a lot about computers. Keep doing what you're doing. Just be a little better than others".

I'm trying to follow his advice but I keep thinking how would I apply my software skills had I bought such a business. So here I am again asking for your help, advice, and experience: let's automate my virtual gas station.

The Setup

I've borrowed the money from a bank and now I have:

- A four-car gas station
- A small convenience store (coffee, cigarettes, milk, newspapers)
- A repair shop that changes oil, brake pads, and tires
- Six employees: one American, two from India, one from Russia, and two from Pakistan; one employee speaks English, and the others speak well in their native languages. I think they have work permits.
- Three decent Wintel computers
- \$1,000 USD software budget

Service-Oriented Architecture

The only difference between my gas station and Wal-Mart is that they have more suppliers, sell more products, and have more customers. But I'm facing similar challenges: I need to deal with various suppliers of gas, food products, and car parts. I also need to have accounting and payroll systems. That's why I'm planning to *architect* a system that has different *services* communicating with each other. I need a service-oriented architecture (SOA).

In a perfect world, all service providers use the same protocol, which is as challenging as having all my employees use only the English language. My

employees are trying hard, because for them it's a matter of surviving. Initially they are exposing just a minimal number of *public services* like fillItUp, getCash, processPlastic, marlboro-LightsPlease, oilChange, and takeTip. Smarter employees quickly add more services to their vocabulary to become more competitive. Each of these *coarse-grained* services may consist of several smaller steps, but consumers don't need to know about them.

Service Coupling

When the guy who sold me the business gave me a pile of different forms to use with suppliers, I thought to myself, "*Tight coupling in action.*" I need to know where these suppliers are located, their services, and how to request them. If a particular vendor changes its request form (*the protocol*), I'll need to get a new one, otherwise I may lose this service. I'd rather be sending a message to some destination saying, "Yakov needs 1,000 gallons of 93-octane gasoline." Expected response: several price quotes from different vendors. This would be an example of *decoupled* services. I don't know who they are and they don't know who I am, but we've *dynamically discovered* each other. I've heard that Jini could help me with this. Is this right? Anyway, in a *loosely coupled* system a service requestor needs to know the name of the service, what data to provide, and what to expect back, but it should be easy to switch from one provider to another.

Messaging and Transport

One of the best ways to request and receive services is by using asynchronous messaging. JMS is an excellent API, but you still need a transport to deliver your messages between the services, for example, message-oriented middleware (MOM). IBM and Tibco offer great MOM products, but I'd need to sell my gas station and get another loan just to pay for it. No, I need to find something for free.

Web Services

Web services seems to be a decent way to arrange my interaction with external suppliers using a free Internet-HTTP-WSDL-UDDI-SOAP combo. If my external vendors will start publishing their gasoline-tires-milk quotes, I'll write a program that will automatically be looking for the best deal in my neighborhood. Can Eclipse IDE help me with automatic generation of all supporting files for Web services?

OOP Plus AOP

It goes without saying that I'm *thinking object* (thank you, Bruce). When I close my eyes, I clearly see the classes Product, Order, and Customer...but after reading about aspect-oriented programming (AOP) these objects become blurry. But AOP is a way to go and I'll dig more in this direction.

Storing My Data

I need a free DBMS. It doesn't have to be fancy and implement sophisticated SQL constructs. Inner and outer joins plus indexing will do. Coding business logic in stored procedures is not in fashion these days. Hibernate looks nice for object-relational mapping. Is this my best option?

Front End

For thin Web clients I'm planning to learn AJAX. If Google did it, so can I. How about rich clients? Swing is not there yet; SWT looks better; .NET is the best but isn't free. Should I spend my money on VB.NET? But I don't know VB! Let me ask my Russian employee Alex if he knows it. He looks like a PhD (does he really have a work permit?).

Help

I'd love to hear your input to this new column. You don't need to write any code, but rather suggest some affordable tools and architectural solutions for my small business. Just provide your feedback to the online version of this article at <http://jdj.sys-con.com/read/108260.htm>. ☎



Yakov Fain is a J2EE architect and creator of seminars "Weekend with Experts" (www.weekendwithexperts.com). He is the author of the best-selling book *The Java Tutorial for the Real World* and an e-book *Java Programming for Kids, Parents and Grandparents*. Yakov also authored several chapters for *Java 2 Enterprise Edition 1.4 Bible*.
yakovfain@sys-con.com



CONFERENCE:
AUGUST 8 – 11, 2005

EXPO:
AUGUST 9 – 11, 2005

Moscone Center West
San Francisco, CA

WHERE **open minds** MEET > >

explore > > analyze > > gain > >

LinuxWorld Conference & Expo is the world's leading and most comprehensive event focusing on Linux and Open Source solutions. At LinuxWorld, see and learn how to best leverage the technology for your organization.

- > **Explore** your options on the exhibit hall floor, which features the world's leading hardware and software vendors.
- > **Analyze** the latest Linux and Open Source technology and discover how companies across the globe can show you how to achieve higher profits and increase productivity.
- > **Gain** knowledge about best practices and solutions by attending LinuxWorld's outstanding educational program.

It's the Linux & Open Source event you can't afford to miss!



linuxworldexpo.com



> Register Online With **Priority Code: D0106**

PLATINUM SPONSORS





Joe Winchester
Desktop Java Editor



How Much Is that Buggie in the Program?

London, the capital of my home country England, has a beautiful gothic style lifting bridge built by the Victorians in 1894 that magnificently spans the river Thames. It allows tall ships to access the river upstream by lifting its center sections, which for the first 82 years of its life was powered by huge steam engines.

Steam has since given way to electricity and in 1998 a \$3M overhaul was done to upgrade the kit and make it ready for the 21st century. On June 3, 2005, however, everything did not go according to plan and the bridge was stuck open. For 10 hours it remained jammed open while police diverted angry motorists to alternative crossings and the engineers worked against the clock to figure out what had caused the historic monument to malfunction. The reason given when she finally came down was that a software error had caused the problem <http://news.bbc.co.uk/1/hi/england/london/4605743.stm>. This problem is not an isolated one and was the fourth to occur in three months.

Two thousand years ago the Romans employed an interesting motivational technique: once engineers had finished building a bridge they had to stand under it while the first legion of soldiers marched across. I wonder if the Tower Bridge IT manager wished he'd have done similarly with his programmers when he got hauled before his superiors to answer why one of the main thoroughfares from South to North London was out of action.

One of my very first IT managers used to ban us from using the word "bug" and had us the noun "defect" instead. His wisdom was that the word "bug" was used by a programmer as a way of shirking responsibility, that the problem was of his or her own making and poor workmanship had caused it to occur. The origin of the term is reputed to have arisen from a moth found between the relay terminals of a calculating machine; it's sobering that

despite all of the advances in software engineering that have occurred since, problems still occur and, worse than that, are expected and even planned for.

Bugs are expensive to fix, and in Keynesian Economics the value of anything is determined as being the cost of the alternative. What is the cost of errors in code?

In 1996 the European Space Agency rocket Ariane 5 exploded 40 seconds after launch at a cost of \$7B due to a straightforward software defect. A data conversion from 64-bit floating point to 16-bit integer threw an exception when the floating point became too large.



The Mars Climate Orbiter in 1998 was destroyed when instead of entering the atmosphere at 90 miles above the surface, it dropped in at around 40 and subsequently burned up. The reason was that some data on the ground was calculated in imperial pounds and reported to the navigation team who thought it was metric newtons.

More recently on January 21, 2004, the NASA Mars Spirit Rover on Mars stopped communicating with Earth. The problem was the file management software that wrote to the rover's flash memory was unable to deal with the volume of data that was occurring at the time and threw an exception fault that crippled the whole unit. Fortunately this was corrected, although by a wing and a prayer – the fix would use the rover's RAM instead of the flash memory, delete a set of in-flight data files no longer needed to reclaim space, reformat the memory and, after

three weeks, the Spirit was up and running again.

Crashing rockets is a very visible and costly failure, but it doesn't have to be such a stellar failure when shipping defective code. Is there any such thing as an inexpensive bug, given that any defective piece of software represents bad function?

The problem with defects is that while they occur, the cost of finding and preventing them has a diminishing return, so the approach often taken is that once no more serious defects can be found in a test pass, all that remains must be minor and the programming is complete. The whole act of testing is an odd part of the software engineering process, because the expectation is that bugs will be found and then fixed before the next round of testing occurs. Edsger Dijkstra, one of the grandfathers of modern computing, once wrote: "Testing can only prove the presence of bugs, not their absence."

Testing therefore is not the verification that a program works, but a search for whatever bugs can be found within the time and scope constraints of its execution. In an odd way the whole process of testing sort of vindicates the fact that programming creates malfunctioning code that needs checking and rechecking before it can be shipped.

What troubles me is that we, as software engineers, aren't doing enough to really create error-free software. Does software have to be buggy because of its size and complexity, or do we use that as an excuse to throw more code at an application when we know its existing code base is flawed? Why is a successful test pass measured as one that finds lots of bugs, and not one that gives the program a clean bill of health? Another of Edsger's words of wisdom summarize eloquently; "If debugging is the process of removing bugs, then programming must be the process of putting them in." ●

Joe Winchester is a software developer working on WebSphere development tools for IBM in Hursley, UK.

joewinchester@sys-con.com

Register Online
www.eclipseworld.net

Attend EclipseWorld, the enterprise development conference!

EclipseWorld is for enterprise developers, architects and development managers who want to take their company's applications to a higher level!

At EclipseWorld you will:

- Save money and improve developer productivity with Eclipse.
- Go beyond the IDE to master the wide range of Eclipse technologies.
- Discover the best, most effective Eclipse add-ins and plug-ins.
- Master techniques for building high-quality, more secure software.
- Get deep inside Eclipse's open-source architecture.
- Improve team collaboration using Eclipse.

Over 50 Tutorials & Classes To Choose From!



BZ Media is a member of the Eclipse Foundation.

**Early Bird
Rates
Expire
July 29!**



August 29-31, 2005



The Roosevelt Hotel • New York City

Produced by **BZ Media**

Platinum Sponsor

SYBASE®

Gold Sponsors



Media Sponsors

SDTimes
The Industry Magazine for Software Development Managers

**Software Test
& Performance**

EclipseSource



queue
acm
advancing a computing today

JDJ

Software

WebSphere®

Methods & Tools

**Extension
MEDIA**

EclipseWorld™ is a trademark of BZ Media LLC.
Eclipse™ is a trademark of Eclipse Foundation Inc.

www.eclipseworld.net

Rich-Client Portlets

The half-object + protocol design pattern

by Marc Domenig

The portlet specification defines a standard infrastructure for Web portals in J2EE. While this infrastructure helps reduce proprietary code in HTML applications, it poses a challenge in Rich-Internet Applications (RIA) that have a hard time managing the state of their user interface appropriately.

An efficient solution rests with RIA approaches that are based on the half-object + protocol design pattern, a pattern that maintains the state of the user interface (UI) on the server side, a crucial enabler for rich-client portlets.

Portlets as defined by the Java Specification Request (JSR) 168 are an emerging standard in developing Web portals. They allow aggregating content, server-side Java applications, and SOAP-based Web Services in a coherent browser front-end that can be configured flexibly.

The Portlet Switching Issue

Unfortunately, this new standard has a shortcoming: it's incompatible with most rich-client technologies. The problem is that portlets must maintain the state of their UI when the user switches back and forth between them. Typical rich-client technologies can't cater to this need, because an active page's UI will be discarded whenever a new browser page is loaded. So the state of a portlet's UI is lost when the user switches to another portlet, and can't be restored when the user returns.



Marc Domenig is the CEO of Canoo. He holds a PhD in linguistics and computer science.

marc.domenig@canoo.com

Tackling this issue requires that the UI's state be saved when a portlet is switched. Doing this is difficult and expensive for technologies that execute their entire presentation layer on the client. Since there's no generally available option to save the browser state on a client, the state has to be saved back to the server and restored from there. Doing this is both ineffective and complex (see Figure 1).

Server-Side Proxy Solution

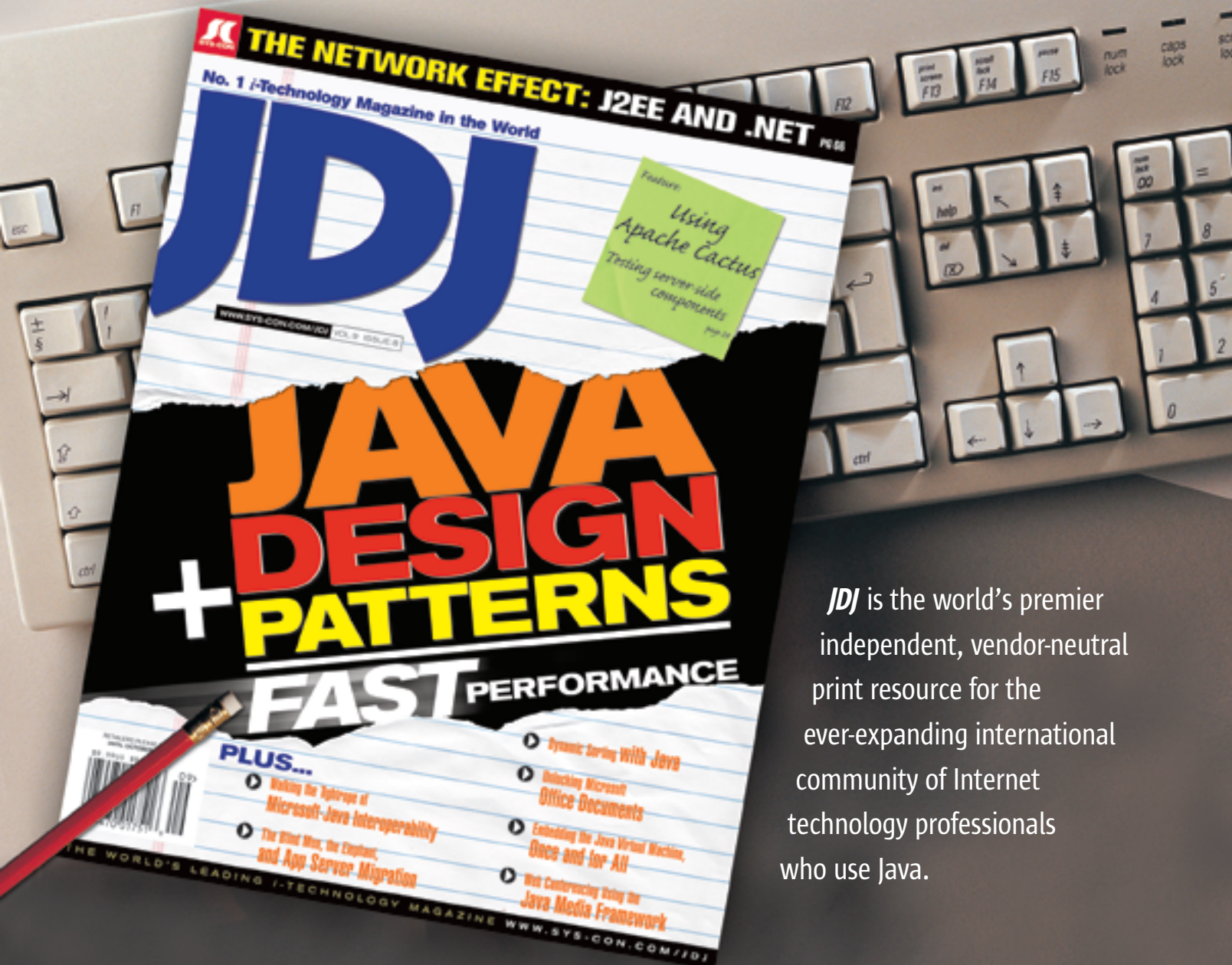
A solution to this problem is to choose a server-side UI model from the outset. Some Java-based RIA products follow this approach. These products offer a server-side proxy library that applies the *half-object + protocol design pattern* to the client-side UI widgets, as described in the article *Server-Side Swing for Rich Internet Applications*.

This server-side proxy approach fits well into the portlet infrastructure for the following reasons:

- **Architecture:** The architecture is the same as for HTML applications. The split between client and server is located in the presentation layer (see Figure 2). There is no application-specific code on the client that has to be saved and restored when portlets are switched.
- **Programming model:** The programming model is server-side, as with HTML applications. This means that the APIs for data exchange between portlets are easily accessible, so rich-client portlets can share business logic as well as data with HTML portlets.
- **Runtime model:** At runtime the UI is rendered by a presentation engine that's independent of an individual application, exactly like a browser. In a portlet context, this engine executes as an applet in

“Delivering rich-client applications as portlets is a major challenge except in approaches that rely on the half-object + protocol design pattern”

The World's Leading Java Resource Is Just a >Click< Away!



JDJ is the world's premier independent, vendor-neutral print resource for the ever-expanding international community of Internet technology professionals who use Java.

Only \$**69⁹⁹**

ONE YEAR
12 ISSUES

www.SYS-CON.com/JDJ
or **1-888-303-5282**

Subscription Price Includes **FREE** JDJ Digital Edition!

OFFER SUBJECT TO CHANGE WITHOUT NOTICE

**SYS-CON
MEDIA**
The World's Leading
i-Technology Publisher

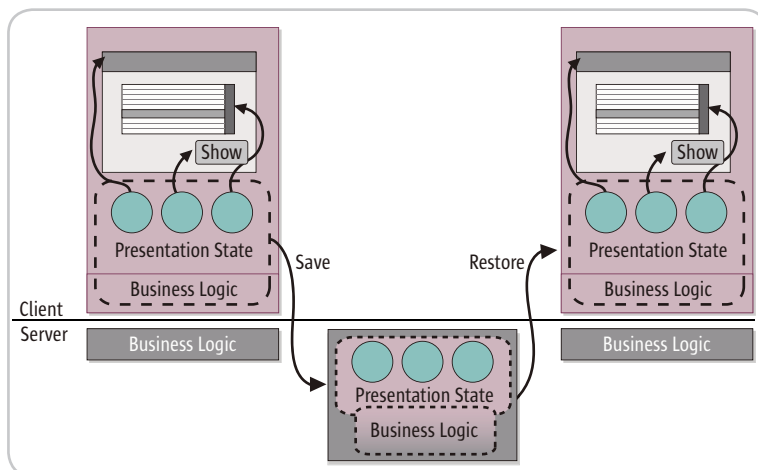


Figure 1 Saving and restoring the UI state when the entire presentation layer executes on the client

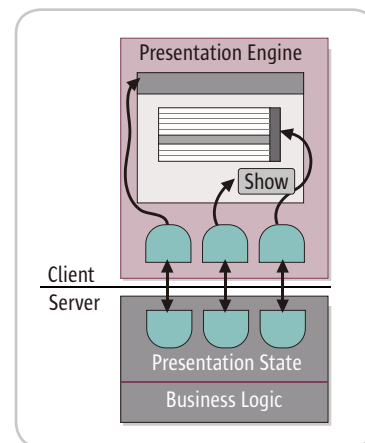


Figure 2 Client/server split in the presentation layer enabled by the server-side proxy solution

the browser (in other contexts, it can also execute in a standalone JRE). It will typically be cached by the browser and doesn't have to be loaded from the server when a portlet is started or re-activated.

- **Built-in synchronization:** The core of the half-object + protocol design pattern is the transparent and optimized synchronization of state between the peer objects across two address spaces. This means that the logic to synchronize the client-side UI widgets and their server-side peer objects comes with the base library. And it's typically optimized for minimal network traffic, transferring only the visible parts of a UI to the client, compressing data, etc. As a consequence, saving and restoring the client-side representation of the UI – as required when portlets are switched – is a task that can be executed efficiently and without a lot of additional and complex code.

Given this proxy approach, the portlet switching issue can be resolved easily and in a way that performs efficiently. The base library already takes care of synchronizing the client-side UI with the corresponding server-side model at runtime. When a portlet is left, the UI can be discarded because the server-side session maintains its state. When the same portlet is re-activated, all that's re-

quired is to re-start the applet, re-connect to the session, and retrieve and display the visible parts of the UI (see Figure 3).

Some of the products mentioned can execute this scenario. Typically, it's done with *pause* and *resume* methods, where the former does some cleanup that lets the UI be scrapped, and the latter restores the UI from the server.

Note that *pause* and *resume* are attractive for other scenarios as well, for example in situations where users want to switch their working place, or recover from a client crash. For this reason, pause and resume are ideally externalized in an API.

Moreover, the portlet integration can be wrapped in a configuration option, so that any application can be deployed as a servlet, EJB, or portlet without changing the application code.

Summing up: portlets are an attractive standard for developing Web portals. Delivering rich-client applications as portlets is a major challenge, except for approaches that rely on the half-object + protocol design pattern. Applied properly, this pattern enables portlet integration of Rich-Internet Applications (RIA) merely as a matter of the deployment configuration. ♦

References

- AltioLive: www.altio.com
- AppProjector: www.asperon.com
- Canoo UltraLightClient: www.canoo.com/ulc
- Classic Blend: www.appliedreasoning.com/products_what_is_Classic_Blend.htm
- Droplets: www.droplets.com
- NexaWeb: www.nexaWeb.com
- RSWT: <http://rswt.sourceforge.net>
- Thinlets: www.thinlet.com
- Gerard Meszaros: *Pattern: Half-Object + Protocol*; in *Pattern Languages of Program Design*; James O. Coplien and Douglas C. Schmidt, eds. Addison-Wesley, © 1995, ISBN 0-201-60734-4).
- Bernhard Wagner: *Server-Side Swing for Rich Internet Applications*: <http://javadesktop.org/articles/canoo/index.html>

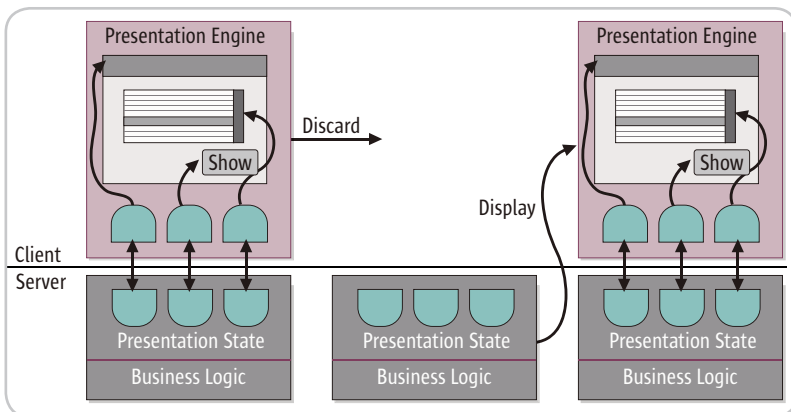


Figure 3 Portlet switching with the server-side proxy solution

A LIMITED TIME SAVINGS OFFER FROM SYS-CON MEDIA

SUBSCRIBE TODAY TO MULTIPLE MAGAZINES

AND SAVE UP TO \$340 AND RECEIVE UP TO 3 FREE CDs!*



RECEIVE
YOUR DIGITAL
EDITION
ACCESS CODE
INSTANTLY
WITH YOUR PAID
SUBSCRIPTIONS

3-Pack

Pick any 3 of our magazines and save up to **\$210⁰⁰**
Pay only \$99 for a 1 year subscription plus a FREE CD

- 2 Year – \$179.00
- Canada/Mexico – \$189.00
- International – \$199.00

6-Pack

Pick any 6 of our magazines and save up to **\$340⁰⁰**
Pay only \$199 for a 1 year subscription plus 2 FREE CDs

- 2 Year – \$379.00
- Canada/Mexico – \$399.00
- International – \$449.00

9-Pack

Pick 9 of our magazines and save up to **\$270⁰⁰**
Pay only \$399 for a 1 year subscription plus 3 FREE CDs

- 2 Year – \$699.00
- Canada/Mexico – \$749.00
- International – \$849.00

CALL TODAY! 888-303-5282

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. |

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.

☐ LinuxWorld 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 |

☐ JDJ

| | | |
|------------------------------------|---------------------|----------------------------|
| U.S. - Two Years (24) Cover: \$144 | You Pay: \$99.99 / | Save: \$45 + FREE \$198 CD |
| U.S. - One Year (12) Cover: \$72 | You Pay: \$69.99 / | Save: \$12 |
| Can/Mex - Two Years (24) \$168 | You Pay: \$119.99 / | Save: \$48 + FREE \$198 CD |
| Can/Mex - One Year (12) \$120 | You Pay: \$89.99 / | Save: \$40 |
| 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 |

☐ Information Storage + Security Journal

| | | |
|------------------------------------|--------------------|-----------------------------|
| U.S. - Two Years (24) Cover: \$143 | You Pay: \$49.99 / | Save: \$93 + FREE \$198 CD |
| U.S. - One Year (12) Cover: \$72 | You Pay: \$39.99 / | Save: \$39 |
| Can/Mex - Two Years (24) \$168 | You Pay: \$79.99 / | Save: \$88 + FREE \$198 CD |
| Can/Mex - One Year (12) \$84 | You Pay: \$49.99 / | Save: \$34 |
| Intl - Two Years (24) \$216 | You Pay: \$89.99 / | Save: \$126 + FREE \$198 CD |
| Intl - One Year (12) \$108 | You Pay: \$59.99 / | Save: \$48 |

☐ Wireless Business & Technology

| | | |
|------------------------------------|--------------------|----------------------------|
| U.S. - Two Years (12) Cover: \$120 | You Pay: \$49.00 / | Save: \$71 + FREE \$198 CD |
| U.S. - One Year (6) Cover: \$60 | You Pay: \$29.99 / | Save: \$30 |
| Can/Mex - Two Years (12) \$120 | You Pay: \$69.99 / | Save: \$51 + FREE \$198 CD |
| Can/Mex - One Year (6) \$60 | You Pay: \$49.99 / | Save: \$10 |
| Intl - Two Years (12) \$120 | You Pay: \$99.99 / | Save: \$20 + FREE \$198 CD |
| Intl - One Year (6) \$72 | You Pay: \$69.99 / | Save: \$2 |

☐ MX Developer's Journal

| | | |
|------------------------------------|--------------------|-----------------------------|
| U.S. - Two Years (24) Cover: \$143 | You Pay: \$49.99 / | Save: \$93 + FREE \$198 CD |
| U.S. - One Year (12) Cover: \$72 | You Pay: \$39.99 / | Save: \$32 |
| Can/Mex - Two Years (24) \$168 | You Pay: \$79.99 / | Save: \$88 + FREE \$198 CD |
| Can/Mex - One Year (12) \$84 | You Pay: \$49.99 / | Save: \$34 |
| Intl - Two Years (24) \$216 | You Pay: \$89.99 / | Save: \$126 + FREE \$198 CD |
| Intl - One Year (12) \$108 | You Pay: \$59.99 / | Save: \$48 |

☐ 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 |

☐ WebSphere Journal

| | | |
|------------------------------------|---------------------|----------------------------|
| U.S. - Two Years (24) Cover: \$216 | You Pay: \$129.00 / | Save: \$87 + FREE \$198 CD |
| U.S. - One Year (12) Cover: \$108 | You Pay: \$89.99 / | Save: \$18 |
| Can/Mex - Two Years (24) \$360 | You Pay: \$179.99 / | Save: \$80 + FREE \$198 CD |
| Can/Mex - One Year (12) \$120 | You Pay: \$99.99 / | Save: \$20 |
| Intl - Two Years (24) \$264 | You Pay: \$189.00 / | Save: \$75 |
| Intl - One Year (12) \$132 | You Pay: \$129.99 / | Save: \$2 |

☐ 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 |

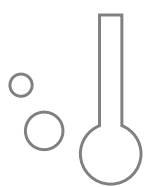
☐ WLDJ

| | | |
|-------------------------------------|--------------------|-----------------------------|
| U.S. - Four Years (24) Cover: \$240 | You Pay: \$99.99 / | Save: \$140 + FREE \$198 CD |
| U.S. - Two Year (12) Cover: \$120 | You Pay: \$49.99 / | Save: \$70 |
| Can/Mex - Four Years (24) \$240 | You Pay: \$99.99 / | Save: \$140 + FREE \$198 CD |
| Can/Mex - Two Year (12) \$120 | You Pay: \$69.99 / | Save: \$50 |
| Intl - Four Years (24) \$240 | You Pay: \$120 / | Save: \$120 + FREE \$198 CD |
| Intl - Two Year (12) \$120 | You Pay: \$79.99 / | Save: \$40 |

*WHILE SUPPLIES LAST. OFFER SUBJECT TO CHANGE WITHOUT NOTICE

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

SYS-CON
MEDIA



Adding Charts to Web Applications

Reviewed by
Yakov Fain

Give your Web app a facelift with WebCharts3D to make it a blockbuster

You can create a sophisticated application that implements complex algorithms, but in many cases you sell its GUI part to your users. The presentation layer of plain vanilla HTML/JSP-based Web applications is usually pretty basic unless you use specialized software. Today we are testing the charting engine WebCharts3D 5.0 from Green-Point.

Let me introduce myself: I am a Java developer with basic JSP skills and a stopwatch on my desk. My task is to show you how you can create a Web page in less than 20 minutes that retrieves the sales data from an external data source and displays them as a chart using WebCharts3D.

To set the scene, I downloaded and installed the beta version of the Tomcat 5.5 servlet container from Apache (<http://jakarta.apache.org/tomcat/>). Actually, WebCharts3D comes with an embedded Web server that can be used for test purposes, but I decided to use a commercial-grade server. I've also created a plain comma-delimited sales_data file with the annual sales data in a cross-tab form:



Yakov Fain is a Java 2 Enthusiast and Educator from Wall Street. He is the author of the best selling book, *The Java Tutorial for the Real World*, an e-book Java Programming for Kids, Parents and Grandparents.

Fain also authored several chapters for *Java 2 Enterprise Edition 1.4 Bible*. On Sundays Yakov teaches Java (see www.smartdataprocessing.com)

yakovfain@sys-con.com

| | | |
|-------------|------|---------|
| Alex | 2004 | 2000.00 |
| Mary | 2004 | 2870.00 |
| Sam | 2004 | 4900.90 |

I spent the next five minutes downloading and installing WebCharts3D 5.0 from www.gpoint.com.

Start WebCharts3D, and you'll see its Designer screen with a large variety of charts to choose from (some of them are shown in Figure 1). You can select one of the following: bar, pie, radar, dial, statistical, and Gantt charts.

Designing a Chart

Every chart consists of two parts: a style and a model (remember the MVC paradigm?). Internally, both

style and model are generated in XML format.

For this demo I've selected one of the flavors of a 3D bar chart (it's the third icon in the first row in Figure 1). This opened the screen shown in Figure 2. On the right side you can modify multiple properties of the chart starting with colors to event processing.

To populate the chart with the data from our sales file just go to the XML Model tab, check the Crosstab box, press the button "Import From File" and select our sales_data file. You'll see the table with our data and XML with hard-code sales values. Switch to the tab Design again to make sure that the chart is populated with our data.



Figure 1 WebChart3D chart gallery

“You can create a sophisticated program that implements complex algorithms, but in many cases it's the GUI part you sell your users”

Since we don't want to create charts with hard-coded data, go to the tab CodeView, press the Options button, and select the model "Imported File" (for databases you'd select the model "Database Query"). The Designer immediately generates the JSP code as shown in Listing 1.

Please note that the line `myChart.linkTo()` points at my `sales_data` file as the data source.

You can immediately see how the chart will look in your Web browser. Let's launch the embedded server: select the Console tab at the bottom of the Designer's screen and press the Start button and wait a second for the message "Server Started on port 8802." To see your Sales chart just open your Web browser and enter the following URL: <http://localhost:8802> as in Figure 3.

The next code snippet shows how you can get the data from a database using the JDBC-ODBC bridge:

```
Class.forName("sun.jdbc.odbc.
JdbcOdbcDriver");
Connection dbConn = DriverManager.
getConnection("jdbc:odbc:...",
"admin", "User Password");
myChart.linkTo(dbConn,"select
ProductName, Sales from Sales",false,
false);
```

Note the links at the bottom of the screen in Figure 3. You've guessed it right! For example, press the PDF link, and see your chart in Adobe Acrobat Reader if you have it installed.

Flash and SVG-Animated Charts

Some of the chart formats can be animated. To animate our sales chart using the Flash file format just select the SWF or SVG link while watching the chart in the embedded HTTP server (or set the chart type to SWF or SVG). You'll see how the bars start growing one after another. You can change their order of appearance by setting the Morphing attributes of your chart elements in the Designer. The charts can grow vertically or horizontally, you can add a blurring effect, and set the length and quality of your movie by playing with the `framesCount` and `framesPerSecond` properties. You can animate the entire chart or by row or by column. You can control the time of the animation, pauses between fades, etc. If you want to get a "wow" from your users, add the animation to your charts.

Deploying the Sales Chart Under Tomcat

Just follow the simple steps below to deploy our chart under Tomcat (the process is literally the same in any J2EE-compliant servlet container):

1. Copy `ws50.jar` into the `WEB-INF\lib` directory to make `WebCharts3D`-supporting classes available to Tomcat.

2. Create the file `getImage.jsp` in the root directory of the Web server. This file is used internally to retrieve the generated image temporarily stored on the server. Go to Designer's File | Setup menu, select the Server tab and enter the application home directory, which in the case of a default Tomcat installa-

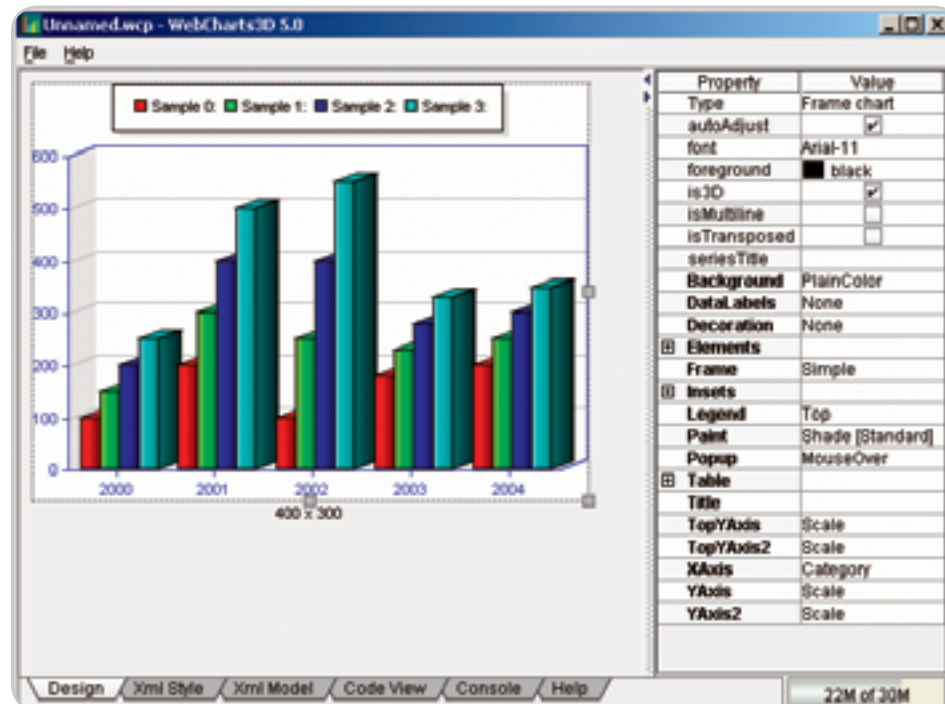


Figure 2 A sample 3D chart

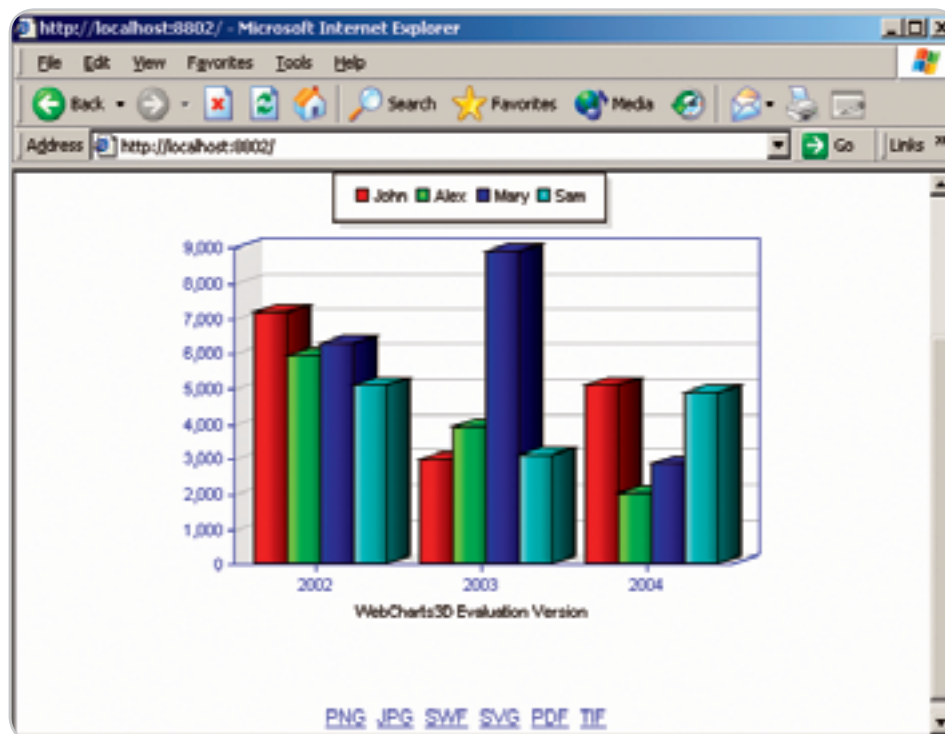


Figure 3 The Sales 3D chart produced by the embedded Web server

tion is C:\Program Files\Apache Software Foundation\Tomcat 5.5\webapps\ROOT

3. Restart Tomcat.
4. In the Designer's CodeView tab press the Save button and save the generated code shown in Listing 1 in Tomcat's root directory as sales.jsp. In the real world you'd copy and paste this code into one of your existing JSPs.
5. Point your browser at <http://localhost:8080/sales.jsp> and you'll see a screen as in Figure 4.

The yellow box is a cool feature called annotation – not to be confused with Java 5.0 annotations – that shows the underlying data as you move a mouse pointer over the chart. If you know Swing, it's similar to tooltips. Figure 4 shows the default annotation, but you can change the text the same way you'd change any other chart property.

The Web page is ready and my stopwatch shows that the entire process including the WebCharts installation took about 20 minutes. Of course, I've only demonstrated a small part of the WebCharts3D functionality, but the result is pretty impressive.

Processing Chart Events

Let's say you want to add a drill-down feature to your chart by opening a particular URL when the user clicks on the chart or one of its elements. In our sales example, it can be a separate JSP that shows some detailed information for each quarter. Just to prove that the chart can react to mouse clicks, open the Elements group in the Designer's properties table and add the following line in the Actions field:

```
javascript:window.alert('$({colLabel) $(row-Label) has been clicked').
```

Refresh the browser window and click on any element of the sales chart to see a pop-up window like "Sam 2002 has been clicked."

Other WebCharts3D Components and Features

- WebCharts3D comes with a JSF tag component that's integrated in all major JSF-enabled IDEs such as Java Studio Creator and JDeveloper. This component supports action and action-Listener properties. For example, you can program the onClick event and dynamically change

the chart's attributes such as dimensionality.

- When you create a chart in the Designer, it also generates the Java code that you can use to embed the chart in a rich client in Swing or SWT. After instantiating the Java MxComponent class, you can add it to any Swing container and treat it like a regular Swing component. Documentation includes a full description of the MxComponent API.
- A beta version of WebCharts3D Eclipse plug-in will be available this summer.

Overall, WebCharts3D is a mature product that's available on various platforms, and exists in Java and .NET versions. Internally, WebCharts3D uses image-caching algorithms that result in delivering charts to clients at an impressive speed. Give your Web application a face-lift with WebCharts3D to make it a blockbuster. ☼

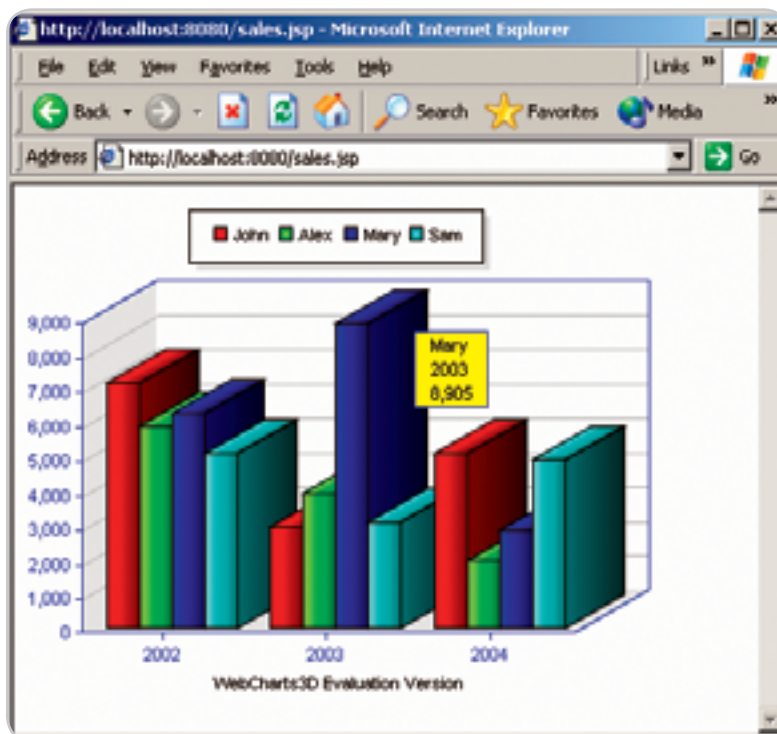


Figure 4 The Sales 3D chart (The yellow box is an annotation)

Listing 1: The JSP code generated by the WebCharts3D Designer

```
<%@ page import = "com.gp.api.jsp.
MxServerComponent" %>
<%@ page import = "com.gp.api.jsp.
MxChartDescription"%>
<%
    // This code can be embedded
    into any jsp page. See server sam-
    ples section.
    MxServerComponent svr =
    MxServerComponent.getDefaultInstanc
    e(application);

    MxChartDescription myChart =
    svr.newImageSpec();
    myChart.width = 400 ;
    myChart.height= 300 ;
    myChart.type = "png" ;
    myChart.style = " <frameChart>
    <frame yDepth=\"3\"/> <yAxis scale=
    Min=\"0\"/> <elements drawShadow=\\
    \"true\"/> <morph morph=\\\"Grow\\\"/>
    </elements> <paint isVertical=\\
    \"true\" min=\\\"47\" max=\\\"83\"/> </
    frameChart>" ;
    myChart.linkTo("C:\\
    sales_data",',','\\
    0',true,false,false,null);
    out.write(svr.
    getImageTag(myChart,"/getImage.
    jsp?image="));
    %>
```


| Advertiser | URL | Phone | Page |
|----------------------------------------|------------------------------------------------------------------------------------|------------------------|-----------|
| Altova | www.altova.com | 978-816-1600 | 4 |
| Borland | www.borland.com/jbuilder | 831-431-1000 | 7 |
| ceTe Software | www.dynamicpdf.com | 800-631-5006 | 43 |
| ClearNova | www.clearnova.com | 877-223-8651 | 33 |
| Common Controls | www.common-controls.com | +49 (0) 6151/13 6 31-0 | 41 |
| EclipseWorld Conference | www.eclipseworld.net | 631-421-4158 x101 | 53 |
| Google | www.google.com/jdj | 650-253-0000 | 37 |
| ILOG | http://diagrammer.ilog.com | 800-FOR-ILOG | 21 |
| InetSoft | www.inetsoft.com/jdj | 888-216-2353 | 25 |
| Information Storage & Security Journal | www.issjournal.com | 888-303-5282 | 61 |
| InterSystems | www.intersystems.com/free8p | 617-621-0600 | 17 |
| JadeLiquid Software | www.webrender.com | +61 3 6226 6274 | 23 |
| Java Developer's Journal | www.jdj.sys-con.com | 888-303-5282 | 55 |
| Jinfony Software | www.jinfony.com/jp6 | 301-838-5560 | 31 |
| LinuxWorld Conference & Expo | www.linuxworldexpo.com | 800-657-1474 | 51 |
| M7 | www.m7.com/power | 866-770-9770 | 27 |
| Microsoft | microsoft.com/connectedsystems | | Cover II |
| NCL | www.nclt.com/jdj | +353 1 6761144 | 39 |
| Northwoods Software Corp. | www.nwoods.com/go | 800-434-9820 | 47 |
| Parasoft Corporation | www.parasoft.com/jtest | 888-305-0041 | 13 |
| Perforce Software | www.perforce.com | 510-864-7400 | 11 |
| Prolifics | www.prolifics.com | 800-675-5419 | 49 |
| Quest Software | www.quest.com/javafuel | 949-754-8000 | Cover IV |
| ReportingEngines | www.reportingengines.com | 888-884-8665 | 15 |
| SleepyCat Software | www.sleepycat.com/bdbje | 510-597-2128 | 29 |
| Software FX | www.softwarefx.com | 800-392-4278 | Cover III |
| SYS-CON Publications | www.sys-con.com/2001/sub.cfm | 888-303-5282 | 57 |
| Visual Paradigm | www.visual-paradigm.com | 408-426-8212 | 35 |

General Conditions: The Publisher reserves the right to refuse any advertising not meeting the standards that are set to protect the high editorial quality of *Java Developer's Journal*. All advertising is subject to approval by the Publisher. The Publisher assumes no liability for any costs or damages incurred if for any reason the Publisher fails to publish an advertisement. In no event shall the Publisher be liable for any costs or damages in excess of the cost of the advertisement as a result of a mistake in the advertisement or for any other reason. The Advertiser is fully responsible for all financial liability and terms of the contract executed by the agents or agencies who are acting on behalf of the Advertiser. Conditions set in this document (except the rates) are subject to change by the Publisher without notice. No conditions other than those set forth in this "General Conditions Document" shall be binding upon the Publisher. Advertisers (and their agencies) are fully responsible for the content of their advertisements printed in *Java Developer's Journal*. Advertisements are to be printed at the discretion of the Publisher. This discretion includes the positioning of the advertisement, except for "preferred positions" described in the rate table. Cancellations and changes to advertisements must be made in writing before the closing date. "Publisher" in this "General Conditions Document" refers to SYS-CON Publications, Inc.

This index is provided as an additional service to our readers. The publisher does not assume any liability for errors or omissions.

Subscribe Today!

— INCLUDES —
FREE
DIGITAL EDITION!
(WITH PAID SUBSCRIPTION)
GET YOUR ACCESS CODE
INSTANTLY!



The major infosecurity issues of the day... identity theft, cyber-terrorism, encryption, perimeter defense, and more come to the forefront in ISSJ the storage and security magazine targeted at IT professionals, managers, and decision makers

SAVE 50% OFF!

(REGULAR NEWSSTAND PRICE)

Only \$39⁹⁹

ONE YEAR
12 ISSUES

www.ISSJournal.com
or 1-888-303-5282

JCP Recognizes the Best Member, Spec Leads, and Specifications

Onno Kluyt



Last month at the 2005 JavaOne Conference, members of the Java Community Process (JCP) program got together with the other Java communities for our traditional annual *Java Communities in Action* event. We mixed, mingled, and celebrated our ongoing efforts to extend Java technology on all fronts. On behalf of the JCP Executive Committees I handed out awards for JCP program achievements over the past year. The Executive Committees (ECs) had selected the honorees, which were easier to pick out since last year's migration to JCP version 2.6 encouraged the Expert Groups to work more transparently.

And the 2005 winners of the JCP Annual Awards are...

JCP Member of the Year

The Apache Software Foundation (ASF) was recognized as the JCP Member of the Year for having made the most significant positive impact on the community, in terms of leadership, investment in the community, and innovation. The ASF is a membership-based, nonprofit corporation that provides organizational, legal, and financial support for a large number of Java-based, open source software projects such as Apache Tomcat, Apache Maven, and Apache Ant. From the beginning, the organization has been an active member of the JCP program and the EC, and many Apache representatives work in the Expert Groups.

Geir Magnusson Jr., the ASF's representative on the EC, accepted the award on behalf of the ASF. Earlier he had said, "We focused on bringing the principles and benefits of community-based, collaborative open source to the JCP, helping put the 'Community' in Java Community Process." The ASF was instrumental in bringing about the changes that allowed for open source

Reference Implementations (RIs) and Technology Compatibility Kits (TCKs), as well as the creation of a scholarship program to provide free TCK licenses and support for qualified nonprofits, individuals, and scholars.

Most Outstanding Spec Lead for Java ME

Ekaterina Chtcherbina and Eric Overtoom accepted the Most Outstanding Spec Lead for Java Platform, Micro Edition (Java ME) award for their work as co-Spec Leads on Java Specification Request (JSR) 253 Mobile Telephony API (MTA). Eric is a distinguished member of the technical staff at Motorola and a member of the Java-Linux software platform architecture team for the mobile devices business. Ekaterina is a senior software architect at Siemens Corporate Technology. She learned effective spec leadership skills by observing her Siemens colleague and mentor, Jan Eichholz, while participating in his JSR 205 Wireless Messaging API 2.0 Expert Group. Eric began his involvement in the JCP program by co-leading JSR 253, drawing on his API development experience at Motorola. Ekaterina said earlier that with two Spec Leads working together, "the task of supporting effective communication can be fulfilled with a doubled energy."

Most Outstanding Spec Lead for Java SE/EE

Bill Shannon accepted the award for Most Outstanding Spec Lead for Java SE/EE. Bill is a distinguished engineer at Sun Microsystems, where he is one of the architects of Java EE. He has led several JSRs and is currently the Spec Lead for JSR 244 - Java EE 5, which aims to significantly improve ease of development for Java EE developers, especially in the area of Web service applications.

The demand for this version compels Bill to keep the feature set tightly focused in order to ship before the end of 2005. Maintaining a schedule is tricky because at least six other JSRs directly support the goals of Java EE 5, so Bill keeps tabs on the progress of those JSRs. Bill has been with Sun since 1982 and previously worked on the JavaMail API, HotJava Views product, the Solaris operating system, and all versions of SunOS.

Most Innovative JSR for Java ME

JSR 271 Mobile Information Device Profile 3 (MIDP3) earned the Most Innovative JSR for J2ME award. Spec Lead Jim Van Peurse of Motorola accepted the award. MIDP3 builds on the success of MIDP2 by enhancing the profile with features that include over-the-air deliverable-shared libraries for MIDP. This enables a new market for MIDP middleware developers; the ability to run multiple concurrent MIDlet suites simultaneously within a single VM; and the ability to run MIDlets on CLDC, CDC, or OSGi environments. All of this is being added while maintaining the important goal of backward compatibility with MIDP2 content.

Most Innovative JSR for Java SE/EE

JSR 175 - A Metadata Facility for the Java Programming Language was awarded Most Innovative JSR for Java SE/EE. Bill accepted the award on behalf of Sun Microsystems. This JSR enables a new declarative style of programming that will simplify many programming tasks by allowing classes, interfaces, fields, and methods to be marked as having particular attributes.

The 2005 winners join the JCP Hall of Fame of previous years' winners. Last year's winners can be viewed at www.jcp.org/en/press/news/2004JavaOne/awards2004.

Onno Kluyt is the director of the JCP Program Management Office, Sun Microsystems, and Chair of the JCP.

onno@jcp.org

SoftwareFX

Zero To Chart

In Less Than An Hour!



9:04 am

To learn more about the Chart FX products visit www.softwarefx.com and decide which one is right for your platform and target. Then download the 30-day trial version or the Chart FX for Java Community Edition which is a FREE, non-expiring, full development version.

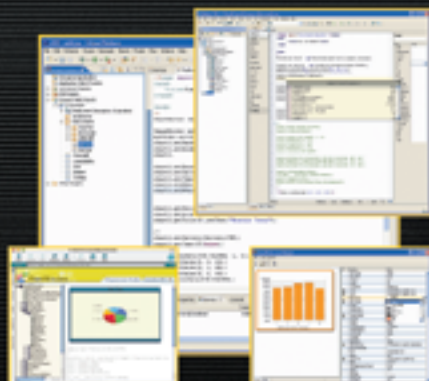
9:07 am

After download, simply install the product appropriate for your needs, platform and target environment.



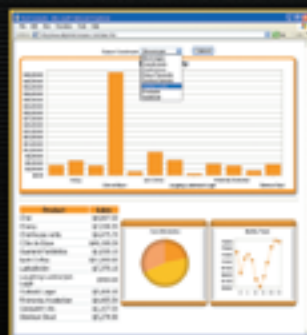
9:13 am

Open the Chart FX for Java Designer to get started with the look and feel of your chart. Use your favorite IDE to add functionality and to populate the chart by connecting to your data source. The Chart FX Resource Center is also available to help with any questions you may have. Then deploy to your specific application server. ➤



9:58 am

Your application is then displayed with an active chart or image that makes any data look great!



Ready... Set... Download!


Chart FX

US: (800) 392-4278 • UK: +44 (0) 8700 272 200 • Check our website for a Reseller near you!

www.softwarefx.com

©2005 Software FX. All rights reserved. Chart FX is a registered trademark of Software FX, Inc. All other brands are owned by their respective owners.



Refuel your Java Pit Crew.

Give them the right tools at the right time with J2EE Solutions from Quest Software.

Your entire crew can depend on Quest's J2EE solutions to resolve issues like memory leaks, performance bottlenecks, and unplanned downtime.

Optimize code quality. Improve performance and availability. Simplify change and configuration management. All with the expertise and speed your team needs to compete successfully.

Find it. Fix it. Change it. Get back in the race with Quest Software.

To learn more about our J2EE solutions, download your free white paper entitled: *Strategies for Architecting High Performance J2EE Applications* at:
www.quest.com/JavaFuel
